Implementation Guide

Success by Design

Microsoft Dynamics 365

Microsoft Dynamics 365

Success by Design Implementation Guide

Published by Microsoft Corporation, 1 Microsoft Way, Redmond, Washington 98052.

Information contained in this book may be updated from time to time, and you should refer to the latest version for the most accurate description of Microsoft’s FastTrack for Dynamics 365 program.

Case studies used in this book are fictional and for illustrative purposes only.

First Edition

Printed in the United States of America

This book is dedicated to Matthew Bogan, our colleague and friend. The way you think and the way you approach everything you touch embodies a growth mindset—that passion to learn and bring your best every day to make a bigger difference in the world. Thank you for being a shining example.

What's inside

Section

Strategy

1. Introduction to Implementation Guide
2. Success by Design overview
3. Implement cloud solutions
4. Drive app value
5. Implementation strategy

1

Guide

Introduction to Implementation Guide

"Our success is dependent on our customers' success, and we need to obsess about them—listening and then innovating to meet their unmet and unarticulated needs. No customer of ours cares about our organizational boundaries, and we need to operate as One Microsoft to deliver the best solutions for them."

– Satya Nadella, Chief Executive Officer of Microsoft

Overview

Microsoft believes that every business is in the business of creating great customer experiences.

As a precursor to the Dynamics 365 implementation resources in this book, this chapter will:

- Explore the opportunities presented as organizations shift into the predictive era.
- Outline our vision for Dynamics 365.
- Help readers understand what an investment in Dynamics 365 invites in terms of enabling holistic, unified digital transformation.

To achieve that, business applications must do more than just run your back office, marketing, supply chain, or even field operations as discrete entities. Organizations need to tap into data signals from across their business—and use them to drive ongoing improvements in the products, services, and experiences they intend to deliver to their customers.

Introduction

In this chapter, we explore how the shift from the reactive past to the predictive era is currently affecting our industry—and will eventually affect every industry. We also look at Microsoft's vision for Dynamics 365 and how our apps allow holistic, unified digital transformation. In our view, these topics are essential to anyone about to embark on an implementation of Dynamics 365.

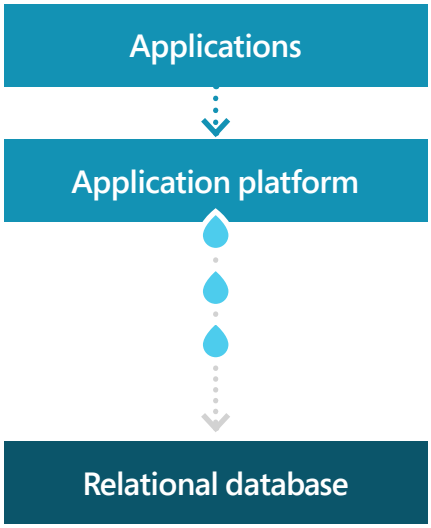
This chapter's content is adapted from the combined thinking of James Phillips, President of Microsoft's Digital Transformation Platform Group; Rolf Harms, Corporate Vice President of Microsoft's Cloud and AI Strategy; and Muhammad Alam, Microsoft's Corporate Vice President of Dynamics 365.

Armed with these perspectives, the rest of this book—which aims to share the mindset and guidance needed to successfully implement Dynamics 365—is sure to take a more meaningful shape for readers. For Microsoft, this is especially important, considering the investment that you are making in us.

Every industry turned upside down

We find ourselves at a turning point: over the coming decades, the role of technology will be turned upside down with profound implications for every aspect of business—from customer experiences to core operational processes to the very nature of products and services.

Fig. 1-1



Picture this: in the reactive past (**Figure 1-1**), a customer purchases a car from a dealer. One day, while going down the road, the car stops running. The customer calls the service center and reports a problem. At this point, a service advisor opens an application, types a few notes about the problem into a form, and then schedules a repair visit. It may be the first time in years that any data related to the customer or the vehicle has flowed—a few kilobytes trickling down, and only in reaction to the car's malfunction—into the dealer's relational database. Only now the dealer can provide service to the customer, but the car's manufacturer might never see a single byte of data from this event. As this example illustrates, in the reactive past, data flows downward from applications at only a drip.

Now consider what is possible today: the same customer purchases a car from the dealer. With appropriate permissions granted by its new owner, the car provides a continuous stream of data back to the dealer and the manufacturer. It may be a self-driving car, with computer vision models trained in the cloud via data captured by the entire fleet's cameras. Every operating parameter of the engine, every press of the brake, every trip route—all are signals that provide a rich basis for predictions to ensure a higher-value relationship between the customer, the dealer, and the car's manufacturer.

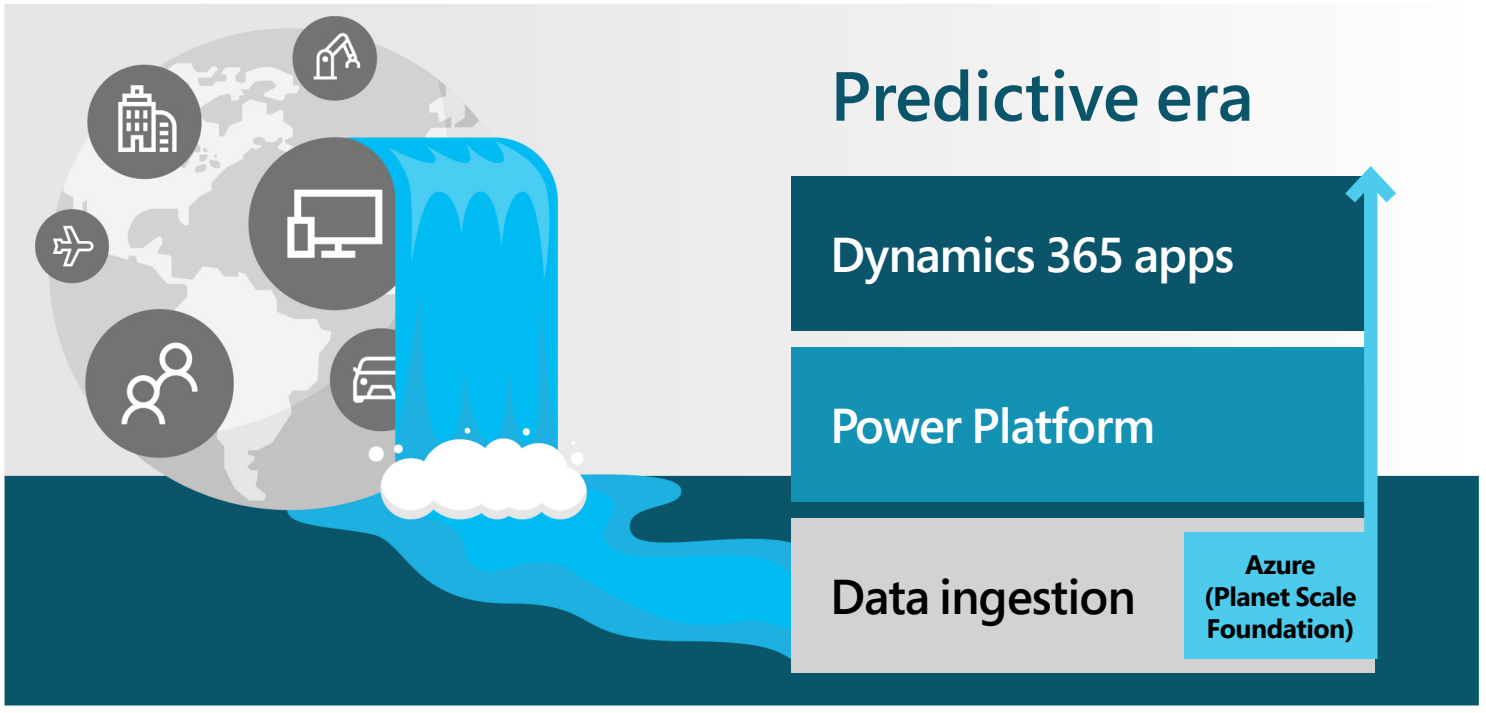
In Microsoft's view, the possibility of a higher-value relationship is present for every product, every service, and every business process. In the predictive era (**Figure 1-2**), the paradigm is that **everything is a source of data and potential insights**. As organizations embrace this shift, they enable an upward data flow that originates from everywhere and, at the same time, informs their products, services, and applications.

For companies that "grew up" in the predictive era, that paradigm is second nature. But for mature enterprises, the path can be more difficult, as it requires transformation of existing capabilities and processes.

For Microsoft, this isn't just theory or prediction. We have been pursuing our own transformation and have faced many of these challenges ourselves. At the heart of our transformation is the way we build products. Instead of the multiyear product cycle of traditional, on-premises products, Dynamics 365 is delivered as a service that's continuously refined and always up to date. It also allows us to capture rich telemetry on bugs and usage to inform ongoing feature development.

Beyond its influence on the way we run our own business, this digital transformation paradigm has affected how we think about success for our customers. Microsoft's vision has always been about democratizing

Fig. 1-2



software and technology capabilities to empower everyone. Looking forward, we have renewed our purpose in the context of the cloud in a world where everything is a source of data and potential insights.

Microsoft's mission is "to empower every person and every organization on the planet to achieve more." Harnessing data to help you predict the future and intelligently guide actions represents a core capability that invites organizations to achieve more by innovating everywhere (**Figure 1-3**).

Microsoft's data-first cloud strategy

The opportunity contained in the shift from reactive to predictive—knowing that the car is on its way to failure before it fails—is hard to overstate. For Microsoft and for our customers, every product, service, and business process is ripe for reinvention. But the reinvention requires a wholesale reimagination—a literal turning upside down—of the systems that power the interconnected fabric of products, services, and applications.

To help companies reinvent their businesses and thrive in the predictive era, Microsoft has invested in applications (such as Dynamics 365 Sales), an application platform (Microsoft Power Platform), and an infrastructure platform (Microsoft Azure) spanning from enterprise and independent software vendor (ISV) developers to citizen developers.

Fig. 1-3

Innovate everywhere

Suppliers
Create a resilient, efficient supply chain.

Assets
Get the most from your assets.

Operations
Optimize your operations.

Employees
Empower your people and teams.

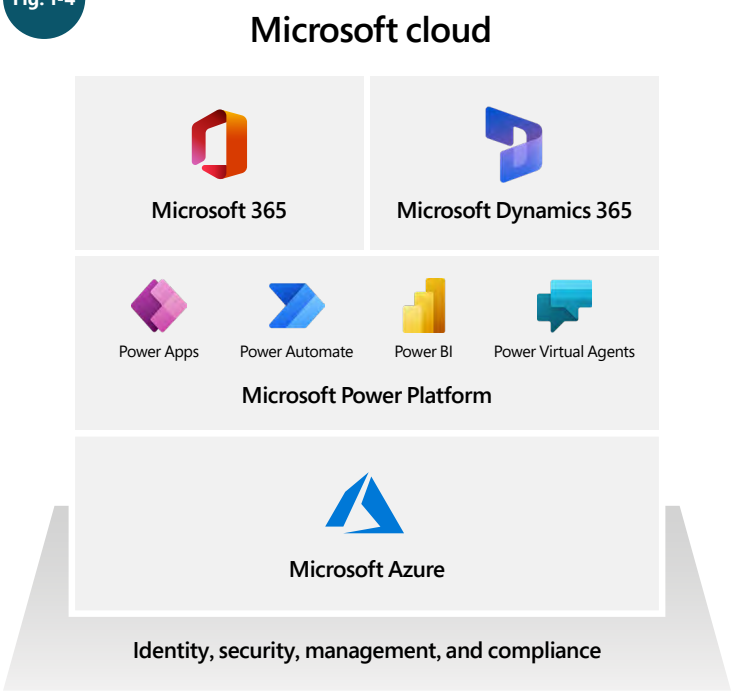
Products and services
Connect your products and services.

Customers
Delight your customers from first touch to delivery to success.

The entire Microsoft cloud (**Figure 1-4**) comes together as a unified digital-transformation platform with consistent security, identity, and compliance boundaries.

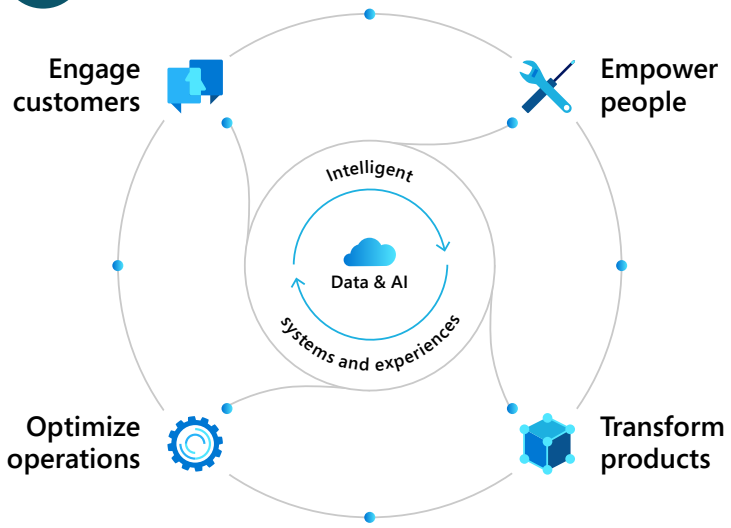
With everything as a source of data and potential insight, Microsoft's goal is simple: help customers take the right action at the right time with the right message via the right channel to achieve the right business outcome. (Throughout this book, you may find that we refer to this concept as the digital feedback loop, as shown in **Figure 1-5**. It demonstrates how organizations tap into data signals from across their business, and use those signals to drive ongoing improvements in the products, services, and experiences delivered to their customers. The customers then respond to these improvements, thus fueling a self-reinforcing cycle.)

Fig. 1-4



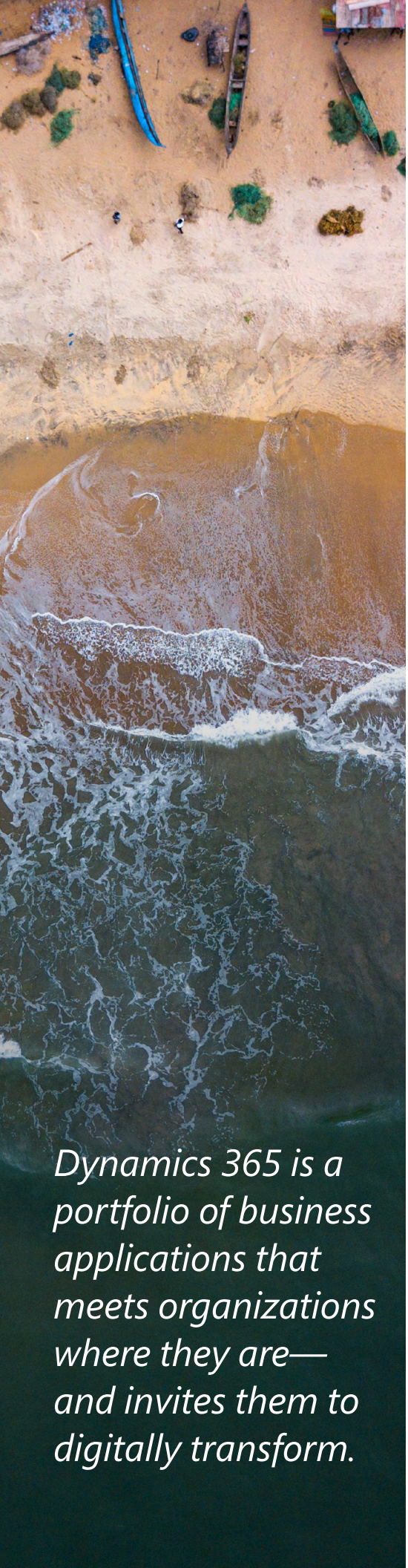
Dynamics 365: Cloud-based business apps for holistic, unified digital transformation

Fig. 1-5



Keeping the opportunities of the predictive era top of mind, we want to take a step back and focus on what Microsoft got right with Dynamics 365, further explain our data-first strategy in terms of Dynamics 365, and provide advice to customers embarking on an implementation of Dynamics 365.

Looking back, it's clear that Microsoft's company-wide push into the cloud set in motion the trajectory



Dynamics 365 is a portfolio of business applications that meets organizations where they are—and invites them to digitally transform.

of what we know today as Dynamics 365. But also notable was Microsoft's decision to embark on a business applications journey into the cloud with its previously named Dynamics CRM and Dynamics AX products.

Although they were separate applications at the time, Microsoft's decision to move forward with Dynamics CRM and AX foreshadowed the importance we place on developing a cloud-based business application solution that offers customers a holistic, unified digital-transformation platform covering the front office, the back office, and beyond.

Fast forward to today, and few of Microsoft's competitors can offer a complete cloud business application with the breadth, depth, and level of integration with existing Microsoft products that Dynamics 365 has.

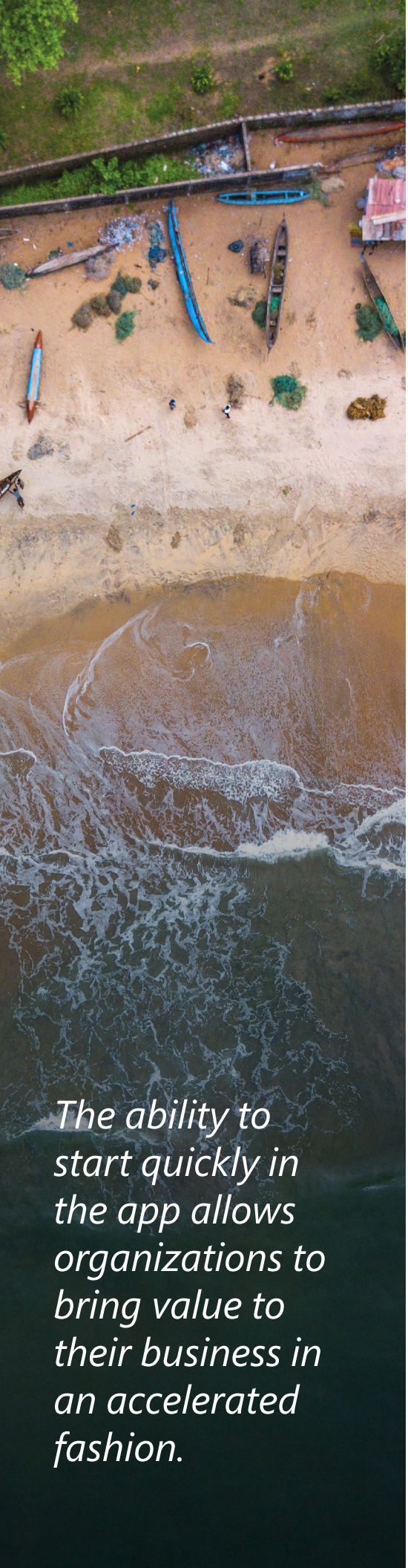
Dynamics 365 is a portfolio of business applications that meets organizations where they are—and invites them to digitally transform. Each app is designed to remove the barriers and eliminate silos within organizations by working together with existing systems—or the entire portfolio of Dynamics 365 apps—for outcomes you simply can't get unless every part of the business is connected seamlessly.

Microsoft's data-first Dynamics 365 strategy

If you're reading this book, then your business likely has already invested in Dynamics 365. However, understanding Microsoft's future vision for Dynamics 365 will help you take full advantage of your investment.

Building on Microsoft's cloud strategy (which includes Dynamics 365), several other factors contribute to Microsoft's data-first strategy to put Dynamics 365 into a category of one:

- Dynamics 365 provides front-office and back-office cloud applications that are consumable in a composable manner, which means that our customers can maximize their investment without having to do an extensive rip and replace of what were historically behemoth customer relationship management (CRM) and enterprise resource planning (ERP) processes.
- Dynamics 365 is built upon the low-code Power Platform to enable pro and citizen developers to build solutions, automate processes, and generate insights using Power Automate, robotic



The ability to start quickly in the app allows organizations to bring value to their business in an accelerated fashion.

process automation (RPA), and Power BI—which no other vendor can offer with its core platform.

- All of these are natively built on Azure, the cloud with an unmatched level of security, trust, compliance, and global availability.
- Add to this the assets of Office 365 (such as Microsoft Teams, LinkedIn, and Bing), which can be used to enrich the business application experience.

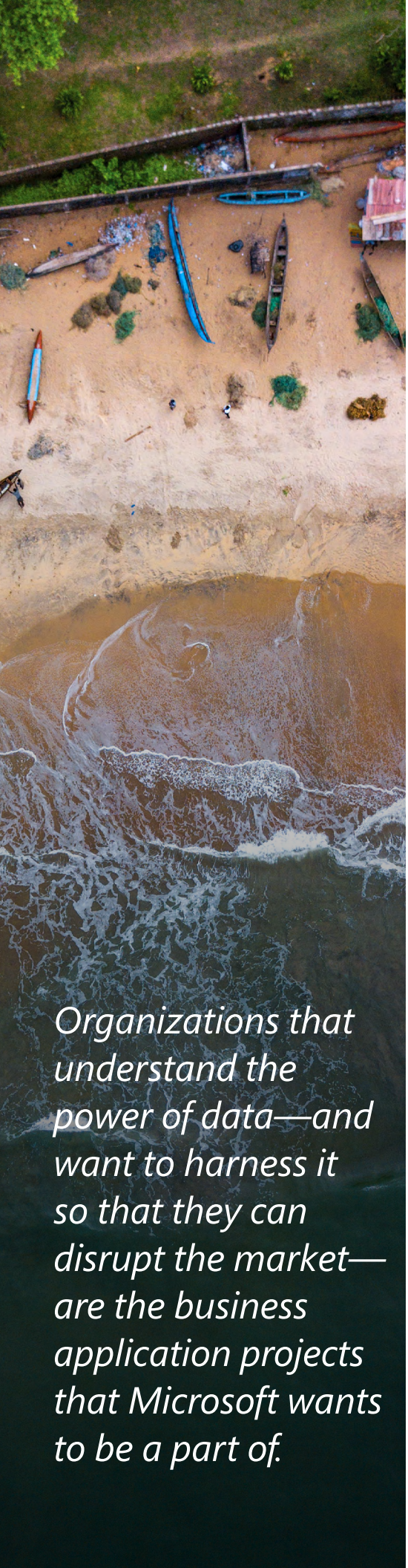
With Dynamics 365, organizations have the only portfolio of intelligent business applications that enables holistic, unified digital transformation and empowers everyone to adapt and innovate—anywhere.

Advice to customers embarking on an implementation of Dynamics 365

Microsoft's ideas on what customers should keep doing—whether they are embarking on an implementation of Dynamics 365 or any other business application—are detailed later in this book, but they include many of the traditional activities, such as business-process definition, quality-requirements management, end-user readiness, and change management.

However, Microsoft recommends against project teams that structure their engagements as massive 9-month to 18-month waterfall efforts. From a cloud services perspective, this old, on-premises implementation style needs to be jettisoned in favor of agility.

The ability to start quickly in the app allows organizations to bring value to their business in an accelerated fashion. The focus of project teams should be on finding the function or area of the application that the business can start using as quickly as possible, and then expand and enhance from there. The application and the business are going to change anyway, so it is in the best interest of organizations to claim value early.



Organizations that understand the power of data—and want to harness it so that they can disrupt the market—are the business application projects that Microsoft wants to be a part of.

More pointedly, Microsoft's view is that being agile does not mean engaging in sprint cycles that still require a 9-month to 18-month team effort to deploy part of the application to end users.

Traditional business application models can no longer keep up. Long setup times, expensive and complicated customizations, and slow adoption curves make it nearly impossible to meet the needs of a changing and demanding customer base.

Think beyond the implementation—and beyond the technology

The key question as you embark on an implementation of Dynamics 365: why are you doing it? Is it just a rip-and-replace effort, or does the implementation represent an opportunity for digital transformation, advance your business model, allow you to leapfrog over the competition, or something else?

In making an investment in business applications, organizations are called upon to clearly identify the value they intend to drive toward. Will the technology get you there? Does the software or the implementation partner have the vision to get you there?

An investment in business applications requires a level of organizational maturity to think about what you and your business are after. It also requires you to think beyond just the implementation of technology.

Organizations that understand the power of data—and want to harness it so that they can disrupt the market—are the business application projects that Microsoft wants to be a part of. This is where we believe the world is going, and our products reflect that.

The journey forward

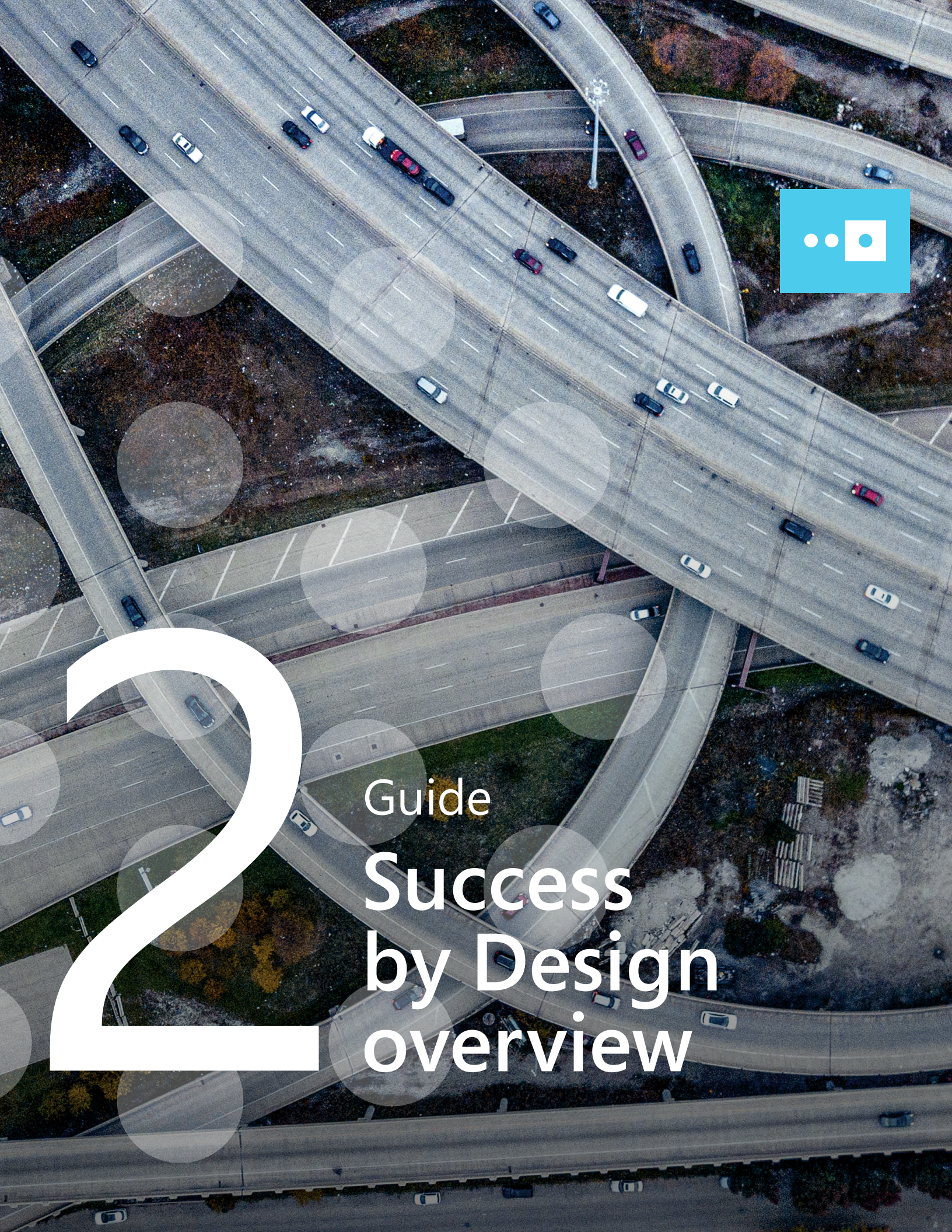
We understand that every organization comes with a unique set of challenges, opportunities, and priorities. No matter where you are in your digital transformation, Implementation Guide offers the

guidance, perspective, questions, and resources to help you drive a successful Dynamics 365 implementation, and harness data for insights that lead to the right actions and the right business outcomes across any business process.

The first section of Implementation Guide provides an overview of the Success by Design framework, which Microsoft created to help our customers and partners drive successful implementations of Dynamics 365. Additional chapters in the first section focus on the mindset and thinking required to implement in the cloud.

Subsequent sections of Implementation Guide address the major topics in the Dynamics 365 implementation lifecycle, including Microsoft's point of view on solution design, environment strategy, data, security, business intelligence, integration, performance, user readiness, support, and much more.

Whether you're a first-time implementer or a been-there-done-that Dynamics 365 architect, we hope you'll bring us along on your journey.



2

Guide

Success by Design overview

Success by Design overview

This chapter introduces Success by Design—a framework and practice created by Microsoft to help project teams implement Dynamics 365.

One of the primary goals of this book is to democratize the practice of Success by Design by making it available to the entire community of Dynamics 365 implementers.

Based on thousands of real-world customer projects, Success by Design is the sum of our Dynamics 365 implementation experience. It offers topic-specific reviews and prescriptive guidance (approaches and recommended practices), which provide a reliable path to project success. Success by Design is intended to be used by Dynamics 365 system integrators, independent software vendors (ISVs), and customers as a means to better architect, build, test, and deploy Dynamics 365 solutions.

For our customers, Microsoft recognizes that Success by Design doesn't guarantee implementation outcomes, but we're confident that it will help you achieve your project's goals while enabling the desired digital transformation for your organization.

For our partners, Microsoft is confident that Success by Design, coupled with your implementation methodology and skilled resources, will increase your team's effectiveness in delivering successful Dynamics 365 projects to your customers.

Success by Design history

As demand for Dynamics 365 cloud services increased across the enterprise, Microsoft identified the clear need to change the way we thought about evolving our services and our responsibility to customers and partners. We recognized that it wasn't enough to design a platform containing a set of features. We needed to also understand what it takes to deliver a fully functioning, end-to-end solution that runs the mission-critical processes of our customers' businesses. From this need, Success by Design was born.

Microsoft believes that customer success is the precursor to every Dynamics 365 product development decision. Because of this, questions fundamental to product development now challenge our engineers at every step of the process:

- In what way must each product feature be delivered to ensure customer success?
- How do we make Dynamics 365 apps valuable, durable, reliable, and performant?
- What aspects of the product should be jettisoned because they don't meet fundamental customer success criteria?

For the first time in the product's history, such questions have led to the successful transition of 100 percent of Microsoft Dynamics 365 online customers to one version, to a dependable and safe deployment process, to a steady and coherent feature release cadence (including many overdue feature deprecations), and to a reliable and performant platform that delivers customer value. But product is only one half of the customer success equation.

Microsoft has put equal emphasis on the need to provide you the prescriptive guidance and recommended practices to ensure a smooth implementation project and a properly designed and built Dynamics 365 solution that successfully transforms business operations. This push has also resulted in the transformation of Microsoft's FastTrack for Dynamics 365 service, which makes certain that our community of Dynamics 365 implementers—customers and partners—has access to Success by Design.

FastTrack for Dynamics 365 is a customer success program run by Microsoft's Dynamics 365 product engineering team that helps customers implement Dynamics 365 apps and realize business value faster. The practice of Success by Design is fundamental to FastTrack's approach, but the Success by Design framework is available for use on any Dynamics 365 implementation project.

This chapter focuses on the fundamentals and practice of Success by Design and its desired result: Dynamics 365 projects whose technical and project risks are proactively addressed, solutions that are roadmap aligned, and project teams that are successful in maximizing their organization's investment in Dynamics 365 cloud services.

Make Success by Design your own

Enabling Dynamics 365 project teams to practice Success by Design means an ongoing commitment by Microsoft to provide the community of implementers—customers and partners—best-in-class business applications, along with the latest in Success by Design project resources.

It also means project team willingness to incorporate the practice of Success by Design into their overall project approach—regardless of implementation methodology or the Dynamics 365 product being implemented. At its best, Success by Design fortifies a project team's chosen implementation methodology with a model for product-aligned project governance.

No matter where you find yourself in your Dynamics 365 implementation, fundamental to Success by Design is the willingness of project teams—despite the common pressures of time, budget, and resources—to pause in order to understand and address technical and project risks before it's too late in the project lifecycle.

To this end, this chapter focuses on the what and the why of Success by Design, as well as how project teams can use it to accelerate customer success throughout the implementation of Dynamics 365.

Success by Design objectives

Success by Design is prescriptive guidance (approaches and recommended practices) for successfully architecting, building, testing, and

deploying Dynamics 365 solutions. Success by Design is informed by the experience of Microsoft's FastTrack program, which has helped our customers and partners deliver thousands of Microsoft's most complex Dynamics 365 cloud deployments.

Success by Design reviews are exercises in reflection, discovery (observation), and alignment (matching to known patterns) that project teams can use to assess whether their implementation project is following recommended patterns and practices. Reviews also allow project teams to identify (and address) issues and risks that may derail the project.

Success by Design should be used as an adjunct to the project team's chosen implementation methodology for the following benefits:

- Reduced risk due to early detection of problems
- Alignment to recommended practices

The result is a roadmap-aligned solution architecture that is performant, scalable, and future-proof. [1]

In this book, we use the term "Finance and Supply Chain Management" to refer to the collective category of Dynamics 365 apps that includes Finance, Supply Chain Management, Commerce, Human Resources, and, in some cases, Project Operations.

For interested customers, Microsoft recommends that project leaders team up with their implementation partner to enable Success by Design within their project. In addition to the Success by Design resources available in this book, it's highly recommended that project teams ready themselves by enrolling in the Success by Design training on Microsoft Learn.

Success by Design phases

Success by Design maps the Dynamic 365 implementation lifecycle into four methodology-agnostic phases: Initiate, Implement, Prepare, and Operate (**Figure 2-1**). In this section and the following sections, we outline the Success by Design phases, their relationship to Success by Design reviews, and the desired outputs and outcomes.

Fig. 2-1

In the Initiate phase, the project team is in discovery mode—gathering and validating business requirements, finalizing the high-level solution approach, making inroads to define all in-scope workstreams, and updating the project plan to reflect these updates. When the project team has produced the high-level solution design and the related project

[1] Acknowledging the possibility that feature deprecations or other changes to product roadmap in compliance with Microsoft policy may occur.

workstreams are more or less defined, Success by Design begins with the Solution Blueprint Review. (More on the Solution Blueprint Review and other reviews later in this chapter.)

In the Implement phase, the project team is focused on building the solution per the agreed-upon solution design and scope. Implementation Reviews are introduced in this phase, having been informed by the findings and recommendations of the Solution Blueprint Review. As we learn later in this chapter, Implementation Reviews are used to more deeply address questions related to the specific aspects of the solution design (data model, security, integration) and implementation practices (ALM, testing strategy). Implementation Reviews are meant to fully address the risks identified during or after the Solution Blueprint Review but before the solution build is too far along.

By the Prepare phase, the solution has been built and tested and the project team is preparing for the final round of user acceptance testing (UAT) and training. Additionally, all necessary customer approvals have been granted, information security reviews completed, the cutover plan defined (including go/no-go criteria), mock go-lives scheduled, the support model ready, and the deployment runbook completed with tasks, owners, durations, and dependencies defined. At this point, the project team uses the Success by Design Go-live Readiness Review to identify any remaining gaps or issues.

In the Operate phase, the customer solution is live. The goal of this phase is stabilization and a shift in focus towards functionality and enhancements that are earmarked for the next phase of the project.

Success by Design reviews

With a high-level understanding of Success by Design phases, we now turn to Success by Design reviews.

Each review raises questions that serve as points of reflection that project teams can use to generate important discussion, assess risk, and confirm that best practices are being followed.

Later chapters of this book will cover each Success by Design review in greater detail. For the most comprehensive coverage of Success by Design, refer to the Success by Design training on Microsoft Learn.

The Solution Blueprint Review serves as the starting point of Success by Design. We suggest that the Solution Blueprint Review be a mandatory review for the project because findings that come from it lead to Implementation Reviews, which offer project teams the opportunity to drill down into topic-specific areas where deeper dives are deemed necessary for further understanding project and solution risk. Finally, the Go-live Readiness Review, which we also suggest as a mandatory review, is the last stop for assessing any remaining risks before go-live.

Figure 2-2 illustrates that Success by Design reviews are not to be conducted as abstract exercises separated from the project. Rather, the scheduling and implementation of each review relies on the availability of key project artifacts and the readiness of the project team to discuss them. (For a more in-depth look at Figure 2-2, visit the Appendix, which provides Microsoft's view of the typical deliverables, activities, tasks, and stakeholders involved in each Success by Design phase.)

Success by Design outputs

With a basic understanding of Success by Design's objectives, phases, and review flow, it's important to pause to understand the makeup of review outputs and their purpose—findings and recommendations.

Fig. 2-2

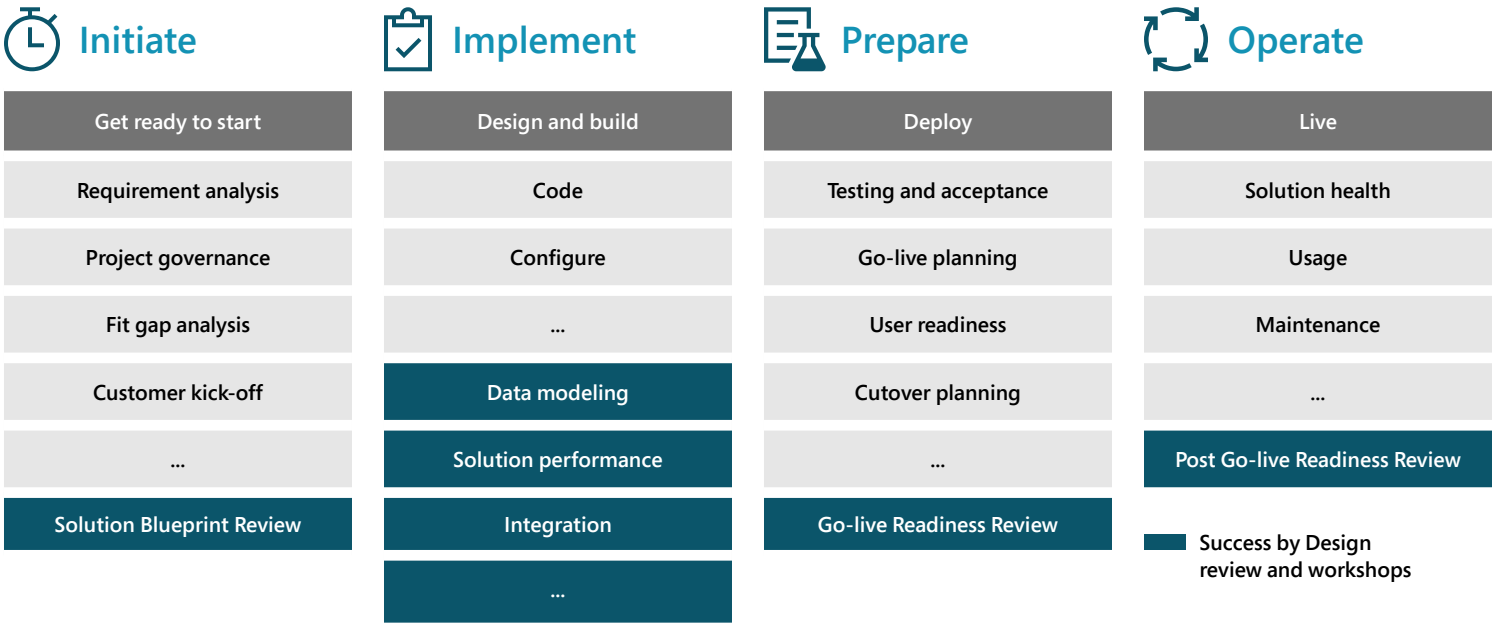
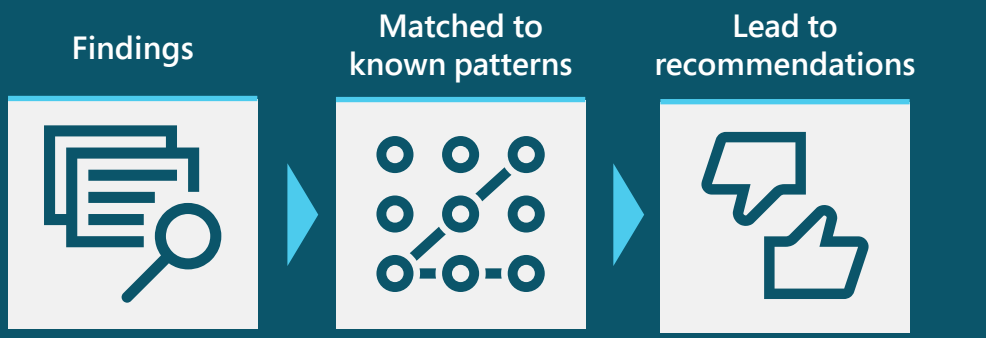


Fig. 2-3



As described in the previous section, reviews are informed by project artifacts (project plans, requirements, fit gap, solution architecture, design documents, and so on) produced by the project team, and other information such as formal or informal project touchpoints. The availability of such information points to the readiness of the team to schedule and conduct Success by Design reviews. Additionally, project teams may rely on Microsoft for telemetry and other tools to inform review discussions and generate review outputs.

The primary review outputs fall into two related categories: findings and recommendations.

Findings come in three types:

- **Assertions** Findings that capture noteworthy aspects of the solution or approach. Assertions highlight what the project team is doing right, typically in line with best practices.
- **Risks** Findings that could potentially impact the implementation negatively if not mitigated.
- **Issues** Findings that are currently impacting implementation negatively or will do so if not resolved.

Findings should include as much detail as possible and be matched to known patterns. Findings matched to known patterns often yield insights that lead to recommendations or actions necessary to resolve the issues identified (**Figure 2-3**).

To better illustrate this point, consider this customer example:

A global corporate travel company is implementing Dynamics 365 Customer Service to drive its call center transformation. As the project team digs into the business's requirements, they learn that the Dynamics 365 solution must account for integration with multiple legacy systems

(many with high transaction volumes). Additionally, the requirements point to a business-mandated customization that the project team agreed could not be achieved using out-of-the-box functionality.

In preparation for the Solution Blueprint Review, the project team parses these and other details, including confirming that solution performance testing was purposely left out of the project scope on the assumption that Microsoft's Dynamics 365 cloud service should be performant on its own. (Although it's true that Microsoft is responsible for delivering a reliable and performant cloud service to its customers, our experience is that solution design and the resulting configurations, customizations, and ISVs to achieve that design may play a role in impacting overall Dynamics 365 solution performance.)

In Success by Design, findings link observations to known patterns that invite actions necessary to address project risks and issues.

Considering that the Dynamics 365 solution in question is projected to support 4,000 users at scale (including the multiple integrations and a key customization mandated by the business), the project team's findings are clear: keeping solution performance testing out of scope is a risk that negatively impacts the project. Among other findings and recommendations not covered in this example, the project team's architect (who led the Solution Blueprint Review) recommends that the project Steering Committee approve adding solution performance testing into the test cycle.

The architect's findings are summarized as follows:

- The solution requires custom development to meet its business requirements.
- Performance testing is not included in the test cycle.
- Following best practices, solution performance testing should be added to the test cycle (and if necessary, the solution performance workshop should be scheduled to further explore the risk and report any additional findings back to the project Steering Committee).

Track success measures

One more step remains in the Success by Design process: tracking success measures. After completing Success by Design training, you may decide

to create your own success measure tracking template. Otherwise, project teams can use Success by Design tooling (as served by FastTrack or your implementation partner) to track success measures. But what are success measures, and why are they important?

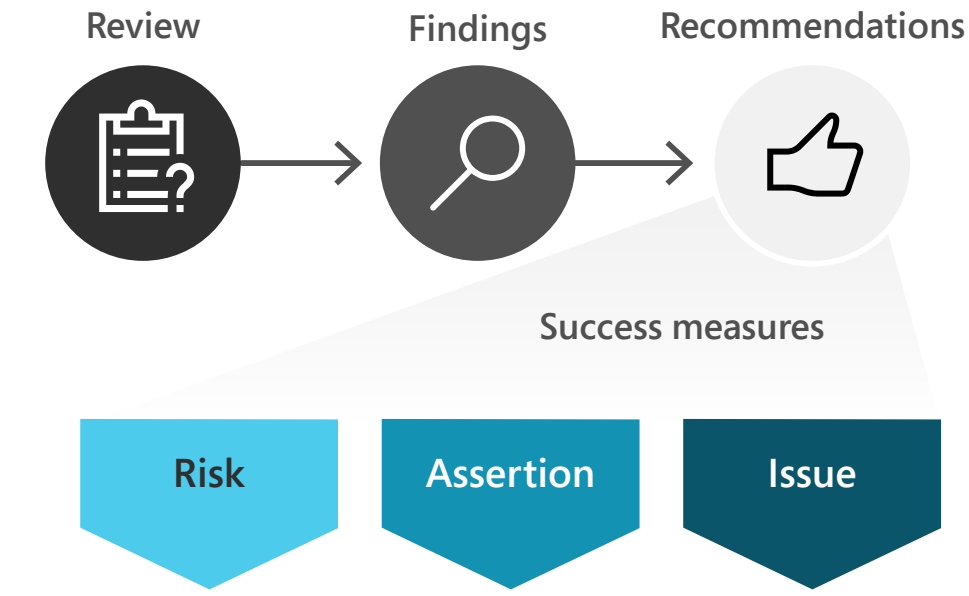
What are success measures?

By its very nature, Success by Design is a project-specific endeavor. For anyone practicing Success by Design across one or many Dynamics 365 projects, the question arose: How do we measure the health of just one of those many projects? Success measures allow us to do just that.

Success by Design empowers project teams to track the health of projects across seven categories and over 30 success measures. **Figure 2-4** highlights some of these categories and related success measures.

Tracking success measures is simple: After a Success by Design review or other compelling project event, your FastTrack Solution Architect or partner architect will access Microsoft's Success by Design tooling and update the relevant success measures for the project. Success measure updates are either red, yellow, or green, and include project-related details relevant to the success measure.

Fig. 2-4



Why are success measures important?

Success measures are important because they provide access to micro and macro project health trends. Tracking success measures for a single project allows stakeholders to assess the overall health of the project at a glance. Similarly, the benefit of tracking success measures over 10, 20, or 100 projects is that Microsoft, the partner, or the customer project

If you're using Success by Design within your Dynamics 365 implementation project, we want to hear from you! To share your Success by Design feedback and experiences, reach out to us at successbydesign@microsoft.com.

team can see patterns that may be impacting them. For example, macro-level tracking may yield an application lifecycle management (ALM) problem. Access to this data allows Microsoft or the partner to understand whether it's a project or product problem, and gain insights on how to fix it.

Conclusion

Success by Design equips project teams with a model for technical and project governance that invites questions and reflection, which leads to critical understanding of risks that might otherwise go unnoticed until too late in the project.

Considering the pace of cloud solutions and the investment that organizations make in Dynamics 365 software and implementation costs, even the most experienced customers and partners with the best methodologies will benefit by incorporating Success by Design into their projects.

As covered in Chapter 1, Microsoft believes that every business is in the business of creating great customer experiences. To achieve that, business applications must do more than just separately run your back office, marketing, supply chain, or even field operations. They must give you the agility to remove every barrier in your way. When this happens, business applications become more than just operational solutions. They become resilient solutions that enable the digital feedback loop and adapt to customer demands, delivering stronger, more engaging experiences around the products you make and services you provide.

Case study

An inside look at the evolution of Success by Design

The content of this case study is based on interviews conducted with Dynamics 365 FastTrack Solution Architects. The goal is to provide readers with an inside look at Success by Design. We hope you enjoy this insider view.

When our Solution Architects got their start with Dynamics 365, they found themselves hooked on the challenge of effectively aligning an organization's business processes and requirements with the (software) product in hopes of delivering solutions that resulted in actual value for users.

Because of this, FastTrack Solution Architects also know the challenge of working with people, understanding the business, shaping software to meet the needs of the business, and delivering a solution that is accepted by users. They know that projects don't always go well and often fail.

It's for exactly this reason that the FastTrack for Dynamics 365 team and Success by Design were created.

A shift to Dynamics 365 in the cloud

Dynamics 365 isn't software that is purchased and implemented as is. As such, implementing Dynamics 365 products rely on qualified partners to get the implementation right.

Complicating matters further, the promise of the cloud has been that you can focus on the application and forget about everything

underneath. But the reality is that cloud implementations require a comprehensive understanding of how design and build decisions may impact Dynamics 365 cloud performance, scalability, and more.

Accordingly, the shift to implementing Dynamics 365 in the cloud remains challenged by on-premises implementation habits in which the mindset is to overly customize—"because you can"—without impact.

The shift to implementing Dynamics 365 in the cloud requires not just technical capability but also a definitive understanding of how to design and build Dynamics 365 solutions within the constructs of the online service.

The evolution of Success by Design

FastTrack for Dynamics 365 was formed to help customers maximize their investment in Dynamics 365 by making sure that they (and their partners) have access to the right information, recommended practices, and a connection to the product group throughout the implementation lifecycle.

When the FastTrack for Dynamics 365 team was formed, these goals were initially achieved without the use of a framework. In other words, FastTrack Solution Architects used their experience and connection to the product team, but this approach contained too many variables to be consistently reliable.

Over time, Microsoft's experience revealed that Dynamics 365 customers were running into very similar problems. For example, FastTrack would identify that some aspect of the solution was designed and built in a manner that wasn't compliant with recommended Dynamics 365 implementation practices. And unfortunately, too often the Solution Architects would catch the issue too late in the implementation lifecycle.

As a result, Microsoft began to ask itself: How can we change the FastTrack approach so our Solution Architects can address problems before they manifest themselves as problems? FastTrack also

acknowledged that its approach couldn't be "one Solution Architect solving the problems of one customer project." Our Solution Architects needed to work in a way that would benefit all customers.

To identify and address problems early, our Solution Architects had to engage with project teams on multiple levels. We needed to understand project plans, how project teams were thinking about doing customization, data migration, and more. This need for early visibility across all dimensions of the project meant aligning to an approach that afforded our Solution Architects the opportunity to get involved in these conversations from the start, to identify the problems and risks early, and to give the right guidance to the project team before more problems arise.

This required a period of FastTrack experimenting with the right approach. We started a concept that we referred to as checkpoints, which involved using surveys, but we found that this approach kept us in a reactive mode. We kept revising our approach and eventually we got it right.

From this experimentation, Success by Design was born: a proactive practice that invites a purpose-built dialog that touches all levels and dimensions of the project (technical and nontechnical) so problems are identified before they impact the project.

Success by Design gets to the heart of how project teams are thinking and what they're doing so that any risks can be identified and mitigated.

With Success by Design, FastTrack joins the customer, our partners, and the product team in a manner that ensures alignment between all three.

As stated earlier in this chapter, Microsoft recommends that project leaders team up with their implementation partner to enable Success by Design within their project. In addition to the Success by Design resources available in this book, it's highly recommended that project teams ready themselves by enrolling in the Success by Design training on Microsoft Learn.

Success by Design for every Dynamics 365 customer project

At some point during the evolution toward Success by Design, Microsoft asked the question: How can we make sure every Dynamics 365 implementation is successful from the start? The key phrase is "*every* Dynamics 365 implementation."

As this chapter highlights, the FastTrack for Dynamics 365 team doesn't have a special lease on the practice of Success by Design. This book's goal is to democratize Success by Design and make it available to the entire community of Dynamics 365 implementers.

The more project teams invest in Success by Design, the more they will get out of it.

3

Guide

Implement cloud solutions

Introduction

The cloud—the constellation of connected servers, data warehouses, and associated software and services—is at the core of a digital transformation strategy.

Organizations of all sizes recognize that the cloud is the fastest way to modernize their operations to increase efficiency while better serving customers. For many organizations, the shift to the cloud simply means running their applications in a public cloud infrastructure instead of the traditional on-premises datacenter. For other organizations, however, a software as a service (SaaS) model that uses cloud applications as well as the underlying cloud platform is a quicker path to digital transformation.

- Adopt a cloud mindset
- Cloud implementation
- Customize and extend cloud applications
- Operate in the cloud
- Evergreen cloud
- Upgrade from on-premises to the cloud

Whatever model an organization chooses, we find that the key to success isn't what implementation looks like, but rather how leaders, architects, and entire companies approach the digital transformation journey. Success in the cloud isn't just about the technology or the features available—it's about the organizational mindset. For a successful digital transformation, your organization must prepare for changes that span the entire enterprise to include organizational structure, processes, people, culture, and leadership. It's as much a leadership and social exercise as it is a technical exercise.

Supporting research and studies

McKinsey The keys to a successful digital transformation

Smarter with Gartner Cloud Shift Impacts All IT Markets

In this chapter, we provide a perspective on what to expect when implementing your cloud solution using Microsoft Dynamics 365. We introduce principles relevant to any SaaS application. Concepts introduced here are elaborated on in subsequent chapters. We hope these principles help you set your course and guide you in your leap into the cloud.

We start by addressing how to develop a cloud mindset in an organization, which is foundational to your digital transformation. Then we delve into factors to consider and understand before you shift to a shared public cloud from an on-premises setup. These include how the shift will impact controls you have in place and how to think differently about scalability, performance, shared resources, and more. After that, we discuss the principles around customization and extending cloud apps. Finally, we explore the operating model, always up-to date evergreen cloud models, and the options to migrate to the Dynamics 365 cloud from on-premises.

Adopt a cloud mindset

Embracing change is a common theme, but infusing it throughout your organization is vital to your digital transformation journey. Therefore, we include it as a thread that runs throughout this chapter and book. Adopting the cloud mindset is also an intentional way to help organizations break the mold and think differently about how applications are designed, developed, secured, and managed in the cloud.

First, let's understand the "why" behind the pervasive adoption of SaaS business applications by organizations across the globe. Traditional approaches to business applications mostly involve the IT department building a custom application or using an off-the-shelf product augmented with customizations to meet the specific business requirements. The typical approach includes phases to gather requirements, build, go live, operate, and then follow the change management process to request changes or add new features. The IT department has full control of the application, change process, releases, updates, infrastructure, and everything needed to keep the solution running for years.

On the surface, this sounds ideal. It has been the standard operating

model for deploying and operating business applications. This way of working, however, is losing relevance in today's fast-paced world of shifting supply chains and evolving customer needs and expectations.

Application development and deployment was disrupted not necessarily because of the cloud or new technology—the undercurrent that made the cloud pervasive was the changing landscape of business. Unharnessed data from ubiquitous smart and IoT devices and other sources fueled startups that disrupted whole industries seemingly overnight. With that, the average lifespan of companies plummeted. This inflection made the survival of a business dependent on delivering the best product, the best customer experiences, and constant innovation to meet ever-growing expectations. This new business arena means businesses must change more often, evolve their processes more quickly, and become more data driven to understand customer needs, pain points, and preferences.

The changes we're talking about aren't triggered by the cloud or technology. These are thrust upon us by the changing business landscape. Adopting a cloud mindset, then, is about transforming your processes, people, culture, and leadership in such a way that your organization can embrace change quickly and successfully. We believe the fundamental organizational characteristics that will determine success in this environment come down to focusing on delivering business value through a secure cloud technology platform. This includes the ability to harness the data that is captured to generate insights and actions. This platform will also rely on automation to quickly react to changing needs.

In this section, we explore these organizational characteristics that help drive the cloud-first mindset to achieve digital transformation.

"Every company is a technology company, no matter what product or service it provides. Today, no company can make, deliver, or market its product efficiently without technology."

—Forbes, "Why Every Company is a Technology Company"

Focus on business value

Technology is a ubiquitous force that influences almost every business. Indeed, the IT department within a bank or a manufacturing company may be larger than entire tech companies. Understanding how technology affects customers and business is important to help any company survive and thrive. It's also important to not let technology

hinder growth or impede business. The role of technology is to deliver business value by driving efficiency.

However, we often see enterprises whose technological landscape has become so large and complex over the years that they fail to deliver the agility demanded by the business. The IT departments are consumed with dealing with the complexity of software compatibility, update cycles, end of life deadlines, aging infrastructure, and antiquated security policies with little time to focus on delivering business value.

An evergreen SaaS cloud approach to your business application, however, takes away much of the infrastructure and operational IT effort to keep the servers running, software patched, and other routine tasks, and turns the focus of IT on the business. Operating in the cloud doesn't mean there are no operational responsibilities, environment management, updates to SaaS services, feature deprecations, user communications, or coordinating change management for enhancements and new capabilities, monitoring, and support. These functions are all still required, but with a key difference: IT is now much closer to the business application layer and has a greater opportunity to focus their energy on delivering positive business outcomes.

IT has a greater opportunity to focus on delivering positive business outcomes.

Traditionally, business applications depict a set of key business processes that allows the users to capture, track, and report on the process and its progress. These business processes usually take the form of having a user interface (UI) built on top of an underlying data store to capture or display the information from the database. These "forms over data" software applications can help organizations keep track of information and make it easily available through reports. However, from an end user's perspective, they're a data capture tool that doesn't really help them do their job any faster or better. This design of business applications hasn't changed in decades. It's an outdated model that doesn't bring any additional value to the business or to the end user, and doesn't differentiate a business's offerings and customer experience from their competition, which is likely following a similar outdated approach.

Additionally, on an individual level, we want to automate routine, boring tasks and focus on more creative, challenging work.

For example, take a point-of-sale application that helps you address your customer by first name and provides them personalized relevant offers based on their loyalty, or recommends products based on their purchase history and Artificial Intelligence (AI). This will be better received by your employees and customers than an application that can barely determine if a product is in stock or not. Simply offering the best deal or best quality product isn't enough in today's hyper-competitive environment. To win, you need to differentiate between your customers, respond to their unique behaviors, and react to fluctuating market demands.

Automate the routine tasks so you can focus on creative, more challenging work.

Businesses likely already have the data to do this, but may lack the technology to turn data into practical insight. For example, Dynamics 365 Commerce taps your systems to gather customer intelligence from a myriad of sources (such as social media activity, online browsing habits, and in-store purchases) and presents it so that you can shape your service accordingly. You can then manage product recommendations, unique promotions, and tailored communications, and distribute them across all channels via the platform.

Dynamics 365 applications are focused on delivering value to the business and empowering end users, changing the business application industry paradigm by forcing every business to innovate faster.

The "If it ain't broke, don't fix it" mentality common in large organizations, where business applications can remain unchanged for up to 10 years, puts the enterprise at risk. Unless your business is a monopoly and can sustain without investing in technology, the hard reality is most businesses today won't survive without strategic plans for digital transformation.

You need to ask yourself some questions:

- Has the rest of your industry not transformed and embraced technology?
- Do your customers demand a better and faster process?
- Is your competition growing their market share by eating into yours?
- Does your competition have a better Net Promoter Score (NPS)?

If you answered yes to most or if you don't know the answers, your business might be failing to adapt and react to changes in customer behavior and expectations. Businesses that don't invest in technology will face challenging disruptive competitors, either global behemoths or innovative startups. You need digital transformation technology that gives you the command of three essentials: deep, cross-channel customer insight; synchronized operations from front end to back end; and scalability to drive rich experiences as needed.

Decision-making around such technology now carries existential consequences. Lack of strategic visionary thinking, long-term planning, and matching investments can prove to be catastrophic.

Decision-making around digital transformation now carries existential consequences.

The critical point is your business application shouldn't be designed and deployed with an expectation that it will remain unchanged for long periods of time. Instead, the core design principle for business applications should be ease of change. You should take advantage of the wonders of automation to deliver changes as often as daily and weekly, at the whim of business. This continuous development philosophy is core to a Dynamics 365 SaaS application with weekly maintenance updates and multiple waves of new features delivered every year. Realizing the value of these cloud applications, therefore, is about adopting the latest best-in-class capabilities, continuously delivered. With this new mindset, SaaS services aren't just used to meet business requirements—they're helping drive business value and adoption.

Increasingly, technology is becoming so powerful and democratized that non-developers and people with no formal training in software development are empowered to build applications, create workflows that automate their work, and use machine learning with ready-made AI models to analyze data. This concept of citizen application development has drastically transformed businesses, where end users are developing solutions to solve business problems and sharing these applications with colleagues. This organic approach to app development has short-circuited the application development lifecycle, helping address the perennial issue of software adoption. IT departments are now responsible for the governance of these applications, securing the access to data and environment, and focusing on building

the underlying services and connector interfaces that the business can consume in their applications.

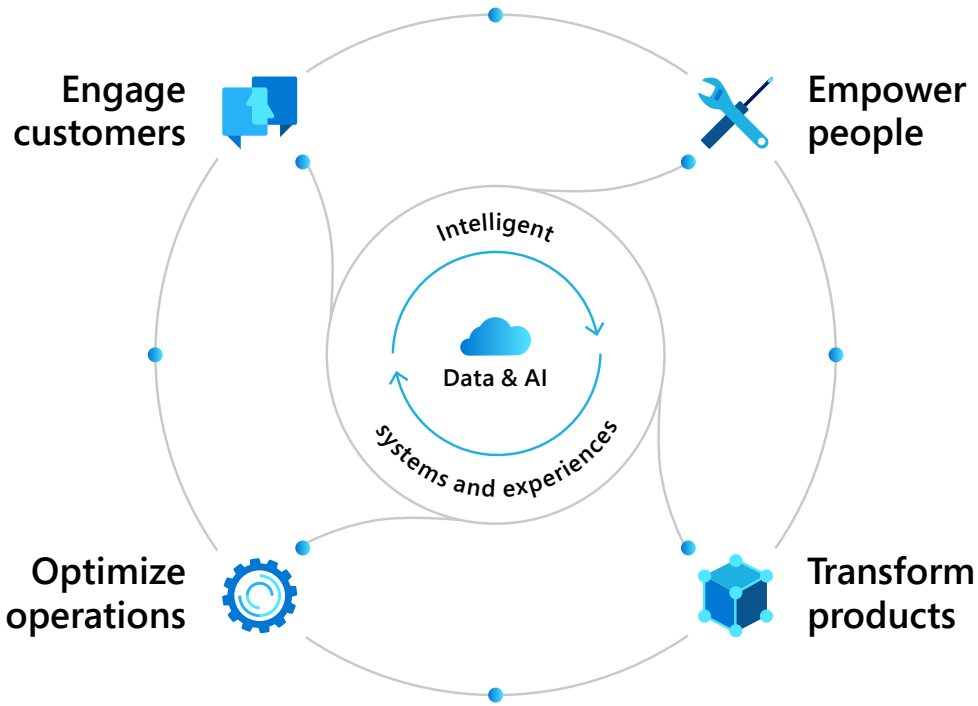
These scenarios all help illustrate how a transition to the cloud can play an instrumental role in refocusing technology to deliver business value.

Driven by data

This disruptive innovation we're seeing across several industries is fueled by data. Almost everything around us generates data: appliances, machines in factories, cars, apps, websites, social media (**Figure 3-1**). What differentiates one company from another is the ability to harness this data and successfully interpret the information to generate meaningful signals that can be used to improve products, processes, and customer experiences.

Organizations recognize data is an asset, and becoming data driven is a key motivator for digital transformation. Still, the accounting rigor that we see in finance is seldom observed when it comes to managing organizational data. If data is the new currency, then we should treat it like that. Get an understanding of your organizational data estate—the databases, data warehouses, and data lakes—that help your company manage its corporate data. Develop a holistic view of customer data across departments, functions, and applications, while acknowledging the regulatory boundaries on data usage. These steps are practically a prerequisite to digital transformation.

Fig. 3-1



Depending on the volume of data, its age, and organizational silos, the deduplication, correlation, and conflation of data from different systems could become a complex project on its own.

However, modern AI-powered customer data platform (CDP) technologies could come to the rescue. A CDP technology creates a persistent, unified customer database that is accessible to other systems. Data is pulled from multiple sources, cleaned, and combined to create a single customer profile.

It's also important to define your organizational data strategy for business applications, develop guidelines on how new applications store their data, what schema they use, and how other applications in the organization can understand and use this data. You need to use the most optimal datastore for your use case. For example, storing credit card transactions in Dataverse (a repository to store and manage data used by business applications) is suboptimal; a data lake might be a better choice.

Being data driven is about better understanding your data's quality and relevance, and being able to generate valuable insights that are actionable and can drive meaningful changes to your processes. Going back to the point of embracing change, your data strategy should reflect how you design your process, develop your applications, and operate. If it takes several months to deploy a change request or years to roll out an improved process, there is little point.

A data-driven business application cloud platform like Dynamics 365 enables you to:

- Create a data estate that is intelligent and AI-ready
- Connect and consume data from legacy on-premises systems, other line of business applications, and third-party applications
- Have the necessary controls to secure and meet data compliance requirements
- Build smart applications that are continuously improving
- Infuse data intelligence directly into the application, empowering users to make effective and smart decisions
- Have agility so applications can quickly adapt to changes in your process

Overall, the intelligence and insights you generate from your data will be proportional to the quality and structure of your data. You

can explore this topic more in Chapter 10, "Data management," and Chapter 13, "Business intelligence, reporting, and analytics."

Think platform

You're investing in the platform, not just the application.

An organization can take several approaches towards digital transformation. In many cases, you start with a single application being deployed to a SaaS cloud. A key responsibility of IT decision-makers and enterprise architects is to deliver a cloud platform for their organization's digital transformation. Individual applications and their app feature sets are important, but you should also look at the larger picture of what the cloud platform offers and what its future roadmap looks like. This will help you understand if the platform can meet the short-term and long-term objectives of your digital transformation.

Thinking about the platform versus a single app offers clear benefits. You avoid reinventing the wheel with each additional application, and you instead deliver a foundation approved by governing bodies, with clear patterns and practices. This approach limits risk and brings a built-in structure that enables reuse. Platform thinking doesn't necessarily mean that other cloud platforms or legacy applications have to be rebuilt and replaced. Instead, you can incorporate them as part of an all-encompassing platform for your organization, with well-defined patterns and swim lanes that enable integration and flow of data between applications. Bringing this "systems thinking" to deliver a platform for business applications can drastically reduce the amount of time lost getting security and design approvals for individual apps, thereby improving your agility and time to value.

Your business applications platform should already provide the following:

- Necessary security and design approvals for processing data in specific categories based on its sensitivity and regulatory requirements
- Clear guidelines on extending and customizing the platform to ensure supportability and agility
- Established governance process to manage access and service operation
- Approved integration patterns for communication with other platforms and on-premises systems, with a clear process for approving deviations

- Data storage guidelines for apps and patterns for sharing
- License entitlements that are available for individual apps
- Documented service protection limits and best practices that your applications should comply with
- Resources for additional learning and guidance with access to experts

The due diligence and foundational work you do when choosing a platform in coordination with IT and business decision-makers can save weeks, if not months, of effort downstream when developing individual apps. This thorough evaluation also helps drive predictable success.

Embrace DevOps and automation

Due diligence now helps drive predictable success in the future.

Traditionally, the process to manually release a change to applications was convoluted, involving several teams, change boards, manual regression tests, and approvals. It took weeks. This complexity makes teams averse to change. Enhancements and updates are then consistently deferred, which affects the adaptability of your applications to shifting business and technology priorities. Automating builds and releases and building in testing automation is crucial to allow for continuous software updates.

The biggest selling point of the cloud is to stay current and quickly deploy software. You can't rely on teams to manually test and deploy each and every change to achieve that vision.

Traditionally, a time lag existed between code and test and deploy. A bug in production meant repeating the lengthy cycle. The fear of breaking code meant that teams tended to delay updates for as long as possible. With automation, you deploy fast and potentially fail fast. This is where the culture of fail fast comes in. Automated processes help companies empower their teams to take risks. You fail but quickly release a fix. Over time, failures decrease and success rates improve, instilling confidence in teams to deploy and innovate, and delivering value to the business faster. At the core of the cloud mindset is understanding and embracing DevOps. DevOps is the union of people, process, and technology to continually provide value to customers. DevOps is a cultural change in which we break down silos between developers and administrators, and create shared responsibility for the

quality of released software. The DevOps process automates software delivery and speeds up the technology deployment lifecycle (**Figure 3-2**).

The idea is to enable development, IT, and quality and security teams to collaborate to produce more reliable and quality products with the ability to innovate and respond more quickly to the changing needs of business. Implementing DevOps inherently depends on automation: automating the build process to enable continuous deployment, regression tests to improve speed and reliability, and administrative actions like user and license management.

At this point, we should also explore continuous integration (CI) and continuous delivery (CD) in the context of the cloud. The idea is each code check-in from the developer should go into the main build and get tested early. This takes CI to the next level by allowing for automated testing and continuous, automated code deployment.

If a change includes a bug, the build fails, and the cycle repeats. The automation allows for changes to happen quickly without unnecessary overhead—a team can focus on the business logic and not the infrastructure. Therefore, with CD we always have the most up-to-date working software.

Fig. 3-2

CI and CD bring agility to cloud solutions by allowing for the latest changes and bug fixes to move to production code without delay. This is a paradigm shift from the world of on-premises software, in which each change had to wait until all user stories were completed, the full package was deployed, and a testing team tested the change. With automation, we're reducing time and improving quality.

But this isn't just a technological change. DevOps requires close collaboration between development, testing, and deployment teams. This is a cultural and mindset change in which the whole team trusts each other, feels empowered, and is collectively responsible for product quality.

Preparing your organization for the fast-paced world of the cloud means that automation isn't optional anymore. Instead, investments in automation will deliver significant efficiencies and help improve the overall reliability of your applications.

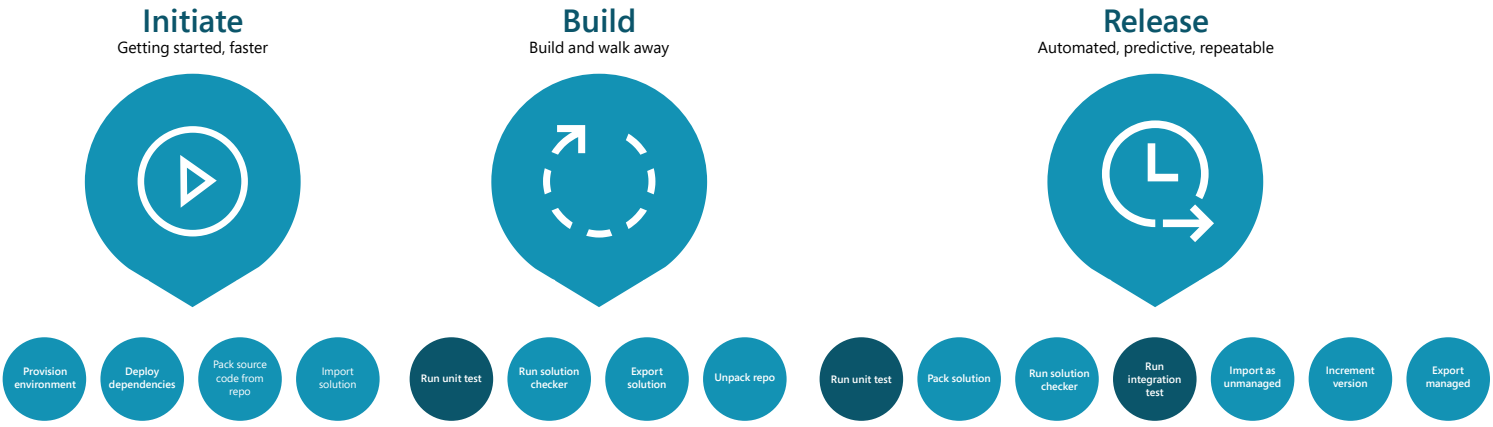
Dynamics 365 offers deep integration with Azure DevOps with build tools to help automate your deployment pipeline and effectively manage the application lifecycle (**Figure 3-3**).

Cloud implementation considerations

Implementing solutions from cloud SaaS products can reduce management overhead and allow you to deliver business value more quickly by focusing on the application layer. Fundamental application considerations, however, still apply to SaaS cloud applications. Security, scalability, performance, data isolation, limits, and capacity are still critical, but could have a different approach when compared to an application deployed on-premises. This section focuses on some of these considerations for SaaS cloud apps.

Fig. 3-3

ALM Powered by Azure DevOps



Initial build pipeline instantiates pristine development environment daily.

Building pipeline automates manual steps. No more need to upload to solution checker and manually export solution, unpack, and push to repo.

Automated release pipeline removes manual steps. Weekly, daily, or hourly releases become the new standard.

Security readiness

The responsibility for security in SaaS cloud applications is shared by both the service provider and the customer. That means your existing security policies might not be suitable to meet the security requirement in the cloud. The SaaS service provider processes customer data and is responsible for aspects such as securing the backend infrastructure and data encryption in transit and at rest. As a data controller, the customer is still responsible for securing access to environments and application data.

The IT information security team should clearly understand the boundaries of these shared responsibilities to ensure the following:

- The SaaS service provider meets the organizational security, privacy, and compliance requirements. This is usually done in the beginning to approve the platform for use supported by a regular review or audit process to ensure continued compliance.
- The security team is aware of the controls and configurations required to govern access to data and fulfill the customer security responsibility. These could include defining the data loss prevention policies, access control policies, and environment management policy. They usually have a default setup with a process to manage deviations for specific apps.
- The governance process makes sure the application-level security requirements are met. This is primarily managed by the application teams and driven by business requirements for each app deployment. You might have additional checks before deployment to ensure compliance.

A good example of this is General Data Protection Regulation (GDPR) in the EU, in which the SaaS service itself might be fully GDPR compliant from a data processor standpoint but the customer is still responsible for implementing data controller processes like a “forget me” request or managing marketing consent for contacts.

Organizations need to review their information security policies and ensure that they’re cloud ready. Doing this exercise earlier, before the app deployment cycle, will help avoid delays. It’s important to closely work with your cloud SaaS service provider to ensure all your security

requirements are met. Microsoft publishes the details of how this shared responsibility of securing data is fulfilled from a data processor perspective on the Microsoft Trust Center, which provides detailed information on our security and privacy policies as well as the certificate of compliance for various regulatory norms and internal standards.

Scalability

The scalability of the SaaS platform is a key consideration for business applications. Being able to scale out and scale up to support seasonal workloads or spikes in user activity will impact the overall user experience (both staff and customers) and effectiveness of business processes.

The cloud scalability parameters available in SaaS differ from a traditional on-premises system. Instead of adding more servers or increasing the power of machines, these parameters could be translated as available API capacity. You also shouldn't assume that cloud means infinite scale and computing power that can process everything thrown at it. The good old coding and design best practices are still relevant, but might need to be adapted. Although managing and scaling cloud services is complex, increasing or decreasing your data storage, processing power, and networking requirements can be done seamlessly. In many cases, you can do so automatically or with simple configuration changes. This microservices architecture, including capacity-based routing to resources and storage management, is transparent to Microsoft customers.

To summarize, the SaaS cloud offers the flexibility to scale horizontally and vertically to support thousands of concurrent users. Embracing the capacity-based model for resource consumption in the cloud helps not only build optimized applications but also a better plan for operating cost post deployment.

Latency and performance

Performance is a key consideration for business applications—it not only impacts the end user experience and adoption, but can directly impact business goals and key performance indicators (KPIs) of success.

In on-premises deployments, enterprises had complete ownership and control of the infrastructure, and could guarantee and monitor applications' latency and performance. In the cloud world, it's not as straightforward.

Research shows that a few milliseconds of latency lead to a big percentage drop in page load times. For e-commerce companies, this could mean a sizable drop in user attention and therefore sales revenue. Low latency is not a good to have, but a critical deciding factor in an enterprise's brand and reputation. The same applies to back office business applications operating in the cloud—user experience and productivity can be significantly impacted in a high-latency environment.

Several contributing factors can impact application performance, including network latency, user device and browser, application design, and customizations. SaaS services are inherently scalable, have vast compute power, and are available from multiple locations around the world, but that doesn't necessarily guarantee performance if the solution is poorly designed, or if users are accessing the service from environments with high network latency.

Your cloud applications are powered by hundreds of pieces of virtualized hardware running on Azure datacenters sprinkled around the world. A crucial decision you must make, then, is choosing the datacenter to deploy your application so that all users of the application get an acceptable level of latency.

Performance also can suffer because of a poorly designed application. If you're changing standard code in SaaS or deploying custom applications in platform as a service (PaaS) or infrastructure as a service (IaaS), your team must design a high-performing application and carry out proper performance tests, and only then do you deploy to production. Now, most cloud providers work on a shared network infrastructure in which your requests go through the internet and are used and shared by everyone.

Network latency is a crucial factor to consider alongside other architectural and design decisions that impact performance.

You can test latency using the Azure Latency Test and Azure Speed Test 2.0 for each datacenter.

One option for businesses looking for more reliability and security is to use a private connection. Cloud providers offer dedicated channels, for example Azure ExpressRoute. These connections can offer more reliability, consistent latencies, and higher security than typical connections over the internet.

Isolation in the shared cloud

Private connections offer reliability, consistent latencies, and higher security

The cloud operates on a shared common infrastructure across different businesses, which leads to economies of scale. But although the infrastructure and processing may be shared in public clouds, you have data isolation, which means one customer's data isn't shared with others. Security and data use policies are taken very seriously by Microsoft; it's imperative that we maintain the highest data and security standards. These standards and practices are audited and certified on a regular basis by trusted third-party organizations. Security isn't an afterthought, but ingrained into how we develop applications, so that the computing infrastructure is shared and other security related aspects are isolated.

The other aspect of sharing resources and infrastructure in the cloud is the possibility of monopolizing resources or becoming a noisy neighbor. While rare, these cases are more often caused by poorly developed runaway code than a legitimate business use case. Automated monitoring and protection throttles are built into the Dynamics 365 platform to prevent such situations, so it's important to understand and comply with these service boundaries and throttles when designing for cloud platforms.

Service protection and a capacity-based model

Service protection and limits are used in cloud services to ensure consistent availability and performance for users. The thresholds don't impact the normal user operations; they're designed to protect from random and unexpected surges in request volumes that threaten the end user experience, and the availability and performance characteristics of the platform or solution. It's crucial to understand the protection limits and design for them with the appropriate patterns, especially around high-volume workloads like integrations and batch processing.

Plan for the daily peak and the monthly maximum order transaction volumes expected to ensure that the service is licensed to satisfactorily support the peak as well as the total maximum expected volumes. Also, plan and prioritize for integrations and Open Data Protocol (ODATA) requests based on the volumes, so they're honored and not throttled due to request limits. These checks and balances prevent overutilizing resources, preserve the system's responsiveness, and ensure consistent availability and performance for environments running Dynamics 365 apps.

The cloud provides us the scalability to deal with large workloads, but a shared public cloud doesn't mean you have unlimited capacity or computing power. In many cases, these capacity limits are enforced via licensing. You have to embrace this capacity-based model of cloud infrastructure and plan to operate within the entitlements, taking into account usage profile, data volumes, and integration patterns. Understanding these protections and service boundaries that apply to a specific service helps bring clarity on available resources. It also drives the right behavior when it comes to design and development so the solution is optimized to operate within the allocated capacity, or additional capacity is procured to meet the requirements in advance.

Check out Chapter 20, "Service the solution," for more information.

Integration with on-premises systems

The SaaS cloud might be the preferred approach for new applications, but in most enterprise solutions, you could still have on-premises elements. Therefore, it's common to build integrations with these on-premises systems. Establishing these integration patterns in compliance with your internal security policies while allowing cloud applications to authenticate and connect to on-premises services is critical. This could involve allowlisting URLs and IP ranges on your firewall, or using technologies like the Azure AD Application Proxy or an on-premises gateway that allows such integrations without having to open inbound connections into your customer's corporate network.

Even from a design standpoint, patterns that worked on-premises could be challenging in the cloud, due to latency and additional security restrictions imposed. An example is the encryption of traffic leaving a customer network or routing via a peering network.

Cloud and edge

The public cloud is the default and recommended choice of deployment for the majority of Dynamics 365 customers, but you may want to extend the cloud to unlock important capabilities for key scenarios where connectivity is an issue. Your industry may need a solution on the ground, such as brick and mortar commerce, field service, a manufacturing execution system (MES), warehouse operations, and project operations.

These implementations are called cloud and edge, where cloud is the system of intelligence and edge is the system of record.

For example, Dynamics 365 Commerce customers use the cloud and edge model, in which the store operations run on-premises (edge) and the main instance handles centralized back office functions like finance, data management (such as products and customers), and analytics (cloud).

Patterns that worked on-premises could be challenging in the cloud.

Cloud and edge deployments provide business continuity for mission-critical functions like point of sales, warehousing, and manufacturing. We want you to be able to continue running operations when disconnected from the cloud. In case of a network outage, Dynamics 365 Point of Sales can sustain itself with basic operations and keep data generated locally. When connectivity resumes, data is synchronized to the cloud. Also, whenever throughput is high and more heavy processes are run in parallel, user productivity may be impacted. We want to enable you to scale out and run manufacturing and warehouse processes in isolation so high user productivity is always supported.

Another reason for choosing an on-premises deployment model is that these businesses have made significant investments in local IT infrastructure. We want to ensure you receive a meaningful return on investment (ROI) before taking the full cloud route.

Another reason is internet connectivity. Regions with poor internet connectivity see more occurrences of on-premises deployments. Organizations operating in such regions go with an on-premises model to mitigate the risk to their business from poor connectivity. They also want to maintain a stable user experience for their staff.

Customize and extend cloud applications

A SaaS application with rich features also provides powerful capabilities to customize and extend the app to meet specific business

requirements. Customizations to the SaaS application can be at several levels. Many involve little code (embracing the low-code/no-code philosophy) and offer the flexibility to tap into the underlying PaaS layer. This allows you to take advantage of professional software development techniques that include custom code.

Chapter 15, "Extend your solution" covers the details of customization and app extensions, considerations, and scenarios. In this chapter, we focus on the principles to keep in mind when making decisions about customizing the application or extending platform capabilities. While these principles broadly apply, in some cases a tradeoff is necessary, in which the approach depends on specific situations, in-house skills, timelines, budget, user expectations, processes, and other factors.

Know the platform

When using a SaaS cloud platform like Dynamics 365, it's important to know that this isn't a one-time software purchase. It's an ongoing partnership between the customer and service provider, with an operating contract that governs the service-level agreement (SLA), security, ongoing updates, and more. You need to understand how the service runs and how updates and changes impact your solution (and therefore business), and employ the supported way of extending and customizing the application in harmony with the service contract. With this knowledge and compliance, you can ensure continued support and avoid unexpected issues.

Even when using supported extension techniques, you need to adhere to best practices. A good example is that while the platform might allow you to run synchronous code for up to two minutes when creating a record, running to the time limit would block a user's UI thread for two minutes when they save a record. Is that acceptable? Similarly, you could use custom code to limit access to records and implement a custom security requirement, but certain access mechanisms could bypass your custom code, leading to exposure. The point is to carefully design your customizations and assess their impact, particularly when those customizations affect the end user's experience or deviate from the security control constructs provided by the platform.

In general, the low-code/no-code techniques that rely on configuration rather than custom code are optimized by the platform, and should be a preferred approach. However, custom coding may be required in some use cases. We have a no-cliffs philosophy with limitless possibilities to extend the application using the underlying platform. The ability to address this by using the underlying PaaS is a superpower in the world of SaaS applications.

Be aware that custom coding and PaaS extensions bring additional maintenance responsibilities and are best left to professional development. Even when using PaaS to extend your solution, serverless technologies are usually better suited to this approach.

Other factors include solution capacity planning, which should consider the impact of your customizations, and the implications of using certain PaaS technologies that use a pay-as-you-go model. We recommend your organization establish guidelines for customizations that take into consideration the platform, user experience, security, and approved patterns for integrations. In addition, enforce best practices with automated checks wherever possible in the build pipeline. Power Platform includes an app checker and app monitor that can help identify unsupported techniques and anti-patterns that can lead to performance and scalability issues.

Using a SaaS cloud platform is an ongoing partnership between customer and service provider.

Don't mimic your existing system

A common misstep is trying to mimic a legacy system or the features of a different SaaS platform when implementing a solution. This mindset can lead to excessive customization and prevent you from realizing the true potential of the platform and all its strengths to build your solution. To focus on business value, you should adopt a business process–oriented approach. It's important that business requirements aren't muddled with implementation details or dictate a specific approach rather than a business outcome. A simpler approach is to look at the business processes for transformation and implement those using the platform capabilities instead of replicating the existing solution. It's closer to business, future-proof, less costly, and easier to maintain than simply lifting and shifting your existing experiences to the cloud. You also avoid technical debt from excessive

customizations. Aligning to the platform allows you to maximize its capabilities so you get an evergreen, always up-to-date solution.

Future-proofing

One of the key principles we have established in this chapter and throughout this book is getting comfortable with change. SaaS enables you to adopt new features to maintain a competitive edge. These features and capabilities are built on top of an existing baseline of features and tables. Although repurposing or using a custom table may not be unsupported, deviations can severely impact your ability to take advantage of future capabilities or can affect the interoperability of your solution.

To focus on business value, adopt a business process–oriented approach.

In addition to the business process–oriented approach to implementation, we also recommend that you review your existing business processes and compare that with what the standard application has to offer. You can benefit from adopting standardized processes backed by research and feedback from leading customers in their industries, because you benefit from all that expertise. Using standard processes also makes your business less dependent on expensive and hard-to-maintain customizations.

The Common Data Model (CDM) is an open standard schema for business data created in collaboration with SAP and Adobe, and is used across Dynamics 365 applications. Aligning to the CDM can help future-proof your solution. It can also make sure you can take advantage of enhancements as well as other business applications across the industry that honor the schema and can easily interpret your data.

Another way to future-proof is to understand your roadmap, investment direction, and trends so you can better align to the future direction of the platform. This helps reduce technical debt and rework.

Understand the tradeoff

Expect tradeoff decisions to be made when implementing your solution, particularly when it comes to the user experience (UX). In Dynamics 365, the platform provides several out-of-the-box controls

and UX patterns that can be reused in a model-driven paradigm to quickly assemble applications. These address common usage patterns and also meet accessibility standards. It's common for a business to make specific requests around application navigation, behavior of a specific control, or even colors to highlight specific controls. From a technical perspective, most of these UX needs can be met either by using custom code to build custom controls or by using a different app paradigm, in which you start with a blank canvas to develop a pixel-perfect UI that you could use in combination with model-driven controls.

Several factors could guide these decisions, including user adoption, but you should also assess the cost-benefit ratio of customization. In some cases, building and maintaining custom controls can be time-consuming and require specialized skills. If you can easily address some scenarios with user training, you can save hundreds of hours of development and test effort. In addition, specialized UX requests are often subconsciously driven by the need to replicate an existing system or just a familiar pattern, even though the requests defy modern UX best practices and accessibility needs.

Independent software vendors

Independent software vendors (ISVs) are third-party organizations that exclusively develop software or software solutions that are compatible with the service to deliver additional capabilities.

An effective way to approach extending your solution is to use an ISV solution from the app marketplace for your platform. This can save time and effort in development and testing, and provide you with a tried and tested solution used by peers in your industry.

Several industry-focused ISVs provide solutions to fill unique product gaps and address specific business needs in industries such as fashion and hospitality. Make sure to perform due diligence when looking for an ISV. For example, an ISV needs to make sure their solution is available for the version you're targeting for go live. Otherwise, deployment timelines can be affected. It's also important that you're aware of the support model for an ISV, their commitment to keep it up to date, and the protection clause if the solution provider goes bankrupt or is sold.

AppSource offers a variety of ISV solutions to suit your needs.

You will also want to know how the ISV handles deprecation notices for phased out software and update cycles.

Chapter 15, "Extend your solution," provides a deeper dive into making a solution work for you.

Operate in the cloud

Successfully operating in the cloud will drive business value for your organization. Having the right skills and the change management processes aligned to your platform will ensure you are set up to operate smoothly and derive continued value from service updates and enhancements.

Center of Excellence

You might have a partner and specialized development team assist with the initial deployment, but once in "business as usual" operations mode, does your team have the skillset to manage the platform, maintain the custom components, and assess and implement appropriate new features? Not having the correct level of expertise and guidance onboard after you go live can inhibit solution advancements and undermine your ability to take full advantage of the platform. Creating a Center of Excellence (CoE) within the organization and investing in developing the skills to operate on the platform is extremely valuable. Organizations that embrace organic application development by end users also need a well-established CoE. This center can administer, nurture, govern, and guide users to adopt the citizen development approach to business applications, empowering everyone to create innovative apps with high standards by using a consistent application lifecycle management (ALM) process.

Microsoft provides a CoE starter toolkit that includes prebuilt processes:

- **Administer** Gain insights into your Power Platform adoption
- **Govern** Establish audit, compliance, and archive processes
- **Nurture** Accelerate your adoption with a thriving community
- **Drive innovation** Pick the most impactful and valuable scenarios for development
- **Consistent ALM** Apply source control strategies using GitHub

You can read more about CoE at Establishing a Microsoft Power Platform Center of Excellence.

- **Standards and theming** Create, manage, and share enterprise-branded themes as well as maintain and follow organizational best practices and standards

Usage and monitoring

Running your business on the cloud means you want the cloud to be always available, responsive, and scalable. You also want to be alerted to unexpected issues in the applications and infrastructure so you can diagnose and fix them. Additionally, you want to comply with SLAs you have in place from other businesses. Having a monitoring capability can prepare your business to meet these objectives.

What to monitor?

Different personas in your company will want to monitor for different aspects of the system. End users may be more concerned with responsiveness, admins may be looking at changes, and the business may be looking at KPIs like the time taken to place an order.

When running a SaaS application, you want to focus on the business, while a service vendor assumes the role of maintaining the solution. As a customer, you still have a level of control over the solution. This is especially important when you have customizations in place to fine-tune the business processes for your organization. If you introduce a new feature in production and it causes unexpected issues, you want to detect it proactively and resolve it before it becomes a problem. Using features and flags to control new features can serve you well in these situations because you can quickly turn off a faulty feature.

For example, you can configure Dynamics 365 and Azure Application Insights to access telemetry and diagnostics data and set up alerts and dashboards to proactively monitor your Dynamics 365 environments.

A proactive and reactive monitoring strategy that empowers your staff to detect and resolve issues should be part of every customer's arsenal.

Best practices for monitoring

Keep in mind the following best practices:

- Decide what exactly to monitor, especially the KPIs, and whether the service is meeting them.
- Monitor your consumption, subscription, and services fees. The more cloud services you use, the more they will cost.
- Use automation to automatically trigger actions and to alert

people when thresholds are reached.

- Track user activities for accountability (including the applications, screens, and features they use), the responsiveness, and the frequency of use.

Adopt and align to a product roadmap

Your cloud solution will continuously evolve and deliver on the promise of digital transformation by delivering new capabilities that help you stay competitive in your industry. Cloud vendors do their own research and listen to customer feedback to improve the product by adding new features. Being actively engaged in this process by giving feedback, staying abreast of the latest roadmap, and planning for upcoming changes is an effective way to operate in cloud. Also, the key stakeholders from the business need to remain engaged and stay plugged in to new features development, investments, and product direction. We recommend developing your own roadmap for business applications and aligning this to the product direction to get the most out of your investments.

Dynamics 365 has two release waves per year in which several incremental enhancements and new capabilities are made available to customers. Adopting these mandatory changes and new features—many of which are included in existing license plans—are a fundamental aspect of operating in cloud.

Stay engaged

Implementing the system is one thing, but getting continuous return on investment (ROI) from your cloud solution requires engagement. Companies who stay passionate and curious about the solution and adopt the latest features are the ones who enjoy most success. We urge all customers to maintain engagement through conferences and events throughout the year, and stay connected by using community groups. Several official and unofficial communities have been formed by customers, partners, and Microsoft engineering. Take every opportunity to influence the product through active engagement with product leaders, advisory boards, table talk with product managers, preview programs, and collaborative development opportunities.

You can choose from application-specific forums like the Microsoft Dynamics Community Help Wiki, Yammer groups, blogs, events, and how-to videos where you can discuss ideas, learn features, learn roadmaps, and ask questions.

Evergreen cloud

One of the benefits of a modern SaaS solution is that every customer runs on the latest version of the service. It's the only model that scales. With this approach, every security patch, bug fix, performance enhancement, and functional improvement accrues to all implementations across the globe. Microsoft assumes responsibility for keeping the platform current. This means that you no longer have to pull large teams of people together every few years to perform a traditional upgrade over many weeks or months with limited added value for the business. The evergreen approach and the model of continuous updates gives your business access to the latest capabilities to stay head of the competition and meet changing customer expectations.

Dynamics 365 has embraced this approach of operating on one version in a way that delivers benefits without disruption. The improvements and enhancements embrace the CI/CD model, with service updates delivered on a weekly basis, but these changes aren't targeted at the end user experience. You can take advantage of release waves to preview upcoming changes and test and prepare your users before the changes are deployed.

Achieving the right balance of delivering improvements and business capabilities in a safe manner without disrupting the user experience is paramount to our evergreen cloud approach.

Chapter 20, "Service the solution," delves further into the updates wave approach.

Upgrade from on-premises to the cloud

Organizations over the years might have invested in on-premises deployment of business applications. These applications could serve the users and business well in their current state, but keeping them

on-premises over time could limit your opportunity for expansion. In these cases, you should explore upgrading or migrating an existing on-premises business application to the cloud and bringing along existing business processes and data to speed up your digital transformation.

Dynamics 365 provides the tooling and support to help on-premises deployment upgrades or migrations to the cloud (subject to compatibility and security checks). This option to upgrade to the cloud also helps organizations take advantage of their existing investments in implementation, as long as the supported implementation techniques are used and compatible with the cloud. Still, everything we discuss in this chapter on the cloud mindset, considerations, extension, and operations is fully applicable.

When upgrading or migrating to the cloud, consider the following:

- The most critical factor in deciding on the upgrade-to-cloud approach is to assess if the current application being upgraded meets the requirement of the business and is well adopted by users.
- The data model, design, and data quality should ensure that the application being upgraded becomes a stepping stone to accelerate digital transformation. The upgraded application should serve as a foundation for further growth and adoption of new capabilities in the cloud. It shouldn't stifle cloud adoption by carrying over poor design and data.
- The cloud platform should be approved by security to store and process data in accordance with the organization's security policies and regulations. The patterns used for authentication and integration also need to be revisited to make sure they're cloud ready.
- Understanding the data category, flow, transformations, location, and encryption during and post migration will help you navigate organizational security policies and plan your cutover.
- The upgrade to the cloud can be a multi-staged process that involves stringent compatibility and validation checks, so it's critical to understand the process and review the prerequisites early in the process.
- Some components in the on-premises application may need to be remediated or replaced to ensure cloud readiness. Examples to look at include deprecation, unsupported customizations, and non–cloud ready patterns that pose a security or performance risk.

The Dynamics 365 Migration Program can help you take the first step toward cloud success by migrating your on-premises solution with expert guidance from Microsoft. For more information, refer to the overview of our program or perform a standard migration assessment.

You can also move from Dynamics AX to Dynamics 365 in the cloud.

- Existing integration and peripheral applications will require changes to ensure they work with the cloud applications.
- Any impact on process, performance, and usability due to latency, service protection limits, or license capacity should be considered.

The ability to bring existing Dynamics 365 on-premises applications into the cloud with proper due diligence can be a game-changer for your digital transformation.

Conclusion

Embracing SaaS applications to run your business can significantly accelerate your digital transformation, but it's also important to recognize that organizational cloud maturity will play a significant role in your strategy's long-term success.

Organizations working on their first major implementation in the SaaS cloud should expect some level of disruption to the existing ways of deploying and managing applications. We hope this chapter serves as a primer for Success by Design thinking by helping set the context for the changes you can expect and driving you towards greater cloud maturity.

Checklist

Adopt a cloud mindset

- [ ] Have a shared understanding at every level, from the executive sponsor to the developers, of the business impact being delivered.
- [ ] Ensure the organization has done its due diligence organizing the data estate and understands the impact of the new solution on the data estate.
- [ ] Gain approval for the cloud platform for use with the appropriate application data category from information security and compliance.
- [ ] Ensure the respective teams understand all administrative, operational, support, and monitoring aspects of the platform, and the organization policies, processes, and patterns conform to the SaaS cloud.
- [ ] Implement DevOps and CI/CD pipelines to support automation for build, testing, and deployment.
- [ ] Design the solution according to service boundaries and available licensing capacity, and ensure the controls to further expand and scale the solution are understood.
- [ ] Design the solution to meet the nonfunctional requirements, considering factors like network latency, end-user environment, and devices.
- [ ] Ensure that when upgrading existing on-premises solutions to cloud, the data model, design, and data quality make certain that the application becomes a stepping stone and doesn't stifle cloud adoption by carrying over poor design and data.

Customize and extend

- [ ] Create guidelines for when to customize and extend out-of-the-box apps and only adopt documented extension techniques.
- [ ] Avoid deviations and repurposing the out-of-the-box tables and data models (Common Data Model) because this inhibits future adoption of new features and capabilities.

Operation

- [ ] Ensure that the necessary expertise is available via the partner (system integrator) or internal teams post-deployment to support and evolve the solution.
- [ ] Engage with the community to keep up with the latest innovation and help influence the product direction using channels, events, blogs, and other mediums.

Case study

Rev up for speedier service

A family-owned business based in South Australia is one of the largest retailers in the region, with ventures that span fuel, convenience, quick-serve restaurants, and real estate.

One of its key businesses is a chain of retail stores that serves busy guests across Australia.

Their focus on customer service has led the company to reimagine the way its store systems are designed and connected. As their retail chain has grown, they have faced common challenges: outdated technologies, changes in customer behavior, and the pace of change with existing suppliers. The company was ready for an overhaul and an infrastructure solution that could serve its goals now and in the future.

Given the high volume of traffic that the retail chain processes and the diversity of its retail store locations, the company was seeking a hybrid solution with reliable, flexible connectivity.

Scalability and speed are critical for their retail business, and they needed an infrastructure design that optimizes for both.

“We serve millions of guests each month,” says the company’s store systems specialist. “To achieve speed of service, we know that we need to have something on-premises to retain our workloads in the store. And, at the same time, we need to have that cloud connectivity to the back office.”

A cloud engineer at the company adds, “We have stores in rural areas where they don’t have high internet speed connections. We wanted to have something that is both in-store and in the cloud that would synchronize when it comes to failovers and redundancy, so that we can have the best of both worlds.”

When it came to renewing store hardware and to keep up with the pace of innovation, the company chose to go with a hybrid model that integrates Azure, Azure Stack Edge, and Dynamics 365 Commerce.

The new design impacts the stores’ day-to-day connectivity. “What we are planning to do is a direct connection to Dynamics 365, and the failover needs to happen in Azure Stack Edge,” one engineer said. “We expect to have a minimum of 99 percent of uptime.”

One of the team’s other top considerations was how easy the infrastructure would be to manage, so they factored in what the company was already using across its retail business.

“We have already heavily invested in Microsoft products. We use Dynamics 365, Microsoft 365, Power BI, and multiple other products,” explains one engineer. “We wanted to have a single pane of glass where we could administer and monitor all of the systems.”

The hybrid deployment model using Azure Stack Edge and Dynamics 365 Modern Point of Sale provides store operations redundancy in case of network outage.

The new infrastructure design provides the connectivity, consistency, and efficiency that the company needs. Over time, the company expects that it will help them avoid store downtime, maintain real-time stock levels, simplify its infrastructure management processes, and reduce maintenance and compute costs.

The new infrastructure design will impact the customer experience at the store level. “We offer a comprehensive omnichannel experience to guests walking into our stores,” says the store systems specialist. “We enable our guests to pay at the pump without having to come into the

store by using the [retail chain] app. We also enable our guests to click and collect their food, coffee, or supermarket needs. To enable that, we need real-time connectivity to our head office systems—that is key for us. With Azure Stack Edge and an in-store database, we are hoping to get that real-time connectivity."

This orchestration ensures connectivity and consistency, as a cloud engineer explains. "We deployed three virtual machines. One is used to monitor the in-store cameras, and the other is used to store a Dynamics 365 database backup. If the store loses its internet connection to Dynamics 365, it would be able to operate normally while that virtual machine continues with the operations. Then, once the connection is restored, it will be able to synchronize back to Dynamics 365."

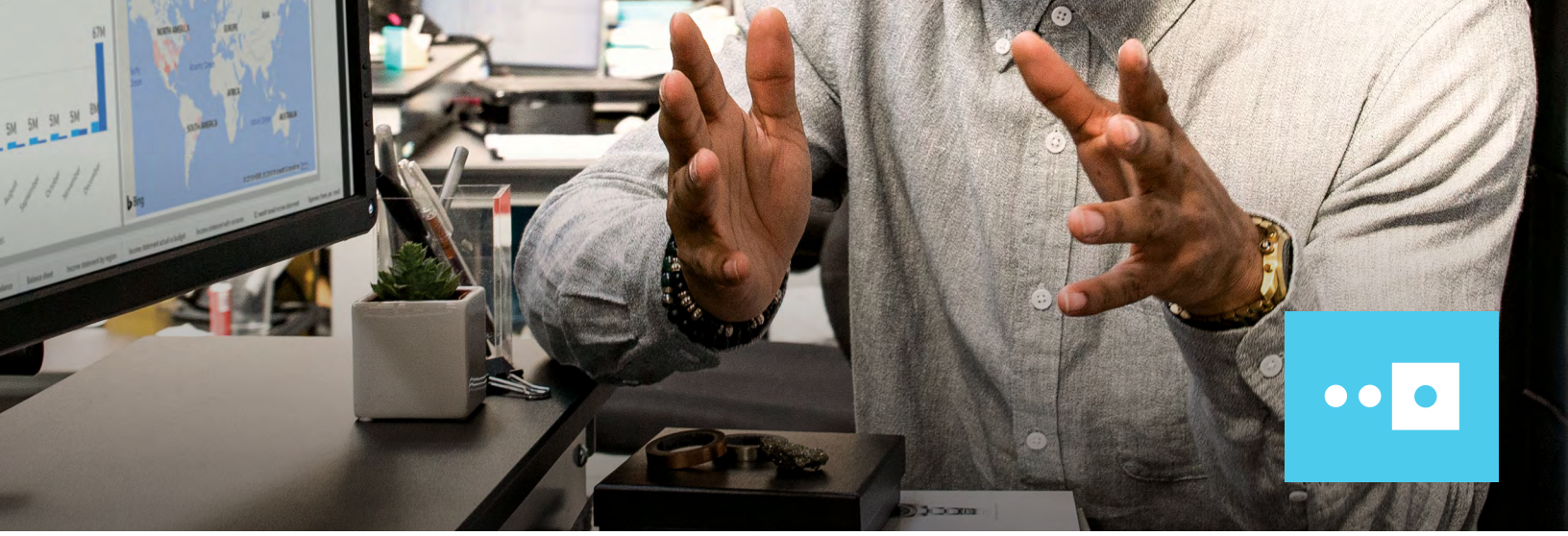
As the company continues to roll out its new infrastructure, it's piloting a solution to recognize vehicles' number plates based on real-time CCTV feeds. One benefit of this is loss prevention, in situations where a number plate is tied to a known offender in a published database.

They also plan to use the in-store cameras to do machine learning and AI on Azure Stack Edge virtual machines to predict stock levels and alert the staff, all locally, without the cloud.

4

Guide

Drive app value



Introduction

In the constantly evolving world of Dynamics 365, software as a service (SaaS) cloud applications, and changing business needs, it's important that your business applications evolve and are consistently delivering value.

This chapter outlines the approach to digital transformation and discusses strategies on how to shorten the time to value, cut through the noise, and focus on business outcomes.

The digital transformation journey of an organization usually involves various phases and levels of maturity. A green field implementation could start with a pilot or even a minimum viable product (MVP) targeted at a specific business process, then incrementally expand the solution with more capabilities for the users and onboard new business processes. Alternatively, replacing an existing system could mean that a significant amount of functionality needs to be made available at the start in order to have parity. No matter where you start, realizing the long-term goals of a SaaS-based digital transformation requires effective planning.

In this chapter, we explore how to create a digital transformation roadmap for your organization that can help accelerate the time to value while maintaining relevance in a constantly changing technology and business landscape. This guide can help you navigate the challenges

Key topics in this chapter:

- Think like a startup: start lean, build a successful MVP, evolve quickly
- Phases and incremental strategy
- Manage the change stream: business, user, and product
- Achieve hyper-scale with expansion approaches and enable organic growth
- Better together: satellite applications and aggregations

of expanding your solution and drive continuous, long-term business value through digital transformation.

Approach to digital transformation

Throughout this book, we discuss several concepts related to Success by Design, including how to initiate, implement, and deploy a project, but the scope of Success by Design isn't limited to a successful go live or operational excellence. The long-term goal of Success by Design is to create the right foundation for an organization to evolve their business application landscape and expand their digital transformation footprint.

Success by Design creates the foundation for an organization to evolve their business application landscape.

Every implementation related to business applications doesn't necessarily qualify as a digital transformation program aligned to the business; several systems are migrated to the cloud in their current form as a part of infrastructure modernization or to ensure supportability. In some cases, this is just the first step towards enabling digital transformation; opportunities could arise to take it further and deliver transformational value to the business rather than looking at it as just a tactical upgrade to projects run by IT with little involvement from the business.

The discussion in this chapter targets business-focused digital transformation programs and their expansion. Before we go into the details of expanding a solution to further drive value, let's discuss the fundamental approach to transformation: depicting your current business model, the factors that affect your business model, how they can trigger transformation, and defining your goals and scope of business processes for a transformation roadmap.

Represent the business model

Before you define a digital transformation roadmap, a good strategic exercise is to holistically develop an understanding of your business model. You can do this at an organization, department, business unit,

or team level to understand the value that you currently deliver, who your customer is, and how to deliver that value and the key activities and resources involved.

You could use several different techniques or frameworks to describe an organization's business model and how they create value. One framework that we discuss in more detail is the Business Model Canvas, developed by Alexander Osterwalder. It offers a simple, holistic, and easy-to-understand representation of a business model. This model isn't officially endorsed by Microsoft or part of Success by Design, but you can use the model to not only map your existing business model but also highlight what aspects of your business model are changing—which your digital transformation strategy will help you adapt to.

The business model represents the "why" and what it takes to deliver the value to customer; how we go about doing it is the business process definition.

Disruptions in the business model create opportunities for transformation.

Triggers

Disruptions and inflections in the business model create big opportunities for transformation. In a world that's changing at a pace greater than ever, businesses have to reinvent themselves more often, improve customer experiences, and attract and retain talent. The opportunities for impact are plentiful (for a deeper discussion, see Chapter 1, "Introduction to Implementation Guides").

Let's explore some examples of how changes in the business model, disruption in an industry, new regulations, or even external factors like global events can be a catalyst that triggers changes to business processes and applications.

- **Changes in customer segment** As businesses evolve their products and expand their target customer base and demographics, they often see that their approach to nurturing their potential customer base also needs to evolve. Targeted messaging that resonates and is delivered through relevant channels becomes extremely critical. Does your marketing automation and customer data platform enable this?

- **New channels** Customer channels that the business uses to deliver their product and services are evolving. For example, customers expect remote delivery of services via digital channels instead of an in-person visit to a store, which is further necessitated by evolving circumstances such as the COVID-19 pandemic. What does this change in channels mean for your order management systems or other related business processes?
- **Changes in value proposition** Businesses might differentiate themselves by coming up with unique customer value propositions in highly competitive sectors, such as offering a lucrative subscription model, to drive recurring revenue and shareholder value. For example, customers aged 25–40 are more open to subscription-based purchases versus outright ownership. The automobile industry now offers cars on a subscription basis and charges based on miles driven, as opposed to customers paying full purchase price. How are your processes and business applications affected by this change in business model?
- **Changes in customer relationships** An organization's approach to engaging with customers evolves with the business. For example, an organization could change from an aggressive customer acquisition phase to retaining and upselling within their existing customer base after acquiring a fair market share. How is your customer relationship management (CRM) strategy changing, what does this mean for your CRM system?
- **Changes in revenue streams** As businesses evolve, they create new revenue streams. For example, an energy provider in a highly competitive market might provide home services to expand their revenue stream, which triggers changes to customer processes and related business applications. Do you know how your revenue stream changes impact your business applications?.
- **Resources and sustainability** More organizations are committing to environmental sustainability. How do your manufacturing and operations track your carbon footprint, impact on emissions, and procurement from sustainable sources?
- **Custom manufacturing** With the advances in 3D printing technology, a manufacturing company can offer custom, made-to-order options to their customers. How does this impact your operations processes and enterprise resource planning (ERP) application?

- **Partnerships and acquisitions** Inorganic, acquisition-based growth is common in several industries, and so is the strategic partnership between organizations to grow market share where they have complementary services. How do these changes affect process alignment and your supporting business applications?
- **Cost and efficiency** New technologies offer efficiency advantages that can reduce operational costs. For example, delivery companies use electronic signatures instead of paper-based solutions to get real-time notifications and provide new value to customers. Are there opportunities to drive process efficiency using the latest technology?
- **External factors** Changes in the socioeconomic situation, political climate, and regulations can affect the way organizations conduct business. For example, the EU's decision to bring in the General Data Protection Regulation (GDPR) affected every business with employees and customers in EU, which required changes to their business applications to ensure compliance. How are new and changing regulations impacting your business?

Changes to the business model can kickstart digital transformation.

These triggers can all disrupt your existing business model. You can use the Business Model Canvas to highlight what aspects of your business model are changing and align your digital transformation strategy to these changes, which will eventually help your business adapt and grow.

Digital transformation goals

Changes to the business model (caused by a variety of triggers) can kickstart digital transformation. These transformation goals must be defined in business terms, with measurable metrics that could potentially involve several milestones. Let's look at an example in which an organization's goal is to create a 360-degree view of their customer. They want a holistic view of their customer, their activity history, product purchases, subscriptions, and more. But this goal doesn't relate to a business outcome or a measurable key performance indicator (KPI). A measurable goal for your digital transformation could be the business's ability to increase upsell to existing customers by X percent or reduce the time to address customer queries by Y percent. This requires a 360-degree view of the customer.

You can use objectives and key results (OKR) as a goal-setting framework for digital transformation.

Although most programs begin with a set of business goals, it's equally important to communicate and reiterate these goals to the IT and technical teams implementing solutions. In long-running transformation programs involving several technologies, teams and partners might lose sight of the actual business goals by letting technology become a distraction.

Business process

The next stage of the process is discovery, which involves key business stakeholders. With a holistic view of the evolving business model and transformation goals in place, you can identify the related business processes and applications that need to change. The discovery exercise should be focused on creating clarity around how the business processes and applications identified need to evolve to meet the corresponding transformation goals.

The scope of change could be automating a manual activity to improve productivity, capture data accurately to improve the effectiveness of strategy, and help your users do their job better and more efficiently. This will enable the business to generate insights about your customers by eliminating data siloes.

Prioritizing the changes to maximize and deliver measurable business impacts continuously (without having to wait for a multi-year transformation program to complete) keeps digital transformation in the foreground, with engagement from the business, end users, and executive sponsors maintaining its relevance. Business agility should be the key focus for long-running, comprehensive transformation, or you risk missed business opportunities and loss of market share.

For more information, refer to Chapter 7, "Process-focused solution."

Change streams

So far, this chapter has discussed the approach to digital transformation and how to develop a transformation plan for your application

and the process involved. This has traditionally been a top-down approach, starting at the business model and then narrowing down to business processes and applications that deliver the most impact. However, you should always keep in line with the broader theme of constant change and embrace the mindset of getting comfortable with frequent and inevitable change. In this section, we explore the change streams that impact the roadmap and scope of planned activities. Planning for how to deal with these changes throughout the program is a key determiner of success.

Business

As we embark on the journey of transformation and start building the business application based on the goals set by the business, it's also important to appreciate that during design and implementation, you may need to accommodate for further adjustments and refinements. This support for agility and change needs to be fundamentally built into the program. This is where some of the iterative methodologies could be beneficial, enabling us to respond to the change without it turning into a disruption.

Embrace the mindset of getting comfortable with frequent and inevitable change.

Those leading or adopting transformational change often find that it's not well defined in the early period, so adopting changes quickly and early is key—but so is flexibility as clarity reveals the resulting business model. When transformation occurs in an industry, what the industry will look like following the transformation is often not known. Companies must be able to adapt quickly in the middle of the project to incorporate these inevitable changes.

User

A key stakeholder in your transformation plan is the business user, who interacts with the application daily. If the process being implemented in the application doesn't meet the needs of the user, or the application doesn't consider working patterns and end user usage, it could lead to poor adoption and lack of engagement. Incorporating user feedback continuously throughout the process using well-defined and frequent touchpoints is key to achieving your transformation goals.

Your approach to designing and developing the application needs to be user-centric; it must clearly articulate how the change being implemented results in value for the user.

You should expect that user experience expectations constantly evolve as users are exposed to a variety of software products both in the enterprise and consumer space. For example, the predictive text fill feature popularized by web search engines is now the expected search experience in business applications. This makes it even more important to adopt newer features and improvements delivered by SaaS applications, which takes us to the next change stream: product and technology.

Product and technology

In the world of cloud-based SaaS applications, you can expect a constant flow of new capabilities, improvements to existing features, and transitions to the latest technology paradigms, which can impact your current application. Traditionally, business applications were built by IT, with features requested by the business and end users. With the SaaS cloud, the service providers themselves are invested in understanding the latest trends in business and delivering innovative capabilities to help their customers stay ahead of the competition. For example, a business trying to transform their sales application can assemble an application using the best-in-class sales capabilities offered by the SaaS provider instead of spending years in research, design, and development to build from scratch, which would make the application obsolete by the time it launches.

Continued investment in microservices by Dynamics 365, such as planning as a service, can transform a core business model for companies allowing them to run material requirements planning (MRP) as needed to drive a significant and transformative way of managing demand and the supply chain.

The enhancement delivered via continuous updates has a much shorter adoption timeframe when compared to traditional major version upgrades.

SaaS application providers that are competing to build and deliver business capabilities for customers are helping advance business applications in way that was unimaginable just a few years ago. Applications that were just forms over data (mostly passive data capture systems used to track, view, and report on known data) have evolved into applications that can automatically capture data, learn from the data, deliver insights, and guide users to the next best action. This makes it extremely important for business to watch for new capabilities being made available and adopt them to accelerate their transformation to be a differentiator in the industry. The key value proposition of SaaS is also the ability to tap into the enhancements and features that are based on broad market research. Activating these features can accelerate your transformation with minimal effort and without any development cost.

Your SaaS application provider is no longer just a software vendor providing an off-the-shelf product—they're a strategic partner to your business with significant impact on your ability to implement your strategy. Developing a strong relationship with the SaaS providers to not just learn about the roadmap but also influence the roadmap through strategic feedback ensures that you're getting the most value out of the product.

External

External economic, social, and political drivers can disrupt your transformation plan. The COVID-19 pandemic is an example of how supporting remote work and creating online collaboration channels became a top priority for most customers in 2020. This required coordinated changes from infrastructure, to network, to the device and application layer across the IT level of organizations. Although it's difficult to foresee and plan for external events, the iterative approach to delivering continuous value in smaller batches allows you to pause and pivot as needed. For example, consider an organization on a long-term transformation program that releases a major capability to users once a year versus an organization that has adopted the DevOps culture of delivering value with regular bi-monthly or monthly updates. The latter company can realize value from investments and is better positioned to pivot when change demands it.

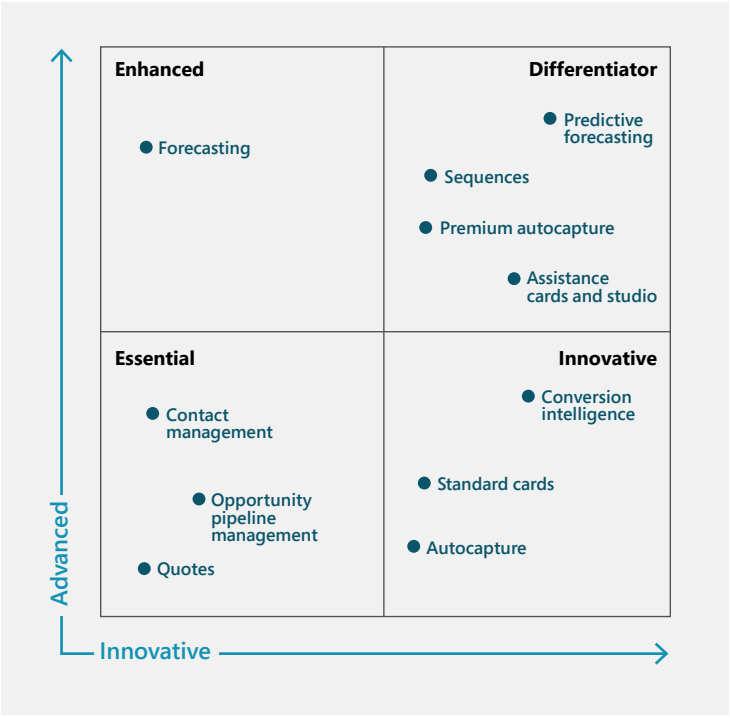
Transformation map

So far, we have discussed a high-level approach to digital transformation and the various change streams that you need to consider. The ever-growing product backlog of feature asks, change requests, and product capabilities can overwhelm and potentially distract from the drivers for transformation. To determine time to value, you need a shared understanding between the business, IT, and users on how a product feature or change aligns with the goals of transformation and impacts the maturity of process implementation.

One way to visualize and communicate your digital transformation goals is to plot the application's capabilities in a transformation map (**Figure 4-1**). You can adjust the definition of each quadrant to your own goals and ambitions for the process, but you can easily get started with the template we offer here. If all the feature asks, changes, and SaaS product capabilities are plotted on this quadrant, you can determine the current state of your business process implementation and agree on the future state aligned to your transformation goals.

For example, a sales system that only has the basic features of contact management and opportunity management may be considered essential, whereas a system that can use AI-driven insights with advanced forecasting might be a differentiator that gives you competitive advantage in your industry.

Fig. 4-1



Creating a dashboard for stakeholders that organizes the requirements and features in this fashion can bring great clarity—not just during initial implementation, but for future increments. It's also important to prepare for time-based decay, in which a feature that is considered a differentiator today might be considered essential in a few years. Similarly, something that is innovative or even considered ahead of its time could become a differentiator in the near future. You should plan to revisit and refresh this

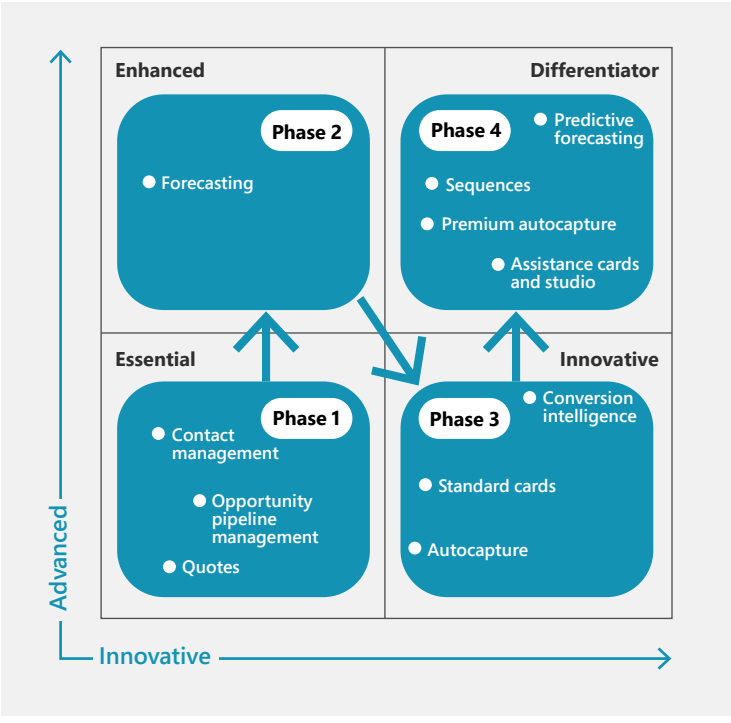
transformation map for each phase, product release, pivot, or strategic inflection in your business.

Phases and increments

Planning and scoping distinct phases of your digital transformation program directly impacts the time to value realization. What is delivered and when it's delivered also has an impact on the level of engagement from the stakeholder and the end user's adoption. These phases and increments are based on the goals agreed upon with the business and allow you to select the appropriate capabilities in the transformation map.

It seems reasonable to start with the essentials and basics in Phase 1 of a solution and then deliver enhancements with most the innovative, differentiating features in later phases of the program. But if the most valuable parts of the transformation are planned for much later in the lifecycle (**Figure 4-2**), you risk stakeholders losing interest, or user perception changing (which is hard to correct). Additionally, the features that were considered differentiators could now be essential due to time decay. A transformation program that fails to deliver beyond the essentials or the initial MVP drives little transformation.

Fig. 4-2



Without the core features, you can't deliver enhanced capabilities. You should plan the phases so you can pick up elements from various quadrants and deliver a holistic solution that has the key process elements, integrations, and reporting that delivers value (**Figure 4-3**). Look for quick wins: the latest SaaS features that you can easily adopt with little effort to bring innovative and differentiator elements into the earlier phases.

Digital transformation is as much about people as the process and technology. The phases and incremental design should also help product owners maintain the right level of engagement from users and stakeholders. This drives excitement and

curiosity, but most importantly feedback. You could draw an analogy from a TV series that shows teasers at the end of each episode or the end of each series to attract viewers back to the program. Do the same for your digital transformation story by delivering value beyond the essentials in each phase so customers come back for more after each sprint.

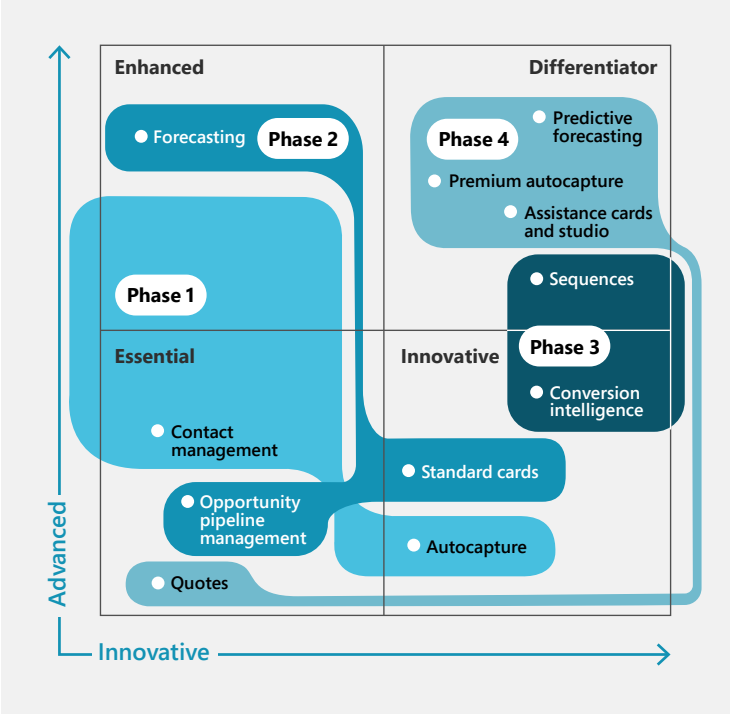
In the next section, we look at how the right approach to MVP strategy when getting started with your digital transformation journey can help get early feedback and drive value sooner rather than later.

Minimal viable product strategy

The concept of MVP, which enables you to quickly build, measure, learn from feedback, and pivot if necessary, has been well established in the startup world.

Bringing the invaluable lean startup mindset to digital transformation can help organizations deal with uncertainty from different change streams and find the value that drives maximum business impact in the least amount of time. Even in the context of digital transformation, MVP is often used to test the value of a solution to a business and gather feedback.

Fig. 4-3



When translating the lean startup methodology to a business application, MVP should not be a solution with just the basic and essential features; you should create a solution that will help transform your process and deliver on the most important transformation goals with the least amount of effort. Most importantly, an MVP is not the end state; programs may be stuck in an MPV state for years.

Going back to the example of a sales application, let's consider if users are currently using a home-grown sales system with a form to enter and view opportunity data. The system has a backend to

store this information, which is also used for management reporting. The business has invested in a Dynamics 365 SaaS-based sales application after assessing its rich features, out-of-the-box AI insights, and integration capabilities with Power BI for reporting and analytics, which allows them to improve their opportunity closure rate and get accurate growth forecasts. A common approach is to first implement the out-of-the-box opportunity management process and reports, and plan future phases to implement AI-based insights with Power BI. Although you can easily follow this route, it doesn't achieve the actual goal of sales implementation. The system delivers an improved user experience but fundamentally does the same thing as the home-grown sales system, with little direct value to the business. Alternatively, using embedded intelligence with opportunity scoring and relationship health can help users target the right opportunities and improve their close rate, which directly impacts business.

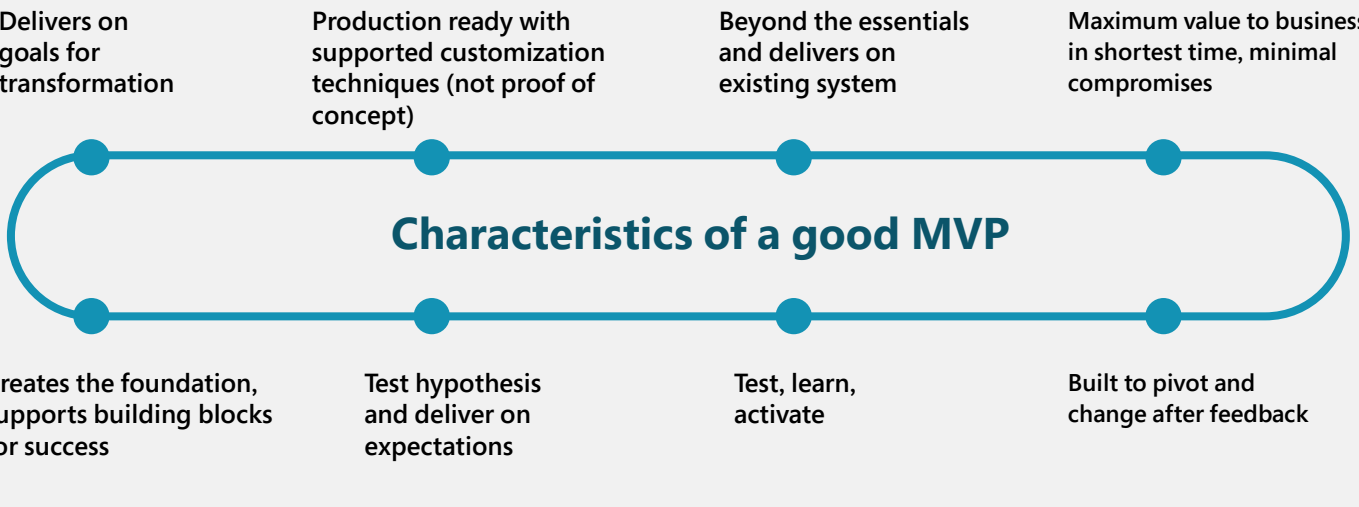
"MVP is not the smallest product imaginable, though it is the fastest way to get through the build measure learn feedback loop with minimum amount of effort."

– Eric Ries, author of *The Lean Startup*

A good MVP has the following characteristics (**Figure 4-4**):

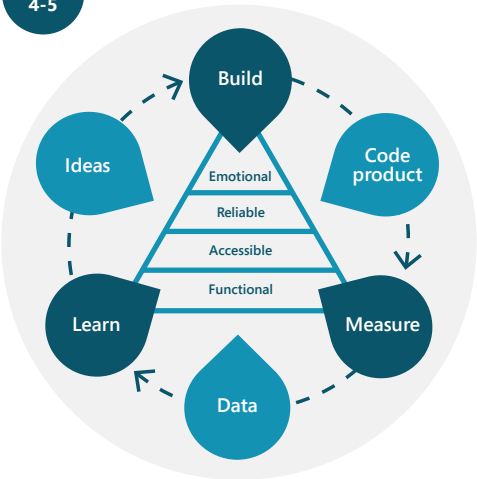
- Delivers on the most important of the digital transformation goals.
- Is production-ready and uses supported customization techniques (this is different from a proof of concept or prototype designed to prove feasibility or get specific usability feedback).
- Goes beyond the essentials and delivers on top of the existing capability of the incumbent system.
- Focuses effort on delivering maximum business value in the shortest amount of time, (you could compromise on some experiences that aren't directly aligned to business outcomes).

Fig. 4-4



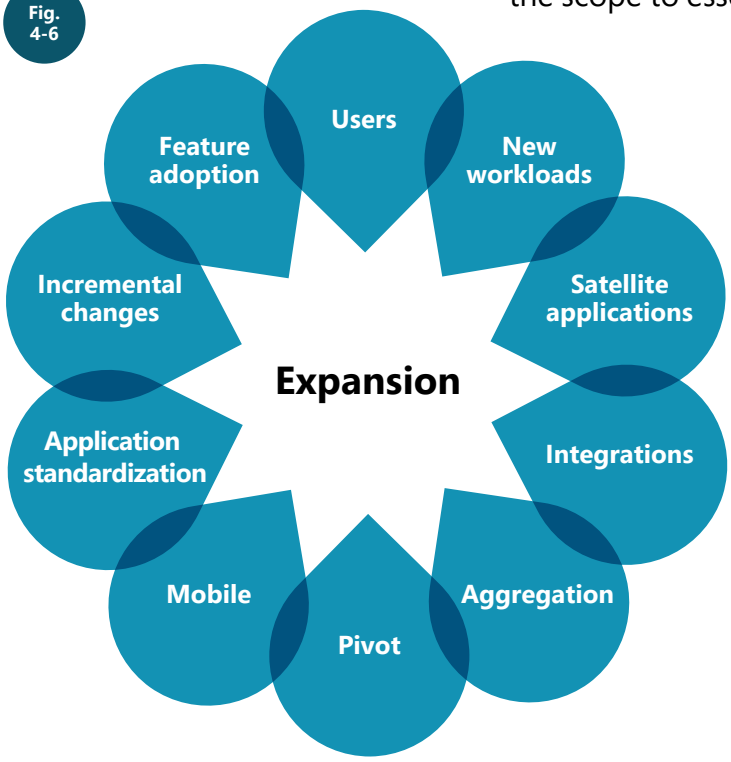
- Is built in a way that allows for quick changes based on feedback, making it possible to pivot. This translates to a greater reliance on low-code configuration versus highly customized, professionally developed, code-heavy experiences.
- Allows you to test and learn on the latest SaaS product features that could be activated (this requires some level of change management but almost no development effort).
- Allows you to test the hypothesis of a specific feature or solution delivering the expected business outcome and adjust based on what you learn.
- Creates the foundation for long-term success—it should support a building block mindset in which you keep adding without disrupting previous steps. (For more information about the architectural skills to realize this approach, see Chapter 6, "Solution architecture design pillars.")

Fig. 4-5



Although the MVP approach naturally works with new greenfield implementations, you may have existing apps that are being replaced or migrated to another platform. Your scope should consider the existing functionality to ensure parity, but shouldn't try to mimic or replicate the experiences of the outgoing application. Focus your MVP strategy on delivering the most value to the business sooner without limiting the scope to essentials (**Figure 4-5**). An MVP with a very broad scope that takes years to deliver may cease to be an MVP.

Fig. 4-6



Drive expansion

All the approaches and techniques we have discussed so far in this chapter can help you create an effective digital transformation roadmap for your business application. They're relevant to the Initiate phase, but you can apply these methods iteratively or revisit them for further expansion. This section focuses on expanding the usage and scope of business applications in different areas (**Figure 4-6**) to drive even greater business impact, while considering additional impact on the following:

- Security and compliance
- Application lifecycle management (ALM)
- Administration and governance
- User readiness and change management
- Environment strategy
- Limits and capacity
- Data and integrations

Users

The most common expansion scenario for a business solution is adding more users. This could be the result of onboarding new regions or countries onto the same application. User-only expansion sounds straightforward, but you could have data migrations on a live system, security and compliance challenges based on the region, or other impacts (**Figure 4-7**).

Mobile

Ensuring support on multiple devices, at times with offline capabilities, could be critical to adoption. In some cases, this needs to be a part

Fig. 4-7 Users

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
High	**Low**	**Low**	**Medium-high**	**Low**	**Low**	**Medium**
Potentially high impact based on regional regulations	If the solution remains the same, impact on ALM is minimal	Based on the environment strategy and the creation of additional production environments, you could see medium impact	Full-fledged user change management and readiness effort is required	Creation of a new production environment impacts the environment strategy	Although each user usually comes with their own capacity integration API, capacity, and storage could be impacted	Usually, data migration can be complex on a live system and in case of additional environments, integrations might have to be replicated across environments

of the initial release (such as in warehousing systems in which goods must be scanned when picked up), whereas in others it could be a quick follow-up. Most out-of-the-box capabilities work as is on mobile apps, with additional optimizations to support the move experience. Our example sales mobile application offers the core sales application features with additional targeted mobile experiences for field sellers (**Figure 4-8**).

Incremental changes

Incremental changes are primarily driven through the feedback and change requests from users and the business. These changes help ensure that the application meets the expectation of ongoing enhancement and continues to build on top of the initial MVP to maintain relevance. It's still important to look at these improvement requests through the business value and transformation lens (**Figure 4-9**).

Feature adoption

As we have seen when discussing change streams, SaaS products make significant investments in researching and implementing business capabilities that their customers can easily enable. This is a key value proposition of the SaaS cloud, and can play a key role in deriving value from your investments. With continuous updates in Dynamics 365,

Fig. 4-8 Mobile

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
High	**Low**	**Low**	**Medium-high**	**Low**	**Low**	**Medium**
Potentially high impact based on the regulations and need for mobile device management or mobile application management features	If the solution remains the same, impact on ALM is minimal	Additional device policies could have an impact	User readiness effort is required to ensure adoption on mobile and account for any variations	Should not have any significant impact	It should not have any significant impact on user capacity	Custom embedded integrations might require work to make them responsive for mobile consumption

new features are made available to customers. A business analysis of the new features with technical impact analysis on the existing solution and customizations (**Figure 4-10**) is a key business as usual (BAU) activity that must be conducted in collaboration with key stakeholders. Most of these feature improvements are made available to customers with no additional cost and can help advance your application to the innovative or differentiator quadrant.

Fig. 4-9

Incremental changes

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Low	**Low**	**Low**	**Medium**	**Low**	**Low**	**Low**
Changes within the approved SaaS service boundaries and data loss prevention policies should have limited security and compliance implications	No major impacts if the solution doesn't have new PaaS components	No expected impact as long as the data boundaries don't change	User readiness effort is required	Usually remains unchanged	Additional integration and app workloads could have some impact	Assuming no integration changes, impact is minimal

Fig. 4-10

Feature adoption

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
High	**Low**	**Low**	**Medium**	**Low**	**Low**	**Medium**
Usually no impact except when connecting to external data sources	No major impact expected	No expected impact as long as the data boundaries don't change	User readiness effort is required to use the new features	No impact	Usually no impact, sometimes features might require additional storage and capacity	Features might generate additional data but shouldn't impact integrations

New workloads

Dynamics 365 provides several business applications that are built on the same scalable power platform. As end users adopt your application, you will find additional opportunities to onboard related workloads that could help eliminate data siloes by enabling stronger collaboration and data sharing. For example, to achieve seamless collaboration when you're onboarding a marketing workload in addition to sales, the sales team can get visibility into the nurture programs their customers are part of, extend invite to marketing events, and so on. Similarly, the accounts payable team could expand and move into a centralized accounts payable processing system. However, new workloads can have significant impact (**Figure 4-11**).

Satellite applications

When deployed, your business application covers core use cases and scenarios, some of which may only be relevant to a specific team, region, or role. In such scenarios, you can deliver such targeted capabilities via a satellite application. Satellite apps are a good opportunity for user-developed apps and automations while using the data model of the core application. You could also use these applications for incubation before incorporating it in the core app. Regarding areas of impact (**Figure 4-12**), it's important to have strong governance around the data model and creation of additional data stores, which can lead to data fragmentation.

Fig. 4-11 New workloads

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Low-medium	**Low-medium**	**Low-medium**	**Medium-high**	**Low-medium**	**Low-medium**	**Medium-high**
This could be medium impact based on the app; core Dynamics 365 apps might have been already approved	App-specific ALM requirements can impact your release and build pipelines	Some apps could require additional admin tasks	User readiness effort is required, but existing users will benefit from a familiar UI	New apps on existing environments won't have a major impact	Additional apps can impact storage and tenant API capacity	Data and integration needs for the new app can have an impact

Integrations

Integrations can play a key role in eliminating data duplication and improving data quality. Most importantly, they reduce the need for users to navigate multiple applications, which prevents context switching. The core integrations should be part of the initial phases and not be left to future expansion phases. Even the out-of-the-box integrations with existing productivity applications like Microsoft 365 can positively impact adoption and make the information seamlessly available across apps while enabling stronger collaboration. However, an abundance of caution is needed; integration can also lead to performance issues and increase the overall complexity and cost of maintenance (**Figure 4-13**), so using the right patterns and practices is critical.

Aggregation

Aggregation can help consolidate multiple business applications with significant overlap of data and processes into a single application. Aggregation is different from integration—it completely replaces the application and brings the core feature set into an existing application instead of exchanging or embedding information. For example, a commercial banking sales system could become an aggregator for all non-personal banking sales systems.

Fig. 4-12

Satellite applications

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Low-medium	**Medium**	**Medium-high**	**Medium**	**Medium-high**	**Low-medium**	**Medium**
If the data flows in accordance with data loss prevention policies, impact should be low	Managing the lifecycle of satellite apps requires changes to existing ALM processes	As the satellite apps grow organically, the appropriate governance policies need to be in place to manage them	User readiness effort is low for citizen-developed community apps, but might require readiness effort if they're adopted more broadly	Depending on the ALM and citizen development strategy, the environment strategy is impacted	Potential impact to capacity; you may also need licenses to support the makers and consumers if they're not already covered by existing licenses	You should avoid fragmenting the data with additional apps using their own data stores, but it may be required in some cases

The cutover approach for old applications is critical; make sure to avoid parallel usage leading to fragmentation of data and other aftereffects (**Figure 4-14**).

Application standardization

This model of growth is driven by creating a generic application that isn't targeted at a specific process but a host of similar processes. The application could even provide a framework that enables the business to onboard themselves. This can achieve hyper-scale, in which you can broadly t-shirt size hundreds of processes or applications that serve more than one business process.

Fig. 4-13

Integrations

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Medium-high	**Medium-high**	**Medium-high**	**Low-medium**	**Low-medium**	**Medium-high**	**High**
Depending on the data flows, additional security approvals may be needed for integration pattern	ALM process could be impacted depending on complexity	Additional monitoring and integration telemetry is needed to support reconciliation of failures	Depending on frontend versus backend integration, the impact and readiness effort could vary	You may need to have stubs or downstream integration environments	Potential impact to API consumption	Expect impact on data movement and make efforts to ensure integrations are built to standard

Fig. 4-14

Aggregation

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Medium-high	**Medium-high**	**Medium-high**	**Medium-high**	**Low**	**Medium-high**	**Medium-high**
Level of impact depends on data classification	Existing ALM processes are changed	Admin processes are consolidated	User readiness is required	This will have impact on environment strategy, requiring a potential merge	Additional users and increased data volumes impact capacity	High impact with respect to data alignment and integration consolidation

A general-purpose application allows scale, but can require additional training and user readiness, so it must be thoughtfully done to be successful.

For example, consider a generic case management system that can accommodate hundreds of different case processes from different internal business functions that broadly involve similar steps. The business can create a team of resolvers, configure service-level agreements (SLAs), provide email notification templates, and create Microsoft Word or Excel templates to capture non-standard data, but they can still use the generic application for core case management capabilities. All these actions have important considerations (**Figure 4-15**).

Pivot

Pivots aren't necessarily expansion, but can trigger major change to an application, which could fundamentally change and further broaden its scope (**Figure 4-16**). For example, a targeted Power App used by a small team could go viral within an organization, and you must pivot to make it an organization-wide application.

Conclusion

As we have seen throughout this chapter, achieving true digital transformation is not a sprint, but a marathon—it's a continuously evolving

Fig. 4-15 Application standardization

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Medium-high	Medium-high	Medium-high	Medium-high	Low	Medium-high	Medium-high
Once approved for appropriate data classification, more processes shouldn't have an impact	Existing ALM processes shouldn't change for a specific template	There could be process specific configurations required for each new workload on the template	User readiness is required for each new process	Should have minimal impact	Additional users and increased data volumes impact capacity	There could be an increase in the data generated by each process

Fig. 4-16

Pivot

Security and compliance	ALM	Admin and governance	User readiness and change management	Environment strategy	Limits and capacity	Data and integrations
Medium-high	Medium-high	Medium-high	Medium-high	Low	Medium-high	Medium-high
Application will need to undergo further scrutiny	Existing ALM processes will potentially change	Admin and governance scope will need to be reevaluated	User readiness might be required	Potential changes to the overall environment strategy	Additional users and increased data volumes will impact capacity	Expect high impact to data and integration based on the pivot

journey with iterations and changes that need to be effectively managed. It starts with understanding the changes to your business model, using triggers to accelerate transformation, and adopting the lean startup mindset to build with an intention to test, learn, and change while staying focused on driving maximum business value in the shortest amount of time.

Going live with your MVP is the first step to driving value. You will iteratively refresh your transformation roadmap, revisit and refresh your program goals, adopt new product capabilities, and drive meaningful expansion while staying true to the core program value and transformation goals. Following this approach can establish your organization as a leader in your industry.

The holistic value of SaaS is much broader; it goes beyond just the technology and features of your application. You're investing not only in the technology but also in the research and innovation that drives the product changes. Simplifying experiences and improving interoperability further reduces costs and drives business innovation.

As a final takeaway, we can't emphasize enough the importance of staying engaged and investing in learning. Stay on top of key product announcements, industry trends, and research studies on technology trends. Engage with product leaders via events, meetups, and advisory boards. This will significantly boost your ability to achieve success.

Checklist

Approach to digital transformation

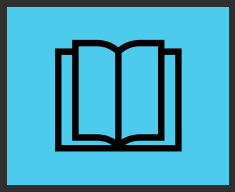
- [ ] Have a clear understanding of the core business model and how the application being developed impacts the key elements of the business model.
- [ ] Clearly articulate and communicate business transformation goals with specific measurable KPIs.
- [ ] Ensure the implementation team understands the business processes in scope and the business teams are closely engaged to optimize the process where needed.

Change stream

- [ ] Account for the various change streams that will impact the program and ensure budget and timelines are adjusted to accommodate.

Transformation roadmap

- [ ] Create an effective MVP scope that goes beyond the bare minimum to accelerate value realization.
- [ ] Define your transformation roadmap and structure the phases to maximize time to value and drive maximum business impact.
- [ ] Identify the expansion models and incorporate that into your program phases.
- [ ] Adopt the latest capabilities to deliver continuous innovation to business.



Case study

Dynamics 365 helps adapt business processes to change

One of our clients, an accounting network, is facing unique and unexpected challenges. The COVID-19 pandemic has inhibited face-to-face work, threatened global supply chains, and shifted regulatory and political landscapes.

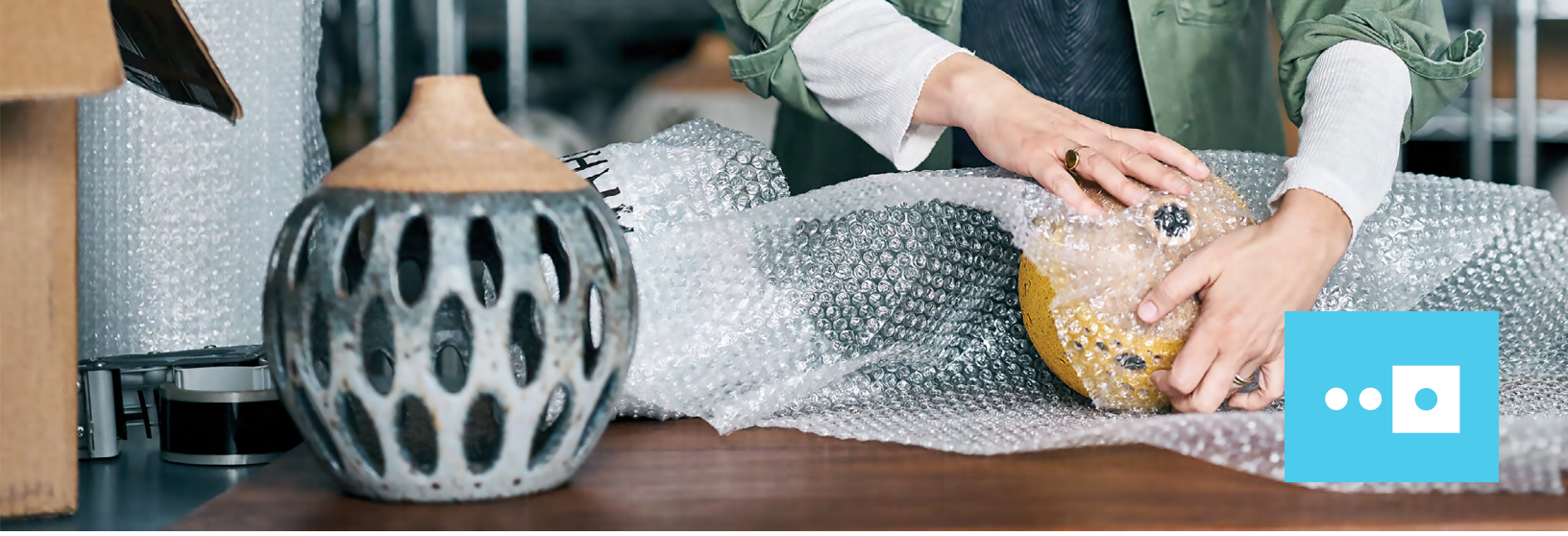
To maintain business continuity and stay connected with customers, the company is adapting their digital selling techniques with the help of Dynamics 365. With 360-degree views into each profile, sales staff can work remotely without interrupting the customer experience. The solution also works with the firm's risk assessment tool to identify vulnerable areas caused by the pandemic, allowing staff to create proactive plans for their clients. The company is keeping their business personal, up-to-date, and resilient by optimizing digital solutions in a time of uncertainty.

This successful pivot clearly highlights the power of Dynamics 365 as an adaptable SaaS cloud platform that can not only deliver strategic solutions but also adapt to changing business needs quickly.

5

Guide

Implementation strategy



Introduction

Organizations embarking on a digital transformation journey consider implementing business applications as key critical milestones for their business success.

The journey from a business vision to a successful implementation of it requires the business and IT owners to define an overarching strategy that acts as a framework to plan for predictable outcomes. An implementation strategy defines these strategic choices and decisions made in planning, implementing, and deploying Microsoft Dynamics 365 applications.

Along with new technology comes substantial change to a company's business processes. It is critical to choose a process-focused, user-centric implementation strategy and methodology that not only supports digital transformation but also helps drives change management and user adoption. All of these factors are critical to making an implementation successful.

In this chapter, we explore common industry methodologies and look at common deployment strategies and high-level views on change management strategy.

Key concepts that will help in defining a successful implementation strategy.

1. Key aspects
2. Business drivers
3. Success metrics
4. Methodology
5. Deployment
6. Change management
7. Conclusion

Key aspects of implementation strategy

The foundation of a successful implementation begins with planning for key factors that define the overall implementation strategy. Let us explore these key considerations.

Understand vision and business drivers

Project teams should be aware of the vision and strategic intent of the customer stakeholders. Having a clear grasp of the key reasons behind the implementation of a project and a collective understanding helps the project team stay aligned with the project's vision of addressing significant business challenges.

It is important that senior management and project sponsors are clear about the business requirements and help to establish the scope and expected outcomes of the project. These goals help drive any decisions that may be needed during the implementation cycle— for example, prioritizing delivery of certain functional features over others.

This helps deliver a solution aligned with business needs and allows for change management and a higher user adoption.

Identify key success metrics

After establishing a common understanding of the business challenges and priorities, it is equally important to have an agreement on what the key performance indicators (KPIs) that measure success look like. The KPIs should be tangible metrics, for example opportunity conversion rate, or average call handling time.

Understanding and identifying the KPIs is critical to ensuring the scope being delivered is aligned with the expectations from the business stakeholders and generates a tangible return on investment on the Dynamics 365 investments.

Some of the examples of success metrics/KPIs include:

- Improve the opportunity conversion rate from X % to Y % over z period.
- Reduce sales cycle from x days to y days over z period.
- Increase net new customer adds from x to y over z period.
- Reduce average call handling time from x mins to y mins over z period.
- Improve service request turnaround time from x days to y days over z period.

The KPIs mentioned above are not exhaustive, customers have specific KPIs that cater to their business scenarios. It is important to acknowledge that each KPI the customer identifies and prioritizes has a direct impact on the functionality that would be deployed on the Dynamics 365 Business Applications solution. Consequently, having a definitive list of metrics to refer back to enables prioritization of the right use cases and allows customer stakeholders to gauge the project success in a tangible way.

Identify and allocate right resources

Setting up a productive team structure is critical to the success of the project, which includes identifying the stakeholders, business process owners, the team that delivers the projects, and the end-user representatives who can be engaged from the beginning of the project lifecycle.

The key roles required for all projects can often be grouped into two types of team structures.

- **Steering committee** The group of senior stakeholders with the authority to ensure that the project team stays in alignment with KPIs. As they monitor the progress they can help make decisions that have an impact on overall strategy, including budget, costs and expected business functionality. Steering committees usually consist of senior management members and group heads whose business groups are part of the identified rollouts.
- **Core implementation team** This is the team doing the actual execution of the project. For any Dynamics 365 implementation project, core implementation teams should include project

manager(s), business subject matter experts, solution architects, functional and technical consultants, developers, testers, change management advisors, and user champions.

Process owners engagement is critical to ensure complete understanding of business processes. Their understanding is required in case of any process changes that may result from alignment with out of the box features. User representation should also be planned properly to adequately cover all the personas. Lack of availability of any of these may result in delays and/or incomplete understanding of processes and also impact user adoption. The involvement of process owners and users can be staggered depending on the implementation phase. This should be planned early on to allow business teams to make the required changes to their schedules and be able to contribute efficiently during the project lifecycle.

The project may require additional roles depending on the scope and methodology. Some roles may need to join temporarily during the project lifecycle to meet specific needs for that period.

For more details on team setup considerations, refer to Chapter 2, “Success by Design overview,” and Chapter 8, “Project governance.”

Allocate budget and funds

Planning a budget for Dynamics 365 cloud applications typically includes costs such as subscription, storage, implementation lifecycle activities of designing, testing and deploying the solution, migration from legacy applications, training, change management, and continuous improvements.

Project lifecycles require constant assessments with careful considerations and actions for the variance between budgets versus actual costs.

A common decision that most teams face is finding the right balance between delivering a high value solution faster by using the off the shelf capabilities versus extending the product capabilities to implement the business requirements and needs. Extending the Dynamics

Forrester's Total Economic Impact (TEI) of deploying various Dynamics 365 apps is a good recommendation to assess the costs and benefits associated with Dynamics 365 Implementation.

Some of the exclusive TEI done by Forrester are:

The Total Economic Impact of Microsoft Dynamics 365 for Finance And Supply Chain Management

The Total Economic Impact of Microsoft Dynamics 365 Field Service

The Total Economic Impact of Microsoft Dynamics 365 Customer Service

365 applications not only requires initial development costs but also maintainability and supportability costs in the long run. This is an area where implementation teams should carefully revisit the cost impact. In the cloud world with rapid availability of new features and capabilities, there is a need to rethink the investments required for extending the application. Refer to Chapter 15, "Extend your solution," for more details on assessing this impact.

Project management teams should give appropriate weight to areas that help them stay aligned with the overall project goals and objectives. It has often been seen that as project scope and timelines change, the activities that fall towards the end of the project lifecycle are usually descoped to meet budget constraints. These activities can include performance testing, data validation testing, training, and change management, which, when compromised, can significantly reduce the quality and success of the implementation. It is important to revisit the project goals and success factors that are defined during initial stages to carefully revise the scope and timelines without compromising on overall quality and user centric approach.

An overall approach is recommended since it helps define the total impact by identifying the cost, benefit, flexibility, and risk factors that affect the investment decision, helping organizations improve their overall business goals of winning, serving, and retaining customers.

Choose a methodology

Before you discuss choosing a methodology for your Microsoft Dynamics 365 project, we need to understand why a methodology is important for Dynamics 365 projects.

Why methodology is important

A methodology is a prescriptive definition of activities to be undertaken for a given project or engagement. It describes the use of a collection of methods to achieve predictable outcomes.

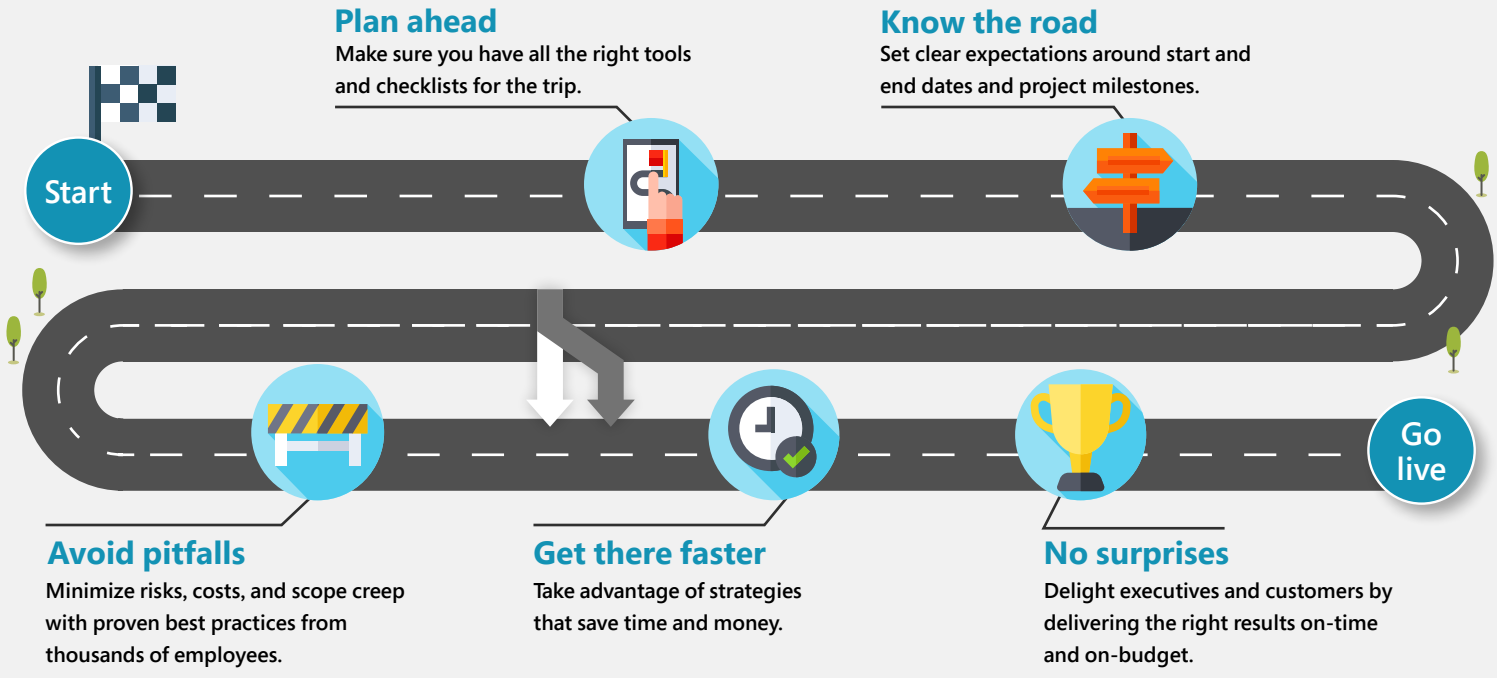
Methodology is the core of project execution that allows the team to take the right path towards:

- Improved consistency and predictable outcomes.
- Setting clear definitions of project phases, milestones, deliverables, and entry and exit criteria to each phase.
- Setting clear definitions of roles and responsibilities required for the project.
- Reducing risks of missing critical activities required for a successful outcome.

Following a methodology is akin to taking a planned journey; the roadmap to success (as illustrated in **Figure 5-1**) involves the following:

- **Plan ahead** Ensure you have clearly defined the destination (project scope), and have all the right tools (resources, governance, and methodology) and checklists (project monitoring) before starting this journey.
- **Know the road** Define the route (solution blueprint) to set clear expectations on start and end dates at milestones so you can meaningfully monitor progress and the team understands what is expected from them.
- **Avoid pitfalls** Utilize proven best practices, as advocated in the methodology, that minimize risks, costs, and scope creep.
- **Get there faster** Take advantage of strategies that help you save

Fig. 5-1



time and money. Plan, action, and regularly monitor and adjust to ensure you are taking the most efficient route to your destination.

- **No surprise** Delight users by delivering a user centric solution and business stakeholders by delivering results on-time and within budget.

Iteration and sprint is a timeframe in which teams deliver incremental value in the form of working tested software. Iterative approaches use the term iterations while Scrum approaches use the term sprints.

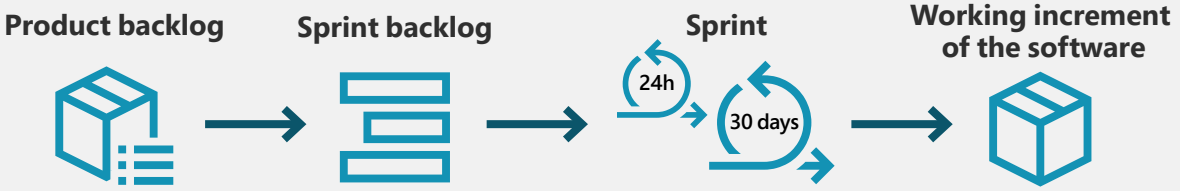
Choose an implementation methodology

Let us take a look at the specific methodologies that can be used for a successful Microsoft Dynamics 365 implementation.

We often see three types of models used to implement Dynamics 365 Business Applications projects.

- **Waterfall model** A delivery execution model based on sequential, steadily flowing, like a Waterfall, series of phases from conception, initiation, analysis, design, development, testing, deployment, operations, to maintenance.
- **Agile model** Agile development is not a methodology in itself, but an umbrella term that describes several Agile methodologies. Some of the industry known methodologies are Scrum, XP, DSDM, Sure Step for Agile (for Business Applications).
    - **Scrum methodology** A delivery execution methodology based on Agile principles where requirements are in flux or unknown prior to beginning a sprint. Sprints involve requirements analysis, design, build, test, and user review. Sprints are a fixed duration (usually 30 days), as illustrated in **Figure 5-2.**
- **Hybrid model** A delivery execution model based on Agile principles where requirements within the context of business processes are defined, analyzed, and prioritized and an overview design solution blueprint is generated prior to beginning the sprints. Sprints involve refining designs, build, test, and user reviews. Sprints can be of varying durations.

Fig. 5-2



Now let's take a deeper look at each of these methodologies.

Success by Design Framework is methodology agnostic and is aligned to ensure proactive guidance and predictable outcomes irrespective of chosen methodology. For more information, refer to Chapter 2, "Success by Design overview."

Agile

An Agile implementation is an iterative approach that uses a number of iterations or sprints.

In Microsoft Dynamics 365 Business Applications projects, the majority of requirements are delivered by the packaged software. There are specific challenges and tasks that are out of the box and need to be managed and mitigated as part of the project methodology with the user stories contained in the form of a backlog. Using those stories, you carve out plans and within them a number of iterations or sprints of development and testing are executed. Within each sprint, you have a number of user stories outlining the requirements.

Agile promotes a collaborative process between the resources that own and specify the requirements for the solution and the resources responsible for the development and rollout of the solution.

The idea is to release software faster, take early and regular feedback, adapt, and release again. This cycle continues until all the requirements are met or the backlog is cleared.

An example of Agile practices using the methodology prescribed by "Sure Step for Agile" is shown in **Figure 5-3**.

Waterfall

The Waterfall approach to solution delivery is a sequential process that depicts a linear flow of activities from one phase to another, culminating with the solution being promoted to production and then into operation.

This is a traditional methodology of implementing on premises business applications. It is a linear, noniterative approach of implementing and delivering projects.

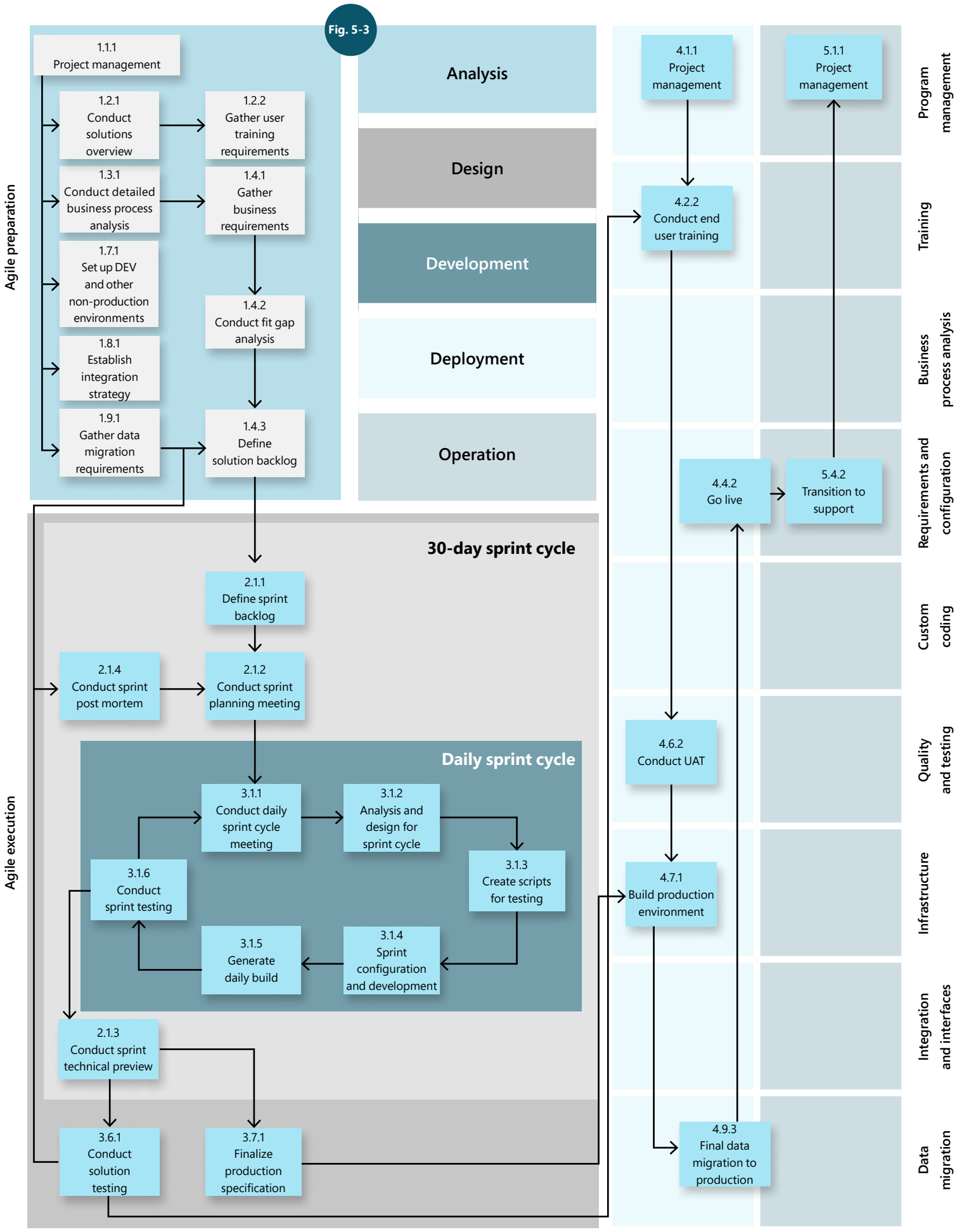


Fig. 5-3
Analysis
Design
Development
Deployment
Operation
Agile preparation
Agile execution
1.1.1 Project management
1.2.1 Conduct solutions overview
1.2.2 Gather user training requirements
1.3.1 Conduct detailed business process analysis
1.4.1 Gather business requirements
1.7.1 Set up DEV and other non-production environments
1.4.2 Conduct fit gap analysis
1.8.1 Establish integration strategy
1.9.1 Gather data migration requirements
1.4.3 Define solution backlog
30-day sprint cycle
2.1.1 Define sprint backlog
2.1.4 Conduct sprint post mortem
2.1.2 Conduct sprint planning meeting
Daily sprint cycle
3.1.1 Conduct daily sprint cycle meeting
3.1.2 Analysis and design for sprint cycle
3.1.3 Create scripts for testing
3.1.4 Sprint configuration and development
3.1.5 Generate daily build
3.1.6 Conduct sprint testing
2.1.3 Conduct sprint technical preview
3.6.1 Conduct solution testing
3.7.1 Finalize production specification
4.1.1 Project management
4.2.2 Conduct end user training
4.4.2 Go live
4.6.2 Conduct UAT
4.7.1 Build production environment
4.9.3 Final data migration to production
5.1.1 Project management
5.4.2 Transition to support
Program management
Training
Business process analysis
Requirements and configuration
Custom coding
Quality and testing
Infrastructure
Integration and interfaces
Data migration

The phases, milestones, and deliverables are clearly defined, and you only move to the next phase when a prior phase is completed.

In the modern cloud world, early stakeholder buy in is critical which requires delivering quicker results, which makes waterfall technique not so suitable.

An example of the Waterfall model as prescribed in Sure Step Methodology is depicted in **Figure 5-4**.

Hybrid

In the current landscape of rapidly changing business needs and changing technology trends, projects commonly adopt a hybrid approach. This is a combination of Waterfall and Agile practices that implementation teams use to meet their project needs. It allows teams to tailor an approach that enables a successful implementation of Dynamics 365 applications. It is also our recommended approach for Dynamics 365 engagements.

Figure 5-5 shows an example of a hybrid approach, based on Dynamics Sure Step 365.

With this approach, we can manage initial activities, like initiation and solution modeling, and final activities like system integration testing, user acceptance testing, and release to production, in a noniterative way. Then the build activities, such as requirement detailing, design, development, testing, are completed with an iterative approach. This helps provide early visibility into the solution and allows the team to take corrective actions early on in the overall cycle. This approach is considered to use the best of both Waterfall and Agile approaches and is a win-win for any size of implementation.

In the hybrid approach, the focus is on a regular cadence of iterative and incremental releases of software features.

We stress for customer stakeholders to participate in the process, especially in the business validation reviews that happen after each sprint

Fig.5-4

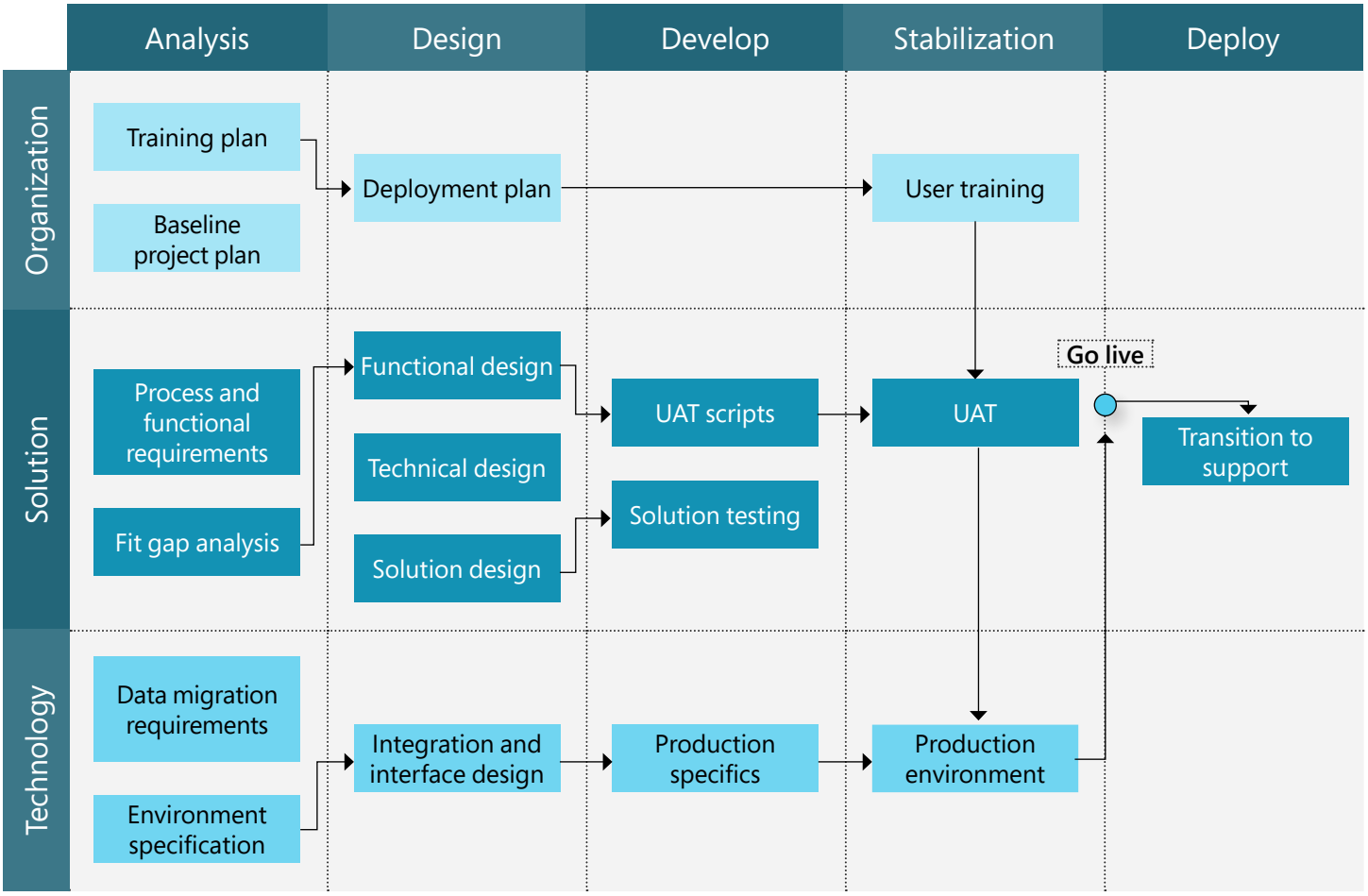
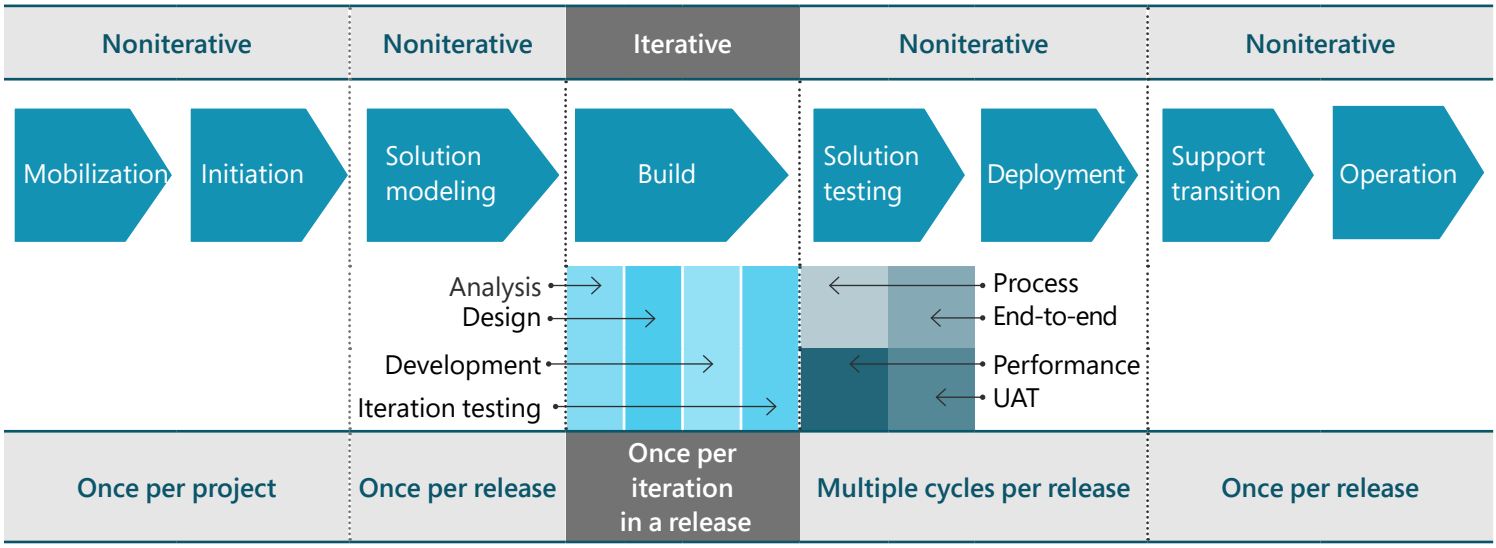

Analysis
Design
Develop
Stabilization
Deploy
Organization
Solution
Technology
Training plan
Baseline project plan
Deployment plan
User training
Process and functional requirements
Fit gap analysis
Functional design
Technical design
Solution design
UAT scripts
Solution testing
UAT
Go live
Transition to support
Data migration requirements
Environment specification
Integration and interface design
Production specifics
Production environment

Fig. 5-5


Noniterative
Noniterative
Iterative
Noniterative
Noniterative
Mobilization
Initiation
Solution modeling
Build
Solution testing
Deployment
Support transition
Operation
Analysis
Design
Development
Iteration testing
Process
End-to-end
Performance
UAT
Once per project
Once per release
Once per iteration in a release
Multiple cycles per release
Once per release

cycle. These reviews are placed at regular intervals during the project schedule, and allow for faster valuation and better stakeholder buy in.

Through these reviews, customer stakeholders get an opportunity to see working software, have discussions with the team, and provide early feedback. The readiness of business processes is validated for production use by an invited business audience by showing the incrementally improving readiness across all the components in the system as working software, like design, data configuration, data migration etc.

The idea is to get early feedback and validation from the business team on the business requirements scope and understanding. This approach allows the project to showcase working solutions faster than other methodologies and achieves a higher rate of implementation success.

Define deployment strategy

A deployment and rollout strategy describes how the company is going to deploy the application to their end users. The strategy chosen is based on business objectives, risk propensity, budget, resources, and time availability, and its complexity varies depending on each implementation's unique scenarios. There are several key factors to consider prior to choosing any of the approaches detailed below.

- What is the MVP (minimum viable product) needed for faster value to customers and then later plan for continued enhancements? Refer to Chapter 4, "Drive app value," to define an MVP strategy.
- Is this a multi-country rollout or single-country rollout?
- Is there a need to consider pilot rollouts prior to wider team rollouts?
- What are the localization requirements, especially in scenarios of multi-org or multi-region rollouts?
- Do we need to phase out a legacy application by a certain key timeline or can it be run in parallel?

- What is the extent of impact on end users' day-to-day productivity?
- What training needs do end users have?
- What is the complexity of integrations during the rollout?

Taking these factors into consideration, you should define a rollout strategy that helps the organization deploy the application in the most effective manner.

The following are some typical deployment strategy models.

Big-bang

In the big-bang approach, as the name suggests, the final finished software solution is deployed to production and all users stop using the old system and start using the new system on the go-live date. The term "big-bang" generally describes the approach of taking a large scope of work live for the entire user base of your organization simultaneously.

The selling point of the big-bang approach is that it concentrates the focus for a shorter rollout period compared to a phased rollout. You trade the cost and disruption of staying in rollout mode for a longer period of time for a longer wait to get value from the solution.

An example of big-bang rollout is when an organization that was using a legacy business application for all of its corporate office finance team, communicates to all end users to stop using the old system and start using the new Dynamics 365 Finance on the go-live date.

The whole implementation can be very large and can encompass multiple countries, departments, or business units. During transition, the new solution is deployed through a number of activities while the old system is turned off so the new system can go live.

The risks of this approach are that the entire project could be rushed, minor but important details could be ignored, and business processes transformed may not be in the wider interest of the organization. In

an aggressive big-bang rollout, the risks are magnified due to the dual combo of a large scope and shortened rollout period.

Change management can also suffer as people are less inclined to use the system that may not be solving the business problems it was meant to solve.

With such large changes happening, there is a potential for issues to arise post go-live and falling back on the older system is very hard, if even possible. It is critical to ensure proper contingency planning for business continuity, and support mechanisms to deal with post go-live issues.

Fig. 5-6

Large project scopes are at much greater risk by using the big-bang approach since the delivery of finished software to the production takes up a longer timeline, there is more to organize during transition. If the final rollout runs into challenges, the wider user community is impacted due to the resulting delays. If Waterfall methodology is used as the implementation, then the end users don't have a feel for the real system when it is finally rolled out. However, if hybrid methodology is used, it is possible to involve end users from the beginning, keeping them updated on what is landing on go-live and keeping any surprises in check.

Pros	Cons
▪ Shorter, condensed deployment times, minimizing organization disruption	▪ May not accommodate for rapidly changing market scenarios and product capabilities, leaving a lack of solution alignment at deployment causing a negative impact for end users
▪ Lower costs from a quick rollout without loss of overall work, but with an added need to account for resource costs to manage the complexity	▪ Daily task delays from users learning to adapt to the new application without any prior experience
▪ Same go-live date for all users	▪ Transitioning back to legacy systems can be costly and difficult, if even possible
▪ Reduced investment in temporary interfaces	▪ Any failures during the rollout have a high impact on maximum users
	▪ Heightened probability of risks emerging from introducing large changes for large groups of users
	▪ Heightened probability of issues coming up in go-live
	▪ No opportunity to learn from early launches to change things
	▪ Costs of ramping up a large rollout team for a short period of time

For large and complex implementations, big-bang projects can require many months or even years to complete. Given the pace of change in most industries it may

not be feasible to wait that long for implementation, especially since the business that was analyzed for requirements could have different needs and priorities a year later, as illustrated in **Figure 5-6**.

Phased rollout

In a phased rollout approach, we are releasing the software in a number of phases or releases. The phases can be planned in several ways and can be based on modules, business priority, business units, or geographies. One approach for a phased rollout can be to release an initial set of capabilities in the first phase and build on that by releasing the rest of the requirements in subsequent phases. At the release of the first phase for a business unit or module, it is expected that the specific business unit's end users stop using the legacy system

Fig. 5-7

Pros	Cons
▪ Higher business value features can be prioritized. The implementation team may also choose to prioritize lower complexity features to adapt to the new functionality.	▪ Longer implementation and deployment timeline due to multiple go live events.
▪ Unlike big-bang, phased rollout releases can bring new use cases gradually, allowing for more buy in from end users as they get used to the new system.	▪ Complexity of data migration from legacy application to the target application increases as it needs to be planned in a staggered approach.
▪ Project teams can optimize later phases by incorporating the learnings from earlier ones.	▪ Organization needs to plan for continuous disruption over longer periods of time, like parallel efforts of supporting deployments while working on future phase implementation.
▪ Risk of disrupting business is less as errors found in the initial phase are limited to specific business areas and users and rest of the business is not affected.	▪ Employee morale may suffer as they face change fatigue. Phased projects need lots of focus and coordination as the team might start losing momentum after first few phases. These projects don't have the benefit of focused intensity as the big-bang approach does. It is important that the change management team keeps a tab on employee morale and plans initiatives to keep their interest alive.
▪ When phases are based on functionality, rather than module, geography, or business unit, it typically results in a faster time to value.	▪ Need to implement temporary scaffolding solutions to manage integrations between new and legacy systems until the new solution is fully rolled out.
▪ For large implementations, it reduces the level of investments and resources needed for ramp up for each deployment relative to a big-bang approach.	▪ Scope creep can occur since project timelines are drawn out. There can be a tendency to incorporate changes and rework, which can affect the work for subsequent phases and their timelines.

and move to the new one. There could still be temporary scaffolding integrations setup to integrate the new system with the legacy systems until all phases are rolled out.

For example, a team can start the implementation with one business unit. Once implementation is complete, they travel to the next business unit. As the team gains experience and learns from past mistakes, the subsequent rollouts become smoother. This leads to less risk and more employee adoption, as illustrated in **Figure 5-7**.

Parallel rollout

In a variant of phased rollout approach, parallel rollout, the legacy system is not discarded but kept alive along with the new system. More than a rollout approach, it is a validation technique allowing users an opportunity to learn, test, and get comfortable with the new system. Typically, for few months both systems are actively used and users are required to key in information in both systems.

- There is much more effort required from the users as they double key information.
- This rollout may be preferred by projects where the business risk is very high and cannot be easily mitigated by testing. At the same time, we need to be mindful of making sure team is not cutting corners on testing and training efforts.
- This rollout is less and less popular due to the extra efforts and costs involved in keeping two systems.

Rollout strategy goes hand in hand with change management strategy as several activities of change management play a key input to successful rollout, as illustrated in **Figure 5-8**.

Fig. 5-8

Pros	Cons
▪ Less risk	▪ High efforts and high costs
▪ Users take their gradually plan to migrate to the new system	▪ Needs for data duplication can get complicated. We recommend having a clear exit plan that is time or workload based, for example, all existing cases closed in old system
	▪ May mask poor training and testing efforts

Define a change management strategy

Adoption and change management planning is a critical aspect of implementation strategy. Change is never easy for any organization and requires thoughtful planning and a structured approach to achieve high adoption and usage. A change management strategy with a clear action plan ensures smooth rollouts and alignment with stakeholder and end user expectations. In the absence of a planned change management approach, many business applications tend to become purely a reporting tool rather than a value driven user centric application.

Bridge the gap between business requirements and results

All too often, projects meet requirements without a true focus on adoption and business outcomes. The gap that exists between technology solutions and business outcomes directly relates to the users who experience a change in their day to day work. Change management focuses on bridging this gap by effectively supporting and equipping the user community to successfully adopt the transition. A change management mindset is required across different functions, including the project development teams, to keep the end user experience as a primary design consideration.

Increase likelihood of project success

PROSCI's correlation data from over 2,000 data points and 10 years shows that initiatives with planned change management are six times more likely to meet objectives than those without a formal change management strategy. The data is abundantly clear. The better we apply change management, the more likely we are to deliver on project objectives.

Take the change out of change

Change is difficult. To the degree that we can, we want to remove the chance or variability associated with change. Project management has

accomplished this by providing direction on sequencing milestones, deliverables, activities, and resources over the lifecycle of an effort. Unless we proactively support and guide people through the changes our projects bring, we leave the likelihood of them embracing change to chance. Change management addresses this by providing employees with the preparation, support, and skills they need to succeed in change.

PROSCI's framework that is followed at Microsoft describes the three sequential steps that are followed in change planning.

Preparing for change The first phase in PROSCI's methodology helps change management and project teams prepare for designing their change management plans. It answers these questions:

- How much change management does this project need?
- Who is impacted by this initiative and in what ways?
- Who are the sponsors we need to be involved to make this initiative successful?

Managing change The second phase focuses on creating plans that integrates with the project plan. These change management plans articulate the steps that you can take to support the users being impacted by the project.

- **Communication plan** Communications are a critical part of the change process. This plan articulates key messages that need to go to various impacted audience. It also accounts for who sends the messages and when, ensuring employees are hearing messages about the change from the people who have credibility with them at the right time. Communication is an understated yet very significant aspect that keeps the various teams connected throughout the entire journey. Some of the key communication aspects to consider are:
    - Providing a sneak peek into what is in it for end users
    - Launch and availability communications
    - Project sponsor vision and direction communication
    - Communication frequency and approach
    - End user feedback incorporation

        - Status reporting, including steering committee reports
    - **Sponsor roadmap** The sponsor roadmap outlines the actions needed from the project's primary sponsor and the coalition of sponsors across the business. In order to help executives be active and visible sponsors of the change it identifies specific areas that require active and visible engagement from the various leadership teams, what communications they should send, and which peers across the coalition they need to align with to support the change.
    - **Training plan** Training is a required part of most changes and is critical to help people build the knowledge and ability they need to work in a new way. The training plan identifies the scope, the intended audience, and the timeframe for when training should be planned for. It is important that the training plan be sequenced in a way that allows for awareness and desire building before they are sent to training. A common aspect that tends to be overlooked is cloud focused training for IT administrators. As organizations implement Dynamics 365 cloud applications, it also may mean a significant change for IT teams, so it is important to plan training that keeps their persona in perspective. They may require training not only on Dynamics 365 administrative activities, but also for a general understanding of cloud concepts.
    - **Coaching plan** The coaching plan outlines how you engage with and equip managers and people leaders to lead the change with their own teams. Managers can play a significant role in aiding the change management efforts, but they need to be engaged as employees themselves first and allowed to work through their own change process. Once that's done, you can give them the information and tools to lead the same change process with their own teams.
    - **Resistance management plan** Resistance to change is expected, so proactively defining the activities to mitigate the areas of concerns should be initiated early on in the project lifecycle. Engaging user champions from the end user community early on to build a user centric solution contributes significantly towards addressing this risk.

Reinforcing change Equally critical but often overlooked, this third phase helps you create specific action plans for ensuring that the change is sustained. In this phase, project and change management teams develop measures and mechanisms to measure how well the

change is taking hold, gauge user adoption, identify alignment with KPIs, and correct gaps.

A successful change management strategy requires continuous executive sponsorship, integration with the project management teams, employee engagement, and frequent and open communication.

Conclusion

When a team defines an implementation strategy, they set clear expectations and guidelines for the team and wider business on how the project goals are going to be attained. This helps define a clear translation of business goals into a structured implementation plan.

To summarize, a well-rounded implementation strategy takes the following into consideration.

- Build a common understanding of vision.
- Identify the methodology based on the specific factors that are driving the implementation.
- Follow the tried and tested approach that is defined by the methodology—do not leave activities out just for convenience.
- Help to take feedback early in the implementation journey by doing a show and tell after each sprint/phase depending on the chosen methodology.
- Keep organizational culture under consideration while choosing a methodology. Organizations that have followed a traditional approach such as Waterfall may need additional change management planning to embark on a modern approach such as Agile or hybrid.
- Consider which industry practices may also have a bearing on methodology. Some industries may have their own tailored approach and that needs to be kept into consideration.
- Formulate a focused approach towards user adoption by creating a change management plan.
- Plan an approach of communicating to various stakeholders, including leadership and end users.
- Continue to improve throughout the process—solicit end-user sentiment and act!

Section
Initiate

6

Guide

Solution architecture design pillars

Introduction

You can't have a solution without first having a vision. When you know the answer you're looking for, only then you can find a solution to get you there.

But it's not always easy to know and articulate what you want, let alone identify the elements that are essential for creating a blueprint of your solution.

This chapter outlines the pillars of a successful solution architecture design that support your vision, and provides you with the building blocks you can use to create a coherent and satisfying outcome. By utilizing the Success by Design framework, we present a structured approach for designing a solution that's enabled by Dynamics 365 apps.

Solution architecture design pillars

Most of us know what "architecture" means in the building industry—it includes the job of the architect, the scope of the work, and what's ultimately delivered. The architect starts with the project requirements

and the budget, and then works on a blueprint of the design. Once the physical labor begins, the foundation is laid, the house is built, and the interior is completed. Many years of maintaining and improving the building may follow. But why and when do you need an architect to design a solution for your organization?

Let's take the simple case of building a new house for a dog. In most cases, this is a single-person job with a general vision of the final product. Building a doghouse is cheap, and the risk of failure is low, so a trial-and-error approach is one option.

But what if you were asked to build something as complex as the Sydney Opera House in Australia? The approach would be completely different: the job can't be completed by a single person, the design needs to be carefully planned, and architects, construction workers, plumbers, electricians, and decorators have to coordinate their workflow. With so many moving parts, there must be well-defined processes, an explicit design, and powerful tools. When all the right pillars are in place, the likelihood of failure is dramatically reduced.

Solution architecture follows a systematic approach by identifying the desired solution and the necessary building blocks, where each builds on the previous block and provides a foundation for the next block. Solution architecture breaks up the work into logical sections, and coordinates those sections, managing and mitigating the risks along the way.

Fig. 6-1

Building blocks of solution architecture design

You start with a vision, which leads to a business strategy where the processes, people, and data are joined (**Figure 6-1**). A solution strategy outlines how those elements can be addressed by using the technology of Dynamics 365 apps. Project and change methodologies, supported by governance and control methods, define the workflows and the delivery.

Solution architecture design incorporates the business vision and its implementation into a blueprint. The first version is created as part of

the pre-sales process and forms the initial high-level understanding of what you plan to build.

Solution architecture enables implementation of a vision, and typically begins after the business strategy has been crafted, but before the solution is built and implemented (**Figure 6-2**). Architecture creates a picture of the finished product and includes the specifications needed to build it.

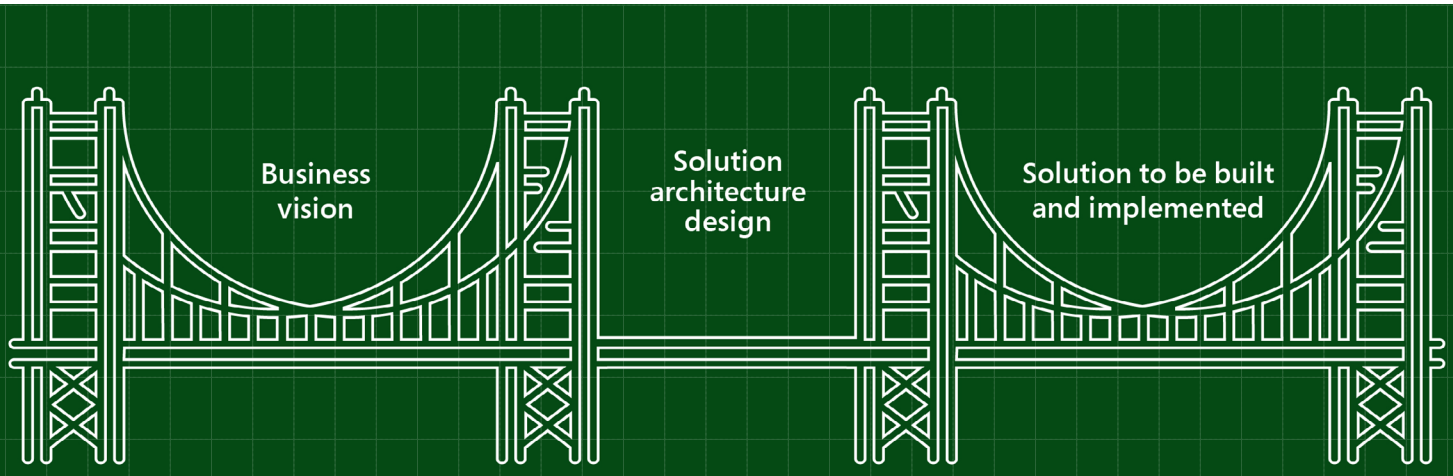
A solution architect provides the necessary quality control by ensuring that key requirements are met, with all relevant interdependencies considered and managed. A solution architect also serves as a translator between the business and IT teams to articulate the impact of technology on the business, and vice versa.

Vision

A vision is the desire to achieve something—to change the present and improve the future.

When an organization decides to go through a transformation, it's usually because one or more people had the ability to predict what was coming and expressed a desire for change, or a change was forced by industry disruption or regulations. Such a change could be in the form of a mission statement or a business case listing its objectives, which might include:

Fig. 6-2



- Finding unified and faster ways of working.
- Earning higher profits.
- Achieving better service or product quality.
- Improving the user experience.
- Empowering users to drive greater value by building apps.

As the vision comes together, the plan for achieving it can start taking shape.

Business strategy

Every vision serves a purpose, as does every organization, and any solution needs to be aligned with this purpose. A business strategy supports your vision by answering fundamental questions, such as:

- **Why** are you making this change, and what are the anticipated benefits? What is the business value sought by the solution? Where do you imagine the organization will be in five, 10, or 20 years?
- **What** business capabilities can your organization offer with the new solution? What business processes can you run? What information and data would you like to record and report on, in line with your organization's services or products?
- **Which** clients, customers, or people inside the organization will be served by the new solution, and who will be affected by it?
- **Would** you like to improve your current line of business or are you open to a new industry?
- **When** do you plan to have the vision materialized? What is the high-level timeline? And do you intend to deliver the solution at once or grow in stages?
- **Where** are the regions—geographically and in terms of business functions—to which the solution will apply? Will you apply it to all or just some of them?
- **Who** is going to plan, design, and deliver the solution? Do you have a preferred partner or do you need to select a vendor?
- **How** will you incorporate technology into your solution? (This is the first step of solution design and the link to your solution strategy, as well as the business case for IT transformation.)

Why should you define your processes?

- It **creates consistency** by allowing for consolidation and unification of processes across locations, teams, and systems.
- It **reduces complexity** by standardizing and rationalizing previously complicated and cumbersome procedures.
- It **eliminates guesswork and potential ambiguity** by streamlining operations and improving communications.
- It **promotes productivity** by eliminating inefficiencies and establishing one workflow for all users.
- It **guarantees quality** by maximizing attention to detail and ensuring work is done in a pre-defined, optimized way each time.
- It **boosts morale** by helping team members take pride in mastering the process, refining their skills, and avoiding mistakes and missed deadlines.
- It **encourages continuous improvement** by giving team members who follow the processes a chance to give input on how to improve them.
- It **increases user acceptance** by providing a higher-quality experience at each launch.

Processes

Process architecture is a commonly understood, shared view of all business processes that an organization uses to deliver a product or service. It represents how an organization operates, in a structured order. Complete, accurate, and well-organized process architecture is the first and most important pillar for a sound solution design. It confirms end-to-end business understanding, provides a structure for understanding the scope, and serves as the basis for testing and training. It also forms a map for requirements gathering. The processes are constructed at different levels to reflect specific areas, functions, locations, and teams. Process architecture is often referred to as a target operating model, a process library, or a catalog.

Any transformation starts with defining your processes, which is also a good time to revisit and improve them.

Dependent activities

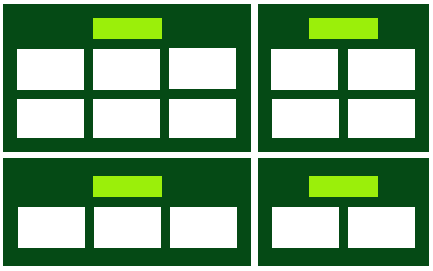
While we're not going to take a deep dive into the business process management life cycle and capability maturity, following the Success by Design framework can help you identify important elements of your solution design.

Process architecture includes multiple dependent activities:

- **Scope management** Defining the scope of the solution is the first step of the design. Ideally, you have a baseline process taxonomy with at least three levels of the process architecture. Start by mapping requirements against it and marking which processes are in scope. You can add to and take away processes from your initial process library.
- **Business analysis** Initially, it's likely that business users don't know or understand the application, and the tech partner doesn't understand the business, so a business analysis helps to join them together. A business analyst gathers requirements and links them to processes; conversely, a good processes structure drives the requirements gathering by asking all necessary questions.

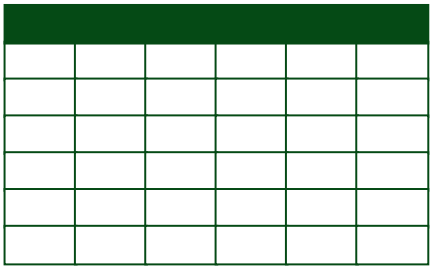
Key deliverables related to processes

Process architecture map



This is a visual representation of your business processes.

Process catalog/ inventory and taxonomy

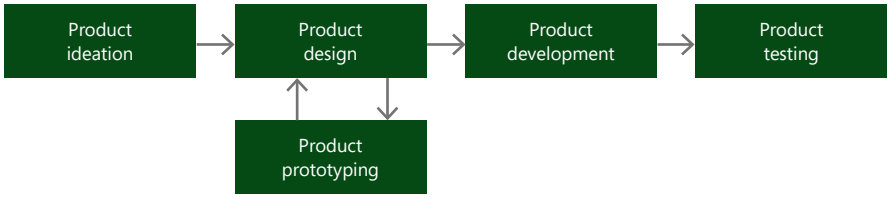


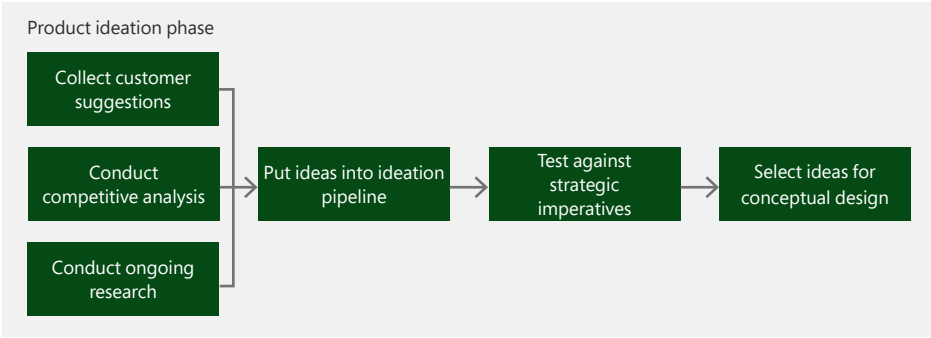
This is a sequenced list of the processes, usually uploaded to Microsoft Dynamics Lifecycle Services (LCS) and synchronized with Azure DevOps.

Linear and cross-functional process flows

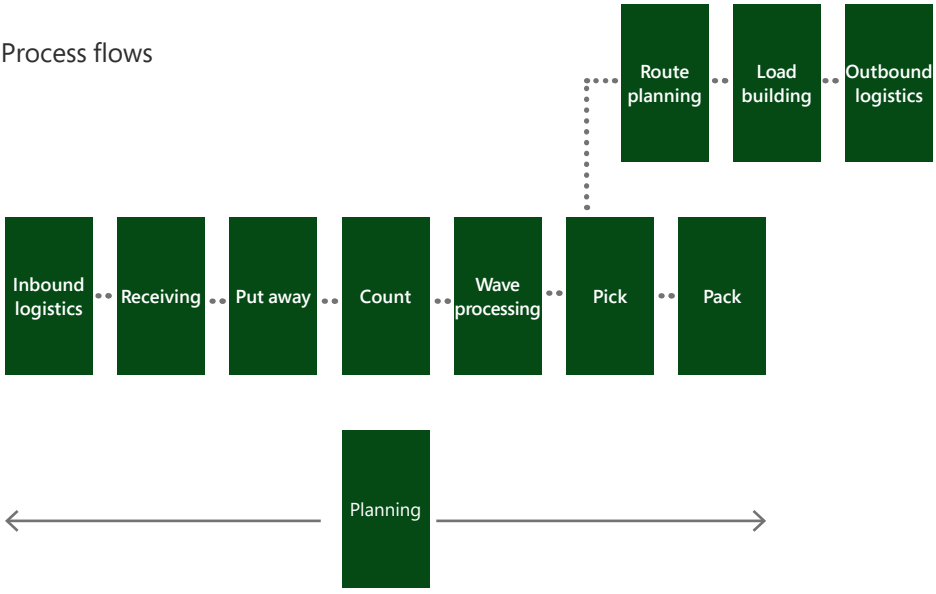
These show the input, output, and activities at process and sub-process levels to help you figure out how a process must change to meet requirements; you can also add details such as assumptions, metrics, timelines, and costs.

Linear flow





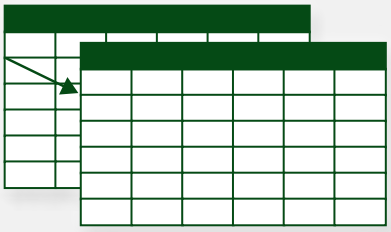
Process flows



- **Requirements management** Every functional and nonfunctional requirement needs to be linked to at least one process. If an applicable process doesn't exist, you must slot it into the relevant section of the process architecture. DevOps is a good tool to use for this success measure, as shown in **Figure 6-3**.

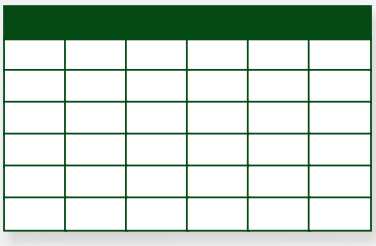
Fig. 6-3

Requirements traceability matrix (RTM)



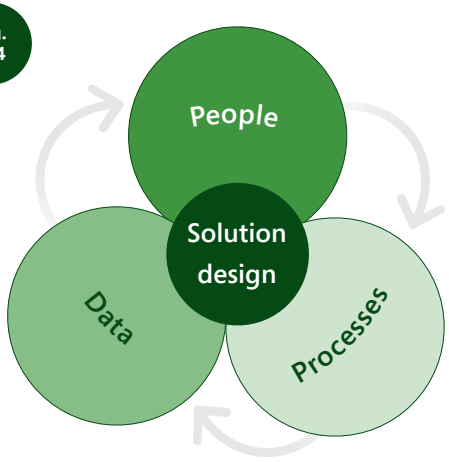
This links requirements throughout the validation process, and is a key deliverable for requirements management.

Fit gap assessment



This is where you mark which requirement and respective process can be addressed by the system, and which ones require customization.

Fig. 6-4



- **Solution design** Once you have a better end-to-end business understanding, it's time to translate it into your system processes. With the Dynamics 365 solution design, a Microsoft partner works with the process leads to develop the solution, and it can be helpful to create process flows that show how the processes can be run in the system.
  - As part of the process, data goes in as input exchanged between people and applications, and data goes out as output in the form of documents, analysis, and reports (**Figure 6-4)**.
- **Test management** Once the solution is designed and developed, the process architecture, along with the RTM, establishes a baseline for testing. When you test every process, you ensure that every requirement is addressed and the solution is appropriate. For more information, refer to Chapter 14, "Testing strategy."
- **Training** Process architecture also defines and drives your training content. You can use your process flows and guides as a first draft for your training materials. Learn more in Chapter 7, "Process-focused solution," and Chapter 2, "Success by Design overview."

Key deliverables related to people

- Organizational structure
- Cross-functional process flows and maps
- List of business users or personas
- Dynamics 365 security roles list
- Mapping between personas and security roles

People

Even though not every process or activity is performed by a person, there's always a step or output that involves people, whether internal (employees and contractors) or external (customers and vendors). This is the key reason why and how the solution design is influenced by the people pillar. People shape the solution design in many ways, including:

- Geographical location
- Time zones
- Languages
- Customs
- Internal and external hierarchies
- Organizational structure

This pillar of solution design is usually represented by the organizational architecture, which visually intersects with process architecture in process maps, and includes:

- Geographical structure
- Organizational groupings
- Line of business
- Reporting lines
- Segregation of duties

Data

The third pillar of solution design is data.

A system with inaccurate, misleading, or partial data will lead to a failed implementation, so you must understand your data and how it fits into the overall solution. With the right data, you can identify actionable and analytical information that improves business intelligence. Think about:

- **Master data** This is static information, such as details about customers and products, that you use and exchange in your business activities.
- **Documents** This refers to documentation, such as sales orders and customer invoices, that you create as part of your business process.

Refer to Chapter 10, "Data management," to gain a broader understanding of the various data functions you'll need before starting a project.

Key deliverables related to data

- Data governance strategy
- Data architecture
- Data migration and integration strategy
- Data quality strategy

- **Reports** This is organized or filtered information, such as a trial balance or aged debt, serving as input or output for your business activities.
- **Transactions** This refers to recorded business information, such as customer payments and expense reports, that you create, store, and report as part of your business process.

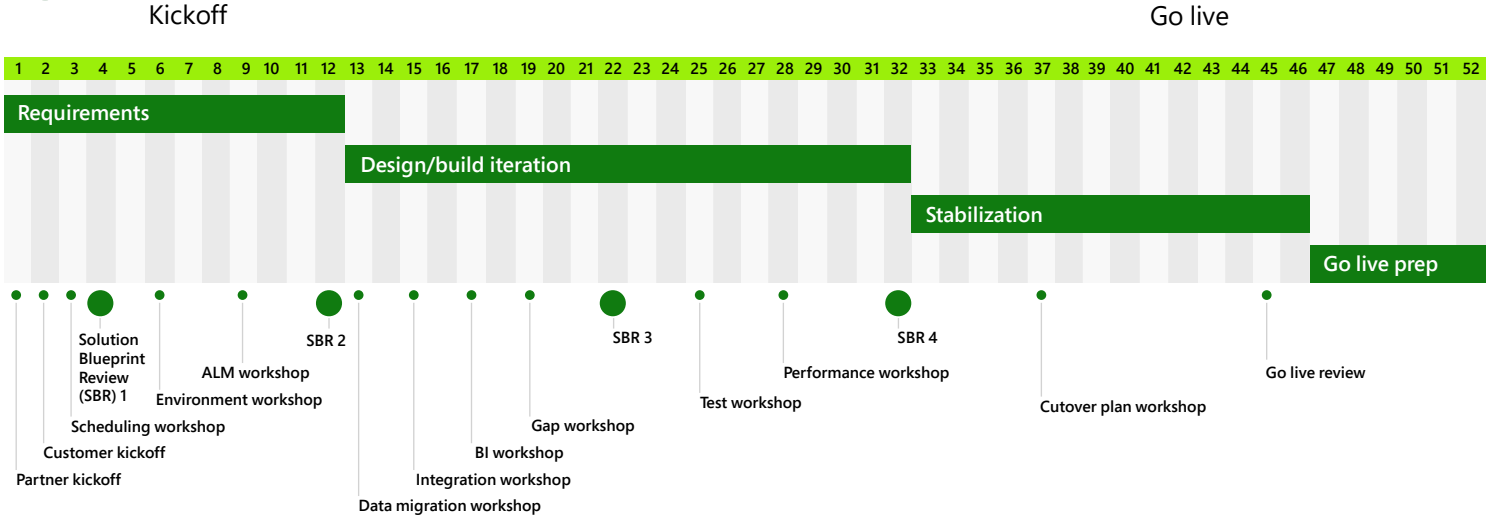
At a minimum, you must start with an understanding of how your data is currently utilized and how it supports your future vision. Planning your data strategy upfront also allows for better scalability.

The world is evolving from silos of data to connected enterprise data that enables the digital feedback loop. That is why a data strategy that applies AI and analytics to make the data actionable is so critical to the overall success of your design.

Solution strategy

Your solution strategy is a consolidated view and approach that defines your overall solution. A solution blueprint is a living document with several review points (**Figure 6-5**) during a project's lifespan to help you identify and take necessary actions, mitigate risks, and resolve issues as they arise. In the Success by Design framework, the blueprint is considered essential for a project's success, and provides a view of the overall solution architecture and dependent technologies.

Fig. 6-5



For more information and a list of activities in each Success by Design phase, refer to Chapter 2, "Success by Design overview."

The Solution Blueprint Review workshop is a mandatory part of the solution design experience, and helps you keep track of your solution design's progress to ensure that your vision and objectives are still viable. It also allows solution architects and the implementation team to review and gain an understanding of your:

- Program strategy
- Test strategy
- Business process strategy
- Application strategy
- Data strategy
- Integration strategy
- Intelligence strategy
- Security strategy
- Application lifecycle management (ALM) strategy
- Environment and capacity strategy

Capturing these details helps you understand the project, and validates your solution design via the complete and well-organized processes, data, and people enabled by Dynamics 365.

Technology

While technology does not drive all the processes, it provides the backbone of products and services required to fulfill your businesses strategy. Your processes dictate which technology to use and when, and that technology will bring your digital transformation to life. For example, the Dynamics 365 Sales app provides value-added details about your organization's lead-to-opportunity pipeline. Other examples of technology that's under the hood to support users include:

- Infrastructure as a service (IaaS)
- Software as a service (SaaS)
- Integration tools
- Business intelligence tools
- Azure AI
- Azure Machine Learning
- Data connectors
- Customer portals
- Mobility solutions

The technology and system selection are usually performed during the sales process as part of the first high-level fit gap assessment.

Finding the right balance

It is easy to say that technology can be used to solve any business process scenario, but technology is simply what turns the gears for your processes, people, and data. It's important to include all stakeholders in the design decisions, because processes don't always align with the technology. This is what we call the "gap" in a requirement, and it's where you might decide to configure or customize an out-of-the-box solution to meet the needs of the process. Think through the different options to find the right balance for your solution, but be sure to challenge the business requirements before customizing the system. You might be surprised to find out that what seemed like a "must-have" customization is more of a "nice-to-have" feature.

Methodologies

A methodology is a collection of methods used to achieve predictable outcomes. Good methodology also demonstrates why things need to be done in a particular order and fashion. The Success by Design framework is a higher-level abstraction through which a range of concepts, models, techniques, and methodologies can be clarified. It can bend around any methodology, including The Open Group Architecture Framework (TOGAF), the Microsoft Solutions Framework (MSF), and the Zachman Framework.

To achieve your solution, the workflows and delivery need to be planned, organized, communicated by various methods, including:

- Project management
- Change management
- Governance and control

For more on this topic, refer to Chapter 15, "Extend your solution."

For more information about the Success by Design framework, refer to Chapter 2, "Success by Design overview."

Project management

There are multiple project management methodologies and approaches, and methodology for enterprise resource planning (ERP) is a topic that

For more detailed information about project management approaches, review Chapter 5, "Implementation strategy."

"When you have bad governance, resources are destroyed."

– Wangarĩ Muta Maathai, Nobel Peace Prize winner

can generate more heat than light. Partners often rely on their own branded methodology to reassure customers about their governance. Project management approaches can be grouped into three core categories:

- Waterfall
- Agile
- Hybrid

As we move increasingly toward a low-code world, there are faster deployments and fewer customizations, and your approach should match the changing world and your customers' expectations.

Change management

Project management (the "how") and change management (the "who") are both tools that support your project's benefits realization. Without a change management plan in place, your organization's objectives are at risk. To drive adoption with your end users and accelerate value realization, apply a vigorous change management approach, which is most effective when launched at the beginning of a project and integrated into your project activities.

The benefits of change management:

- It focuses on the people side of organizational change.
- It seeks individual and organizational perspectives.
- It requires action and involvement by leaders and managers throughout the organization.

Governance and control

The term "governance" often brings to mind just project management or possibly even stakeholder management, but such a limited view of governance may cause an organization to ignore core elements required for success (**Figure 6-6**).

Every project component contains an element of uncertainty and is based upon assumptions made before details are known or fully understood. So, if we know this, why would we expect a project

Read more in Chapter 5, "Implementation strategy," and Chapter 8, "Project governance."

manager to be solely responsible for project governance? The project manager is of course accountable for the project outcome, but 360-degree governance requires everyone—especially the architects and leads—to play a role.

A 360-degree governance method directly affects how well the project team performs and the ultimate result, and is executed on multiple levels:

- A governance board or steering committee has final responsibility for meeting the project goals. It typically is comprised of representatives or stakeholders from each of the major participants.
- A project manager leads the team and administers project elements that include alignment, risk identification and mitigation, escalation, and cross-Microsoft collaboration.
- All project leaders, including the project manager, solution architect, and technical architect, closely communicate and align toward the common goal of solution deployment.

Fig. 6-6

Governance domains

Project management	Stakeholder engagement	Solution management	Risk and issues management	Change control	Organizational change and communication
▪ Weekly status reports ▪ Project plan and schedule ▪ Resource management ▪ Financial management ▪ Risk management ▪ Change management	▪ Executive steering committee ▪ Communication and promotion ▪ Escalation of unresolved issues ▪ Solution decision making and collaboration ▪ Business alignment and tradeoffs	▪ Oversight of functional and nonfunctional attributes ▪ Data, integration, infrastructure, and performance ▪ Customizations and process adaptation ▪ Security and compliance	▪ Objective risk assessment and mitigation planning ▪ Issue identification, ownership, and resolution ▪ Prioritization and severity assessment ▪ Risk response management	▪ Identification and prioritization of requirements and solution attributes ▪ Tracking and reporting via ADO ▪ Organization and process changes	▪ Intentional and customized communication to stakeholders ▪ Periodic effectiveness check ▪ Solution adoption and operational transformation management
▪ Project status meetings and reports ▪ ADO (project tools) ▪ Governance framework	▪ Project change management ▪ Solution management ▪ Organizational change management	▪ Architecture board ▪ Governance framework	▪ Project status meetings and report ▪ ADO (project tools)	▪ Statement of work ▪ Project artifacts ▪ Project status report ▪ Governance framework	▪ Communication plan ▪ Stakeholder engagement ▪ Organizational change management

- The technical architect leads and orchestrates the technical architecture across all application components to ensure optimal performance.
- The solution architect guides users toward the new vision and transformation.
- The change manager ensures all internal and external communications are in place.

Conclusion

As we've explained in this chapter, solution architecture follows a systematic approach that identifies your desired solution and the building blocks needed to construct it. Solution architecture design takes your business vision and breaks it into logical sections that become a blueprint for building your solution. Here is a list of the key steps for solution architecture design:

- Define your vision.
- Create a business strategy.
- Outline your business processes.
- Determine how people are connected in and around your organization.
- Know your data.
- Develop a solution strategy using technology and tools.
- Apply project management, change management, and governance and control methodologies.

7

Guide

Process-focused solution

Introduction

Fast changing technologies and markets demand a rapid rhythm of delivering products and services.

Business applications improve inefficient processes by automating, optimizing, and standardizing. The result: A more scalable business solution.

An approach based on business process mapping and management in an implementation project is fundamental if organizations are to achieve their goals. When implementations use this approach, there is more consistency in project phases, communication flows better, and processes run better.

This chapter introduces the importance of a process-focused approach when implementing Microsoft Dynamics 365 Business Applications. The goal is to emphasize the benefits of employing business processes as the primary framework for a project implementation cycle.

In this chapter we cover the following topics related to a process-focused solution:

- Start your implementation project with business processes
- Opportunity for optimization
- Defining the scope of the implementation
- Defining your requirements
- Fit to standard and fit gap analysis
- Implementation lifecycle connected to processes
- A solution that helps you to operate your business

Start with business processes

Business processes are the main drivers to start defining the solution implemented in your project.

The business language

Every organization has an internal language they use to describe daily operations. This language is immersed in business processes and is often largely based on terminology common to the industry. When implementing business applications like Dynamics 365, this business language is the best, most familiar way for the business to think about their needs and the tools and technologies that they use for work. This business language is not simply terminology. It includes the processes used to conduct day-to-day transactions and activities. Curating and utilizing these processes in the language of the business is important to keep sight of the basic truths of business requirements. It also reduces the risk of the implementation team getting lost in the jargon and nomenclature of the technical tasks involved in the implementation.

When the project starts, putting the business process view of the organization at the core pays dividends.

Start with a business process future vision

Creating a vision of how a business is transformed through technology makes it is easier to focus on the activities needed to achieve that goal. This vision needs to include the understanding of how the business currently works and the state that it wants to reach after.

When discussions about the project and business processes begin with third parties, it's helpful for those who may not be familiar with the organization's business to begin learning this language.

Successful implementations start with a clear understanding of an organization's business model. That includes understanding how the organization creates value through its products and services, as well as its relationships with customers and suppliers. There are multiple business processes strategically aligned to make that business model work. Chapter 4, "Drive app value," describes how to define a business model and the importance of connecting the business model to business processes, and the business processes provide the baseline to draw the roadmap of your solution for the digital transformation.

The processes derived from the business model provide insight on what it takes to reach the desired future. Business processes define the functional scope of the solution being implemented as illustrated in

Fig. 7-1

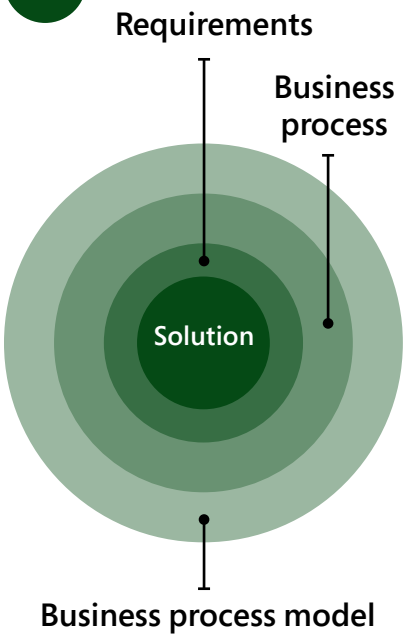


Figure 7-1. Any set of business processes being implemented needs definitions before you can start working on requirements and other aspects of the solution.

When defining your business model, assess the state of your processes. Some organizations may want to use new technology to optimize existing processes through automation. Others are moving to e-commerce or outsourcing parts of the operation, which requires new business processes. Still others might be moving to lean manufacturing.

Many organizations also seek to consolidate operations and introduce standard processes across different parts of the business. Understanding the state of the processes allows you to understand how to reach your goals. The effort required to implement an existing process using new technology is very different than what is required to define new processes or gain consensus from stakeholders.

New business processes won't always be the same across a business. One division, for example, might decide to ship inventory directly to customers rather than to a warehouse, while others continue to use a warehouse. On the other hand, it may be that the model for employee purchasing is the same, so a shared product catalog and ordering process is implemented. In any case, business model analysis, process reengineering, and standardization strategies should be considered part of the project definition. That firmly anchors the processes to the business strategy and also helps generate process-based project goals.

Opportunity for optimization

It is easier to find good opportunities to optimize your process using business process mapping.

Mapping processes

Processes are the heart of every business. They describe how the business operates. The fact that your products or services reach your customer

safely and on time is due to a series of tasks executed by different roles and different business units. Business mapping represents these steps.

Baseline business processes are called “as-is” processes. Depending on the organization, these may be legacy processes or they may be new processes engineered as part of a new solution or business model.

Regardless of where an as-is process set comes from, it is critical that they are described at the appropriate level. As-is processes that contain details about how current implementations are in legacy systems introduce a risk of reimplementing the legacy system. Keeping as-is processes at the more abstract level helps drive the scenarios in scope and enables the team to focus on the desired business outcomes.

It’s worth reminding ourselves that the objective of as-is business process mapping is not to define step-by-step how the solution is designed in the new solution.

The goal of mapping the as-is business process is to:

- Provide structure/skeleton for the definition of the scope of the project.
- Express the project aims in the natural business language of processes.
- Enable a natural way to define the main end-to-end business scenarios (process variants).
- Provide the best language to identify and highlight areas for process innovation and improvements, as well as risks and constraints.
- Become the basis for mapping to the equivalent process design within the new system.
- Create a workable definition of the project that the business can directly interact with without going to the level of individually written requirements.

The value of having business processes mapped early is that it allows everyone involved to understand how the processes operate by using them as a common business language. This helps both third parties and the organization agree on processes across multiple departments and organization layers. It is common for senior managers to be surprised by how processes actually work on the ground.

It is essential that the right team is involved in mapping business processes to the new technology. This team should be guided by the business stakeholder who owns the process in the business operation. This individual is charged with making the right decisions about structure and investment in the process to achieve their goals. The stakeholder is supported by one or more subject matter experts who are familiar with how the business process operates in the real world. These experts can provide depth around the various scenarios under which a process is executed. Their knowledge, coupled with a mindset open to new ways of working, helps drive the right conversation on process mapping.

Here's an example. Let's say a company is looking to move to a B2B business model via e-commerce and seeks to deploy their telephone salespeople strictly to consumers. There may not be existing patterns within the business to replicate, so the new processes should be based on expected outcomes rather copying legacy steps. This mindset should be applied to all processes, as the key deliverables should be the expected "business scenarios" and the desired business outcomes.

When reviewing new process diagrams, the team should consider the following:

- Do the business SMEs recognize their business in the processes?
- Do they provide adequate coverage of the business activities in scope?
- Do they show the key interconnections between the processes?
- Do they show meaningful end-to-end business flows?
- Will they represent a good level at which the project team can communicate with business stakeholders on risks, progress, decisions, and actions?
- Do they illustrate processes sufficiently to identify opportunities for improvement?
- Are they sufficiently well-defined to be able to be used in the next stage where the processes can be applied to drive the design in the Dynamics 365 system such that they can reflect the improvements and the project goals and vision?

These process diagrams help define the baseline for the organization's current business and are a good place to start when plotting optimization

objectives. They should help to identify the processes or parts of processes that are not efficient or well understood.

The key to getting the processes mapped well and quickly is to ensure the following.

- The right people are in the room to direct the mapping.
- Any mapping software or tools that help facilitate the rapid definition of the process and do not slow things down.
- The level to which the process is described is directly proportional to its importance to the business. For example, once the end-to-end scope of a widely used and very standard finance process is defined to a certain level it should not be further broken down to step-by-step processes.
- The process mapping effort is interactive, not a series of documents that go through long handoffs and approvals.
- The tools for the mapping can be anything from sticky notes, to Microsoft Visio, to specialized process drawing software. The process maps are visual and not hidden in wordy Microsoft Excel lists.

This is a vital step that needs to be executed early. Plan so that processes are defined as fast as possible using the simplest tools in the workshop, like sticky notes, pens, and whiteboards. The drawings and comments can then be transferred offline to more sophisticated applications.

Remember that doing this mapping at the start of the project implementation is important. It could become a fundamental part of helping the analysis of the opportunities for process improvement and creating the right baseline for the project.

Modeling your solution for the future

Digital transformation is fast paced. Processes that work today may not remain competitive tomorrow.

Digital transformation is fast paced. Processes that work today may not remain competitive tomorrow. Technologies like the cloud, artificial intelligence, machine learning, robotic process automation, virtual reality, the Internet of Things, big data, virtual conference rooms, and many others create possibilities that did not exist a few years ago.

When reviewing your business processes, it is important to keep in

For example, the Microsoft HoloLens is being introduced to some manufacturing processes with Dynamics 365 applications. Using the HoloLens, the user can understand how to operate a machine. Or if a machine is broken, and support is on the other side of the world, virtual reality enables collaboration between the user and support to repair the machine.

Or if you have a workflow with approvals that require a signature, you could add eSignature software to automate the process and reduce the time for approvals.

mind how innovative technologies can improve process outcomes and collaboration. Similarly, there may be multiple areas in your processes which would benefit from modern options now available. Technology is a key differentiator. Remember, chances are that your competitors may well be considering these options and looking to incorporate them.

Standardize your features with a business processes approach

Earlier, we discussed how a business model analysis can introduce opportunities for process reengineering and standardization.

Assuming there is a good plan for process standardization grounded in the business strategy, this standardization can be aided with a new business application system.

Of course, standardization across business units assumes that the business models are sufficiently similar and there are no local market constraints or competitive reasons for there to be differences.

Various benefits can come from standardized processes assuming the business models are sufficiently similar between the business units so that the other barriers to standardization are low.

- Enables reduction in cost by reducing the variations of business processes that need to be implemented, trained, and maintained.
- Facilitates comparative analysis across multiple groups of users that are executing the same process.
- Facilitates mobility of resources within the organization through standard processes to deal with changes in demand.
- Promotes the adoption of best practices across the organizations. Optimization for anyone improves outcomes for everyone.

Standardization also helps with rolling out the implementation to other business units as it reduces the time and effort compared to building a process from scratch. After the first implementation, there is a proven case of it working within the business. The next business unit to be implemented can discuss the pros and cons of the standard process with business peers.

Similarly, it is important to confirm that the constraints and implications are identified and well understood prior to starting design work.

- Standardization of a process can bring benefits at a group level, but may be seen as additional work by a local business unit.
- Standardization can require significant change management.
- Is the standardization being defined at an industry level and then implemented in the system? This can create a conflict if standard processes are not a natural fit with Dynamics.
- Will the process be designed in Dynamics 365? And will these processes become the new standard?
- Is the entire end-to-end process to be standardized or will there be areas and functions for local variations? If so, does that align with the standard capabilities in Dynamics, or could it need customizations?
- Does the business case for standardization remain sufficiently strong after all the constraints and implications are balanced against the benefits?

In multiple, phased rollouts, process standardization often takes the shape of a "core template" with the expectation of rolling out this "template" to subsequent business units.

Creating process maps is essential to identifying the opportunities and to realizing the potential benefits outlined previously. The use of the common business language of processes is the vital mechanism that brings all of these aspects into focus.

Defining the scope of the implementation

A business application implementation is essentially the delivery of a new capability for end-to-end business transactions using the application. Processes are the foundation for the definition of the solution scope. Processes embody many of the properties that make for a good scope definition.

- Business processes are well understood by the customer as they

are expressed in the languages they know best.
- Processes are richer in context than a requirements list.
- Processes show the connections between different business units, roles, functions, task-level steps, and systems.
- They are naturally hierarchical and can be expressed in different levels of detail, depending on their intended purpose.
- Processes are the connectors that show the data and people feedback loop.

Process mappings can be collected in a process catalog. This catalog helps turn the visual process diagram into data. The business processes in the catalog can then be numbered, labelled, and uniquely referenced. Process hierarchy can be managed as part of the data structures. Typically, these can then be managed in Azure DevOps (or similar tools) where they can then be assigned to various tasks such as configure, build, design, or test and to different project roles.

Business processes are usually interwoven with other business processes. These interconnected processes can span systems, and the connection can be a simple endpoint or an entire subprocess performed in the external system. If the latter, it should be considered for your solution scope. Where the process is in the external system is directly relevant to the end-to-end process, include the full process in the process mapping. This not only helps in understanding the process, but also with making better decisions on customizations and selecting ISVs, which are also stated in your solution scope.

Using process mapping as part of the definition of the scope also helps when working with implementation partners. The process flows helps express the ultimate outcomes for the project and having them reflected the statement of work can help with a more comprehensive definition of the project that results in fewer change orders.

Keep in mind that business processes are always subject to change. As you move forward you are enriching the business process flows, and possibly adjusting the solution scope. That is exactly the flexibility that a business process flowchart offers.

Fit to standard and fit gap analysis

In many projects, inertia drives the old ways of working. There are several dangers in assuming that what works now also works just as well in a new Dynamics system. Recreating processes based on legacy systems (often very old systems) can lead to the following:

- Unnecessary and excessive customizations and costs related to design, coding, testing, training, documentation, etc.
- Stagnation in the business as the natural opportunity for innovation and improvements is lost
- Creating unique, siloed, inefficient processes when Dynamics 365 could provide more standard and recommended ways of working based on the learnings of thousands of customer organizations
- Damaging the natural usability engineered into Dynamics 365 by forcing processes designed for legacy software into Dynamics
- Creating processes within Dynamics that were not designed to be used in that way and therefore reducing the ability to use other related functions because the processes are no longer a natural flow or fit for the purpose
- Reducing the ability to immediately use the documentation, training, sample code, and sample data available for the Dynamics system
- Reducing the ability to leverage market and industry knowledge, insights, and experience in Dynamics 365 for the areas of interest for the business
- Reducing the ability to directly apply tools and apps in the marketplace
- Reduced capacity and resources available to apply customizations that can provide significant business value as the available budget and resources get used on customizations that do not add the same value
- Creating barriers for end-user developers (citizen developers) to create rapid, low-cost application using the Power Platform by creating a more complex, customized data model/process model

When it comes to understanding exactly what is going to be implemented in your future solution, it is necessary to perform a deep analysis about

Processes are the foundation for the definition of the solution scope.

the current processes and what needs improvement. To determine what functionalities of your business applications to keep, and what others could be built or purchased from external vendors, we suggest undergoing the following analysis. We suggest this because it helps to have a standard solution, with minimal customizations, which minimizes costs and therefore maximizes value to the business.

Adapting to the standards of new system

It is recommended to start with a fit-to-standard approach for every project. As part of the definition of the scope there should be a business process catalog created in the early stages of the project. This can now be used as the set of processes that can be configured and enacted in Dynamics 365. Of course, initially, not all the configuration is completed perfectly. However, starting with the core business processes and iterating until there is a good, high-level understanding of how the processes could be implemented within the system helps create the solution blueprint. Putting the fit-to-standard approach at the front of the analysis helps to set the right culture of "adopt wherever possible, adapt only where justified."

In most projects this analysis is not starting from a blank piece of paper; the implementation partner has conducted some part of this analysis as part of providing estimates on cost and time and a solution overview. It is recommended that the customer project team get deeply involved in this process to confirm that the key business processes were correctly interpreted and to start gaining knowledge of the standard processes embedded within Dynamics 365.

The continuation of the fit-to-standard leads to the definition of requirements, fits, and gaps, but this is from the perspective of leveraging the processes enabled within the system first rather than trying to recreate the legacy system in Dynamics 365.

The advantages of starting with this approach using the process catalog are as follows.

- It promotes the reduction of customizations by supporting the delivery of the underlying business needs through configuration rather

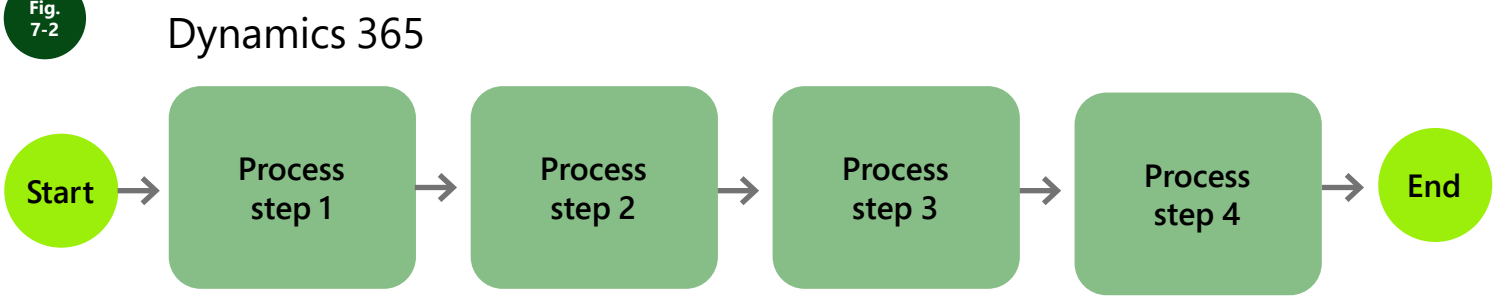
than going directly into fit gap analysis with detailed requirements that may be influenced by the existing or legacy system.

- It helps reduce the risk of missed requirements as the evaluation of the fit with the new system is based on the richer and broader context of business processes. As these business processes are the natural language of business users, their evaluation is more comprehensive, meaningful, and effective compared to working with a list of requirements.
- The process catalog can direct the fit-to-standard assessment by working iteratively through the processes, starting with the higher-level core processes, and then working with the more detailed sub processes and variants. This also helps the business users more clearly see how well their underlying business requirements are being met within the system.
- The project is more likely to adopt modern recommended standard processes embedded in the system.
- It creates higher-quality solutions as the processes are tested by Microsoft and are more likely to be market-tested by others, whereas custom developments and variants, especially complex ones based on the legacy system, will need to be specifically validated by the customer.
- The standard solution allows for more agility in adopting related technologies and by keeping to standards where possible, make it easier to add the real value-add custom extensions.
- It enables a faster delivery of the new solution; there is no need to wait longer for a custom solution to be developed.
- Standard processes are more easily supported by internal and external support teams, including Microsoft Dynamics Support.

The benefits of staying within the standard product wherever possible are clear. The implementations following this approach, often called vanilla implementations, adopt the Dynamics 365 system with its standard configuration as shown in **Figure 7-2**.

Some businesses may have specialized business processes or an innovative idea to improve their competitiveness in the market. In such cases, a process-centric description of the system allows these improvements to be adopted with more transparency.

Fig. 7-2



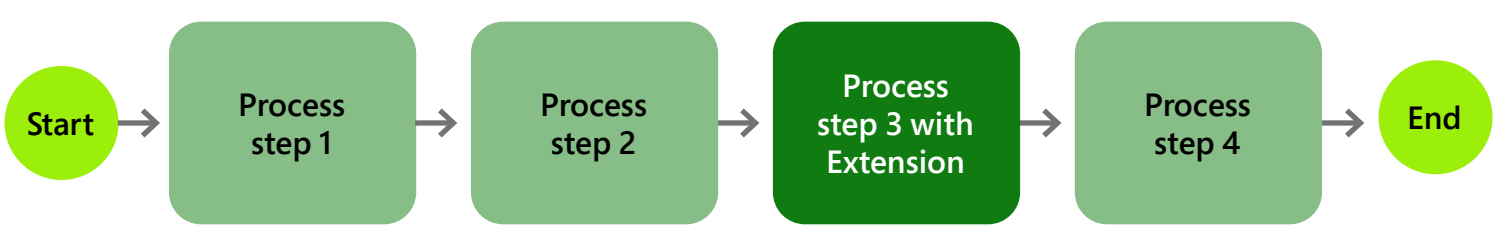
Gap analysis

As discussed in the previous section, adopting a process-centric solution within Dynamics 365 has clear benefits. However, there may be specialized functions that are not part of the standard solution as shown in **Figure 7-3**. That is identified with the fit gap analysis. After having the configurations set, you can look for gaps in the processes and make a decision whether to customize.

Extending a solution can vary in terms of time, cost, and complexity. There are many things to consider when extending. But it offers the benefit of a solution tailored to business needs. See Chapter 15, "Extend your solution," for details on extending a solution. Here we look at the implications of taking a process-centric approach.

- Some extensions can bring innovation into business processes. They can add business value and provide good, long-term benefits.
- It is important to consider potential customizations in light of the rapid pace of innovation in Dynamics 365 SaaS applications. New features are released regularly and keeping an eye on the Dynamics release plans avoids creating extensions that are quickly redundant.
- When evaluating a potential custom extension, using the process maps can help determine the impact, not just on the process being directly changed but also the impact on connected processes.
- When looking at potential solutions for requirements that cannot be met directly in the standard system, consider the various different methods available:

Fig. 7-3



- For simpler designs, consider citizen developer-led applications using the Power Platform, which can generate large value very rapidly and at a low cost.
- For more complex or global solutions, some combination of citizen and professional development through the Power Platform or other technologies may be a more effective and productive route. Using the process catalog can help provide an excellent backdrop to communicate and collaborate between the project team, the citizen developer, and the professional developer.

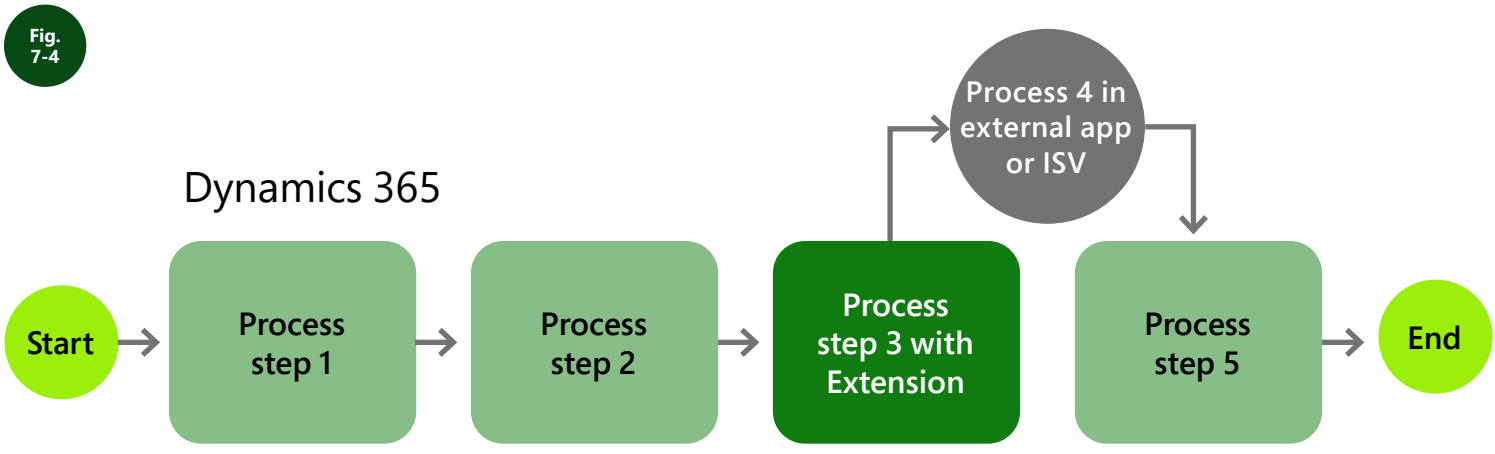
Third-party solutions

An alternative to extending is to buy an existing solution from a third-party vendor, also known as an independent software vendor (ISV). This option is more common when there is a significant gap and developing a functionality in-house can be complex and costly. Sometimes you don't have enough resources, budget, or expertise to develop such a significant solution.

ISV solutions tend to focus on more specialized process areas, normally covering some end-to-end industry-specific process. In addition, ISVs provide specific functionality to cover gaps, as illustrated in **Figure 7-4**. Successful ISVs have expertise and experience in the vertical industry processes they cover and can therefore add value to your business processes.

Dynamics 365 has been designed to meet standard business processes. It enables projects to adopt the standard application more easily and

Fig. 7-4



minimize the number of perceived gaps. It also has the flexibility to customize and integrate external applications. Rather than choosing one or the other, buying some third-party solution along with a standard implementation can build an optimal solution.

Representing the fits and gaps in a business process map in terms of the processes drives a better understanding within the implementation team and the organization. This is also needed to enlist the requirements for the design of the future solution.

Defining your requirements

Some projects start with lists of requirements derived as a set of atomic functional requirements, but without reference to mapped and documented business processes this can pose risks like the following:

- Late discovery of missing process steps
- Misinterpretation of the requirement by parties not experts in the related process
- Lower business engagement in the project
- Longer review cycles as the requirements and designs are less clear

Business process mapping helps draw the as-is processes to understand how the business is running right now, and the to-be processes to show how they work in the future. This also emphasizes the importance of business process mapping early in the project.

After determining what you keep or build, create the list of requirements, aligning them to the business processes for an accurate picture. You can use tools equipped for the traceability of the requirements.

Typically, in a process-focused solution, once the processes have been defined and agreed upon, the detailed requirements are derived and relevant designs mapped. There are tools within Dynamics 365 to help with managing the requirements. For example, Azure DevOps can work to create and track requirements, and if you are implementing

Dynamics 365 for Finance and Operations, you have the Business Process Modeler (BPM) as part of your lifecycle services (LCS).

The list of requirements is easier to be crafted starting from the business process mapping and it can be iteratively revised and refined.

Process-centric implementation lifecycle

The value of taking a process-focused solution does not end with a better definition of requirements. Adapting all the activities in the subsequent phases to be more process-based helps deliver better outcomes, regardless of underlying methodology.

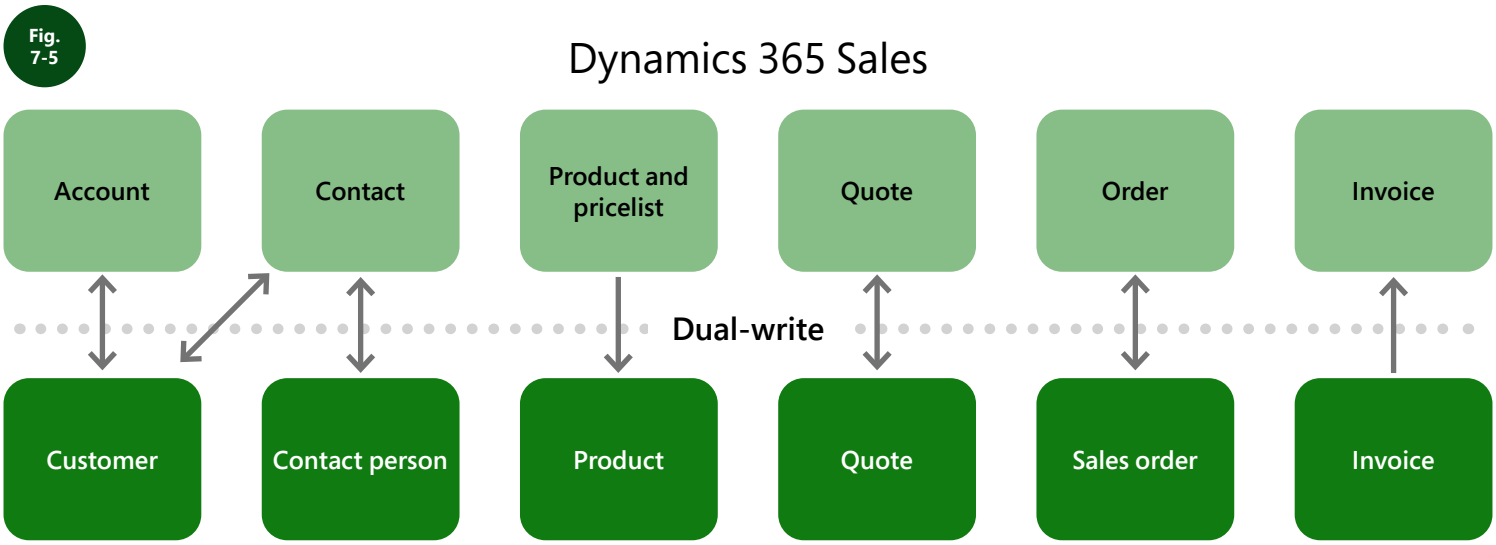
At go-live, a business application such as Dynamics 365 sees system users performing tasks and activities as part of a business process. The users are not executing on gaps, fits, or isolated requirements. It is important to keep this in mind when thinking of the full project lifecycle. The ultimate test of the project success is when the system is in operation and the design build testing actions are intermediate steps towards that goal. It is recommended to drive these phases with the end-to-end business process as the framework upon which all the related activities are planned, performed, and measured.

See the prospect to cash with Dual-Write integration to high-level end-to-end process map in **Figure 7-5.**

Design

When creating the solution blueprint, the vision of the overall solution architecture gives solidity and certainty to the project scope. This is the transition from scope as business processes to scope as designs within the system. As it is expressed in business language, it helps to ensure that business users can be usefully engaged in the review and approval of the scope and proposed solution. There are, of course, other views of the solution architecture in terms of data flow, systems landscape, and integrations.

Fig. 7-5



As part of the design phase in a Waterfall methodology, or during design work in sprints in a more Agile methodology, breaking down the end-to-end business processes into meaningful subprocesses provides manageable, bite sized units of work. It also allows better communication of the project tasks with business users.

The specific user stories, or configurations, and build work at a task level can be related to the subprocesses so that they have the correct business context. This makes it easier for business users to better understand the design and the business reviews and approvals of designs are meaningful.

If the business processes are collected in a structured and hierarchical process catalog, this catalog can be used to do the following.

- Help more easily and logically determine the sequence of work starting with the more foundational process designs before tackling the lower-level designs.
- Help highlight the interrelationships and dependencies between the different business value streams, departments, roles, and functions—this helps with designing a more cohesive and integrated overall solution.
- Help better manage the distribution of the actions and activities by leveraging the sequence and interrelationship of the designs in scope.
- A process flow can directly be mapped to the system design from configuration through data setup and functional run through.

- Help deliver working processes in Dynamics 365 software, by implementing processes, including end-to-end process as early as possible, and by taking a process-centric view.
- Help provide a better understanding of the solution by looking at the current capability of the emerging system to execute end-to-end processes.
- View the system in the context of a process, reducing unnecessary customizations .Better engage the business subject matter experts (SMEs) and business stakeholders by using the business language of processes.
- More rapid delivery of working software that better reflects and makes better use of the Dynamics 365 SaaS cloud world.

Deliver processes in Dynamics 365 as early as possible, taking a process-centric view. The benefits are practical and real, and often the only real barrier to taking a process-centric view of design tends to be the legacy of previous, more technical, and custom development-centric approaches, better suited to the pre-cloud and SaaS world.

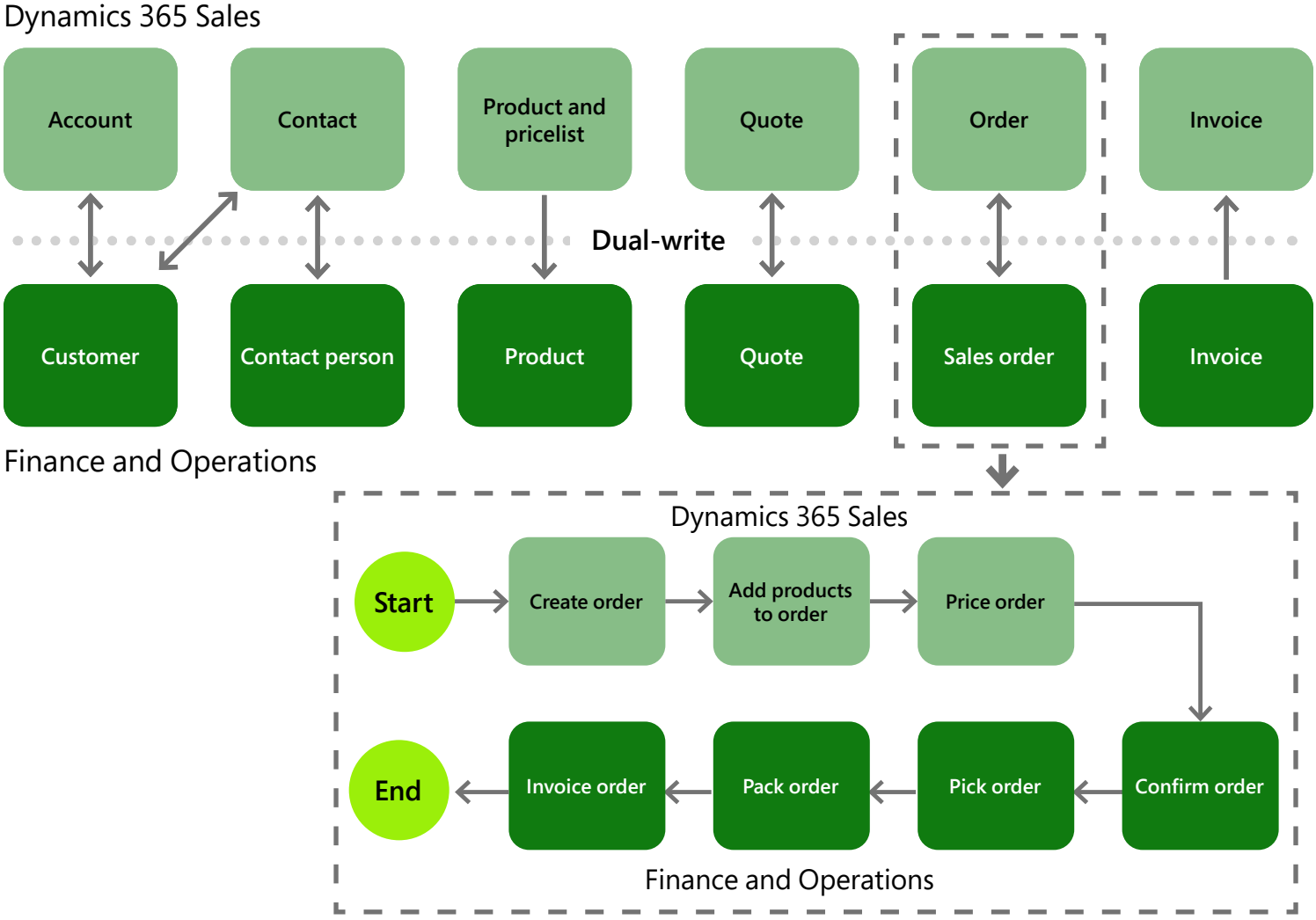
Having the high-level process, you can start breaking down the end-to-end business processes into meaningful subprocesses, as shown in **Figure 7-6**.

Development and configuration

A process-centric view provides an excellent basis for converting designs into software. It can help directly address the question of how to execute the business process within Dynamics 365. The configuration and development tasks have the background of the business process context, and do not need to be treated as an isolated delivery of a task from a long list of such tasks.

This isolation is still frequently seen in projects where the development and configuration work becomes highly technical and is divorced from the business process. This activity can become an end in itself, which does not serve the project well. Instead, when a data- and process-centric view is taken, the configuration and development tasks can be planned and managed in service of delivering a process embedded in Dynamics 365 software.

Fig. 7-6



A process-first view allows the team to collaborate more productively. The process designs, which now reflect the functions and configuration and flow within the Dynamics system, help the various teams communicate the context and better understand the delivery of any development. Compared to working from a task in DevOps, or from a paper design, the process background provides a better context and implications for a build task. For example, taking a process-centric view can help with the mapping of the access necessary for performing process tasks. That helps build a picture of the related security roles.

A process-focused view of the emerging solution also helps avoid the dangers of a primarily “fit gap” approach which can create point solutions. Instead, with a process-first background, there is a high probability that the tasks and activities crystalize into a coherent and well-functioning solution.

The project can also use the process-focused solution to conduct reviews of the solution with the wider business to confirm that the project is delivering on expectations. These reviews are much harder to do well without the supporting process definition. That's because without the processes in focus, the project cannot speak the language of the business. Many organizations are expecting some degree of business process transformation as part of the implementation of Dynamics 365. The reviews of the solution when the new processes are first explained in business process terms helps ground the subsequent discussion of the current state of the solution in clear, well understood business language. This significantly improves the quality and usefulness of the business review by minimizing the technical implementation jargon and instead concentrating on the "business of business."

Issues with resource constraints are well known. Showing incremental progress of the build and development using business process language improves the confidence of the wider business in the project. This can be key for project success.

Testing

A project that has taken a process-centric view reaps the benefits during testing. Other than the necessarily narrow focus of unit testing and some non-functional tests, almost all other test types should be rooted in some form of process definition. When there is an existing set of process definitions and the associated designs, and a system with the designs implemented, there are many advantages to the testing methods.

- It helps ensure full testing coverage and enables earlier detection of any missing or inadequately defined functions or processes.
- The testing can more easily follow the process-focused path already created earlier in the project.
- Testing has a natural focus on evaluating business process outcomes which tends to be a closer match to the intention of the design and eventual production use.
- It enables incremental testing of processes and subprocesses which in turn helps engineer quality into the solution.
- Testing end-to-end processes, which is how the system is used in production, is enabled.

Overall, testing in a business application like Dynamics 365 is a natural match to a process-focused view of the solution. Having a process-centric approach throughout the lifecycle allows testing to become the natural and obvious progression of the project delivery.

See Chapter 14, "Testing strategy," for more details on the overall approach to testing.

Training

Training to use business systems like Dynamics 365 applications is fundamentally learning how to conduct day-to-day business processes using the system. Most of the functional roles in the system interact with multiple functions and processes. Rarely do roles exist in a vacuum of their own process, but instead interact with other processes. Having business process flows defined and available in the process catalog, including flows across the seams of system roles, helps both in the collation of process-based training materials and to guide the training.

During the design process, if the roles are mapped against the processes as part of security design, they can be directly used for testing and for generating role-based training.

Even where some of the system roles may be very specifically restricted to a specialized function in the system, there is often a need to understand the upstream and downstream processes to perform a function well, and to understand any implications of any delays or changes in the flow.

If whole project approach is process-focused, including the configure/build activities, then the process of generating materials for training (such as Training Guides for Finance and Operations and Guided Tasks for Customer Engagement applications) goes smoothly, as there is a direct correlation between the system processes and the training processes.

The process-based training not only helps prior to go-live—it also can be used with new hires and when users change roles. It allows those new to the business to absorb the business process simultaneously while understanding how that process is performed within the system.

Roles can be easily defined in a business process and use the correct security roles for testing and training as shown in **Figure 7-7**.

Support

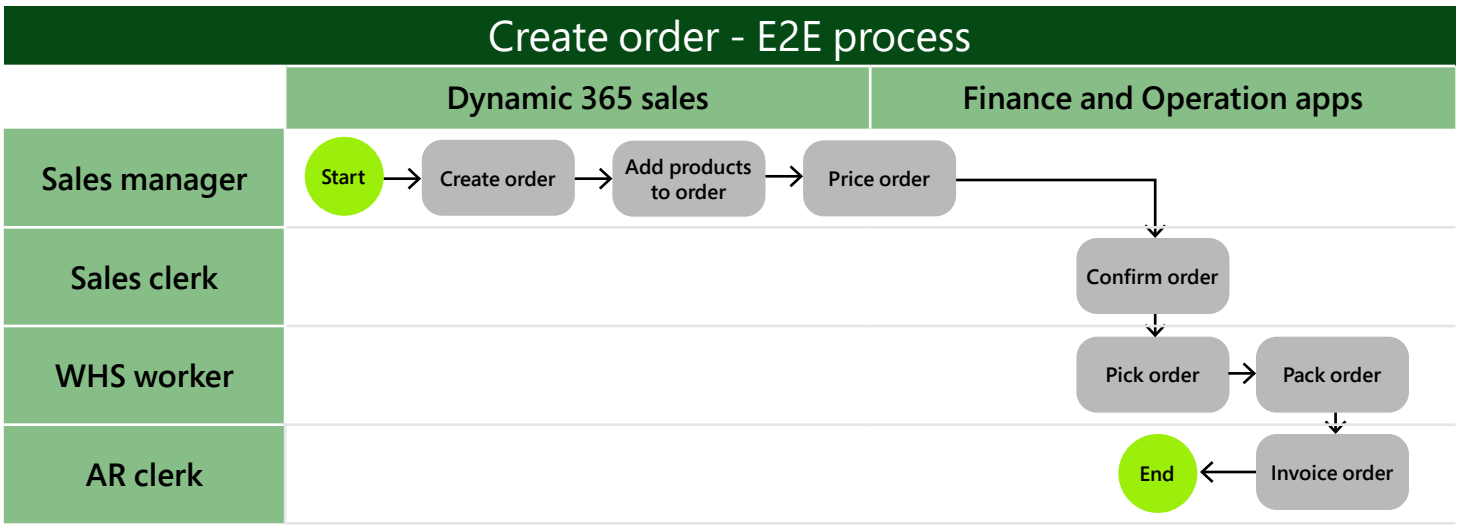
The process catalog created in the project provides more than the framework for implementation. It can also help with supporting the solution. Functional issues raised to support usually need to be reproduced by the support team to enable the team to investigate the root causes. A robust process-based definition of the flow allows support to recreate the steps defined by the agreed standard process flow. The visual depiction and description of the process allows the support team to speak a common language with the business user when discussing the issues.

This ability to place yourself in the business process helps reduce the number of back-and-forth cycles of communication and the overall time taken to understand the issue in business terms. It increases the confidence of the user that their reporting of the issue has been understood. This reduces anxiety and improves user sentiment of the system.

A solution that helps you to operate your business

To design and build a solution that has a great positive impact is not an easy task. There are many aspects to consider, and everything is so

Fig. 7-7



connected that even missing a little piece can significantly affect the whole project. The business processes flows are the common playground where we gather all the implementation artifacts together. They keep all the solution components and phases aligned. They are the line connecting the dots and the main instrument for communication that helps to drive changes on how the business runs and helps to minimize missing those little pieces.

As you have read through the chapter, business processes are involved during all the journey of the implementation and they also remain useful for future improvements. They have a perpetual impact in your business.

The creation and documentation of business process flows is to prepare the fertile ground where the seed will be for all those implementation activities that allow you to harvest optimized and efficient processes, risk depletion, cost minimization, role compliance, integral solution, and, as an ultimate goal, a successful implementation.

Checklist

Business processes focus

- [ ] Ensure the business process view of the organization is at the core of the definition of the project.
- [ ] Clearly articulate the key business processes that are in scope and the respective personas so they are understood by all involved parties in the implementation.
- [ ] Ensure business model analysis, process engineering, and standardization strategies are considered part of the project definition and deliver a strong process baseline before implementation starts.
- [ ] Collect the business processes in a structured and hierarchical process catalog during the requirements phase.

Process-centric solution

- [ ] Use business processes for each phase of the project to deliver better outcomes (all phase activities are better planned, performed, and measured).

Opportunity for optimization

- [ ] Explore opportunities to evolve, optimize, and consolidate your processes as part of the implementation to meet the digital transformation goals and drive user adoption.
- [ ] Ensure the business process definition is complete and considers all activities and subprocesses.
- [ ] Take advantage of the latest SaaS technology to drive efficiency and effectiveness for the process optimization.
- [ ] Ensure future readiness when mapping your business process to the solution by incorporating configurability by design.

Fit gap analysis

- [ ] Adopt a fit-to-standard approach and align to the philosophy of adopting wherever possible and adapting only where justified.

Case study

The journey to a process-focused solution

A large, multinational organization that provides contract-based servicing with some manufacturing and distribution services was planning to replace parts of its 15-year-old tier-one ERP and CRM systems and other related legacy systems.

The wrong road

The project started with the implementation process that they had traditionally applied to other IT projects. The traditional approach was strictly Waterfall with a heavy documentation bias. The initial phase of requirements gathering was conducted by using the old systems as the reference point. A very long list of requirements was gathered, with the assistance of an implementation partner, across the various functions. The requirements gathering phase was extended a few times as the requirements review took significantly longer than expected.

The reviews needed multiple parties to read, understand, and validate the requirements and there was a concern among the business approvers that if they missed any requirement, or even a nuance, their system would not function well.

So, there were multiple document-based reviews where comments

from the approvers were formally submitted and the project team wrote responses to them, which in turn generated even more comments from the approvers. The spiral of these reviews finally concluded, but the business was not entirely convinced that they fully understood what they had approved and hoped that the design phase would provide more clarity.

The design phase was similarly based on writing and reviewing complex design documents and was running late. The focus tended to be on the "gaps" identified, and as these were not always placed in context, the discussion with business users was not always productive and was often at cross purposes. As further delays accrued and the business users were becoming even less confident about the proposed solution, the stakeholders decided to review the project direction and the reasons for the continuous delays and general dissatisfaction.

The change of direction

As part of the review, which included third parties, several recommendations were made. The primary recommendation was to adopt a more process-focused approach and apply the processes as the framework for a more Agile way of working. The first step in the process was to define the end-to-end process at the highest level in order to establish the boundaries of the project scope in business process terms. The subsequent steps then generated two further levels of visualization of the process detail for the business process streams. The project took deliberate steps to try to define the processes in terms of the business flow and desired business outcomes, rather than replicating the existing systems. This process mapping exercise was done at an accelerated pace, working as combined business, project, and implementation partner workstreams.

The right road

Once there was a reasonable map of the processes, the requirements were mapped to the processes so that they could be understood in the context of a process. As a result, many of the requirements were restated to be better anchored in the context of a business transaction and thus less likely to be misinterpreted or removed for being redundant.

The processes at level two were put into a storyboard by using the expertise and leadership of the partner solution architect and the customer's lead functional expert. The customer's lead functional expert used the process flows to create logical end-to-end process flows across workstreams so the delivery of software would be meaningful. The partner solution architect provided the Dynamics 365 application view of the embedded standard processes, dependencies, and constraints to ensure that the sequence of process delivery would be efficient within the Dynamics 365 applications. This storyboard would drive the sequence of work from realizing foundational processes to more peripheral ones. Furthermore, the level two end-to-end processes (such as "Prospect to Cash") were prioritized by considering the core/most frequent/ baseline path through the process as a higher priority. The more specialized and less frequent variations were sequenced to be designed and delivered after the core processes.

This set of processes in the storyboard were then mapped to an overall design within Dynamics 365 applications, generating a process-focused solution blueprint. It was reviewed against the existing technical, system, and data solution blueprint to create a rounded solution blueprint.

The delivery of the processes was distributed into sprints of four weeks each, so each sprint delivered a meaningful part of the storyboard for each of the workstreams, with an emphasis on delivery of end-to-end processes as rapidly as possible. A high-level plan was constructed based on the sequence, dependencies, and estimated effort related to process delivery.

The sprint level planning was performed on the processes in scope for that sprint, defining a more detailed process flow when required. Documentation was kept to a minimum and the processes were designed in the Dynamics 365 system in collaborative workshop environments. All related activities such as data migration, integrations, testing, training, change management, etc. were performed based on the processes in scope. Each sprint culminated in a "conference room pilot" (sometimes called "working demo" or "playback"), where the business SMEs presented the new logical business process and how the processes

were designed and implemented in the system via a demo to an invited business audience.

At each sprint end, the business attendees reviewed and approved the incrementally complete and functioning processes in the Dynamics 365 system instead of reviewing and approving complex and technical design documents. Individual gap designs that had previously been circulating on paper for weeks and months were getting translated into end-to-end working software. Business engagement significantly increased as the project was talking their language and they were working directly with the emerging Dynamics 365 system rather than with abstract design documents and lists of requirements.

Arriving at the destination

The project further reinforced the process-centric project approach, beyond design and build, by using the processes to script and drive end-to-end testing, reporting status and progress as "ability to execute a process within the system." The senior business stakeholders on the steering group also connected with the project more meaningfully as they were finally able to understand the project readiness and business operations implications.

The project successfully went live, and the customer continued to adopt a process-centric view throughout the remainder of their go-lives in other countries. The implementation partner decided to adopt this process-centric approach as their new standard implementation approach for their other projects because they could clearly see the benefits.

8

Guide

Project governance

Introduction

As the footprint of cloud-based solutions increases, we need to work with and address different business and technology challenges.

For example, Dynamics 365 applications are increasingly able to use low-code/no-code technology. Also, customers have higher expectations on speed to value. The number of custom development-heavy, multi-year, big-bang projects is diminishing; customers are looking for early and incremental value, even on long-term, multi-country rollouts. Cloud-based Dynamics 365 software as a service (SaaS) applications with regular, non-breaking system updates are changing the way we deliver business applications. Many of the limitations and sluggish project governance processes of multi-year, on-premises implementations should not be carried into the cloud.

We need to ensure that we're creating a project governance model that is fit for the world today and tomorrow. This includes considering new governance disciplines and reviewing the classic governance model for effectiveness. We also need to consider that many partners, system integrators, independent software vendors (ISVs), and customers may have their own approach and governance standards that they may have used previously.

For a successful project governance strategy, consider these main areas:

- Project goals
- Project organization
- Project approach
- Classic structures
- Key project areas
- Project plan

We explore each of these areas in more detail in this chapter.

Our objective in this chapter is to provide an overview of the importance of good project governance. We discuss recommended practices on how to define the different areas of project governance and analyze and assess the effectiveness of your governance. We also describe in more detail some of the key areas of project governance and the specific considerations that can help stakeholders, project leaders, and implementation teams increase their chances of success.

The project governance topics discussed in this chapter are relevant to any implementation methodology; we focus on the underlying principles and provide guidance in a way that allows customers, partners, and others to evaluate their own approach and adjust as necessary.

Objectives of project governance

Project governance should be designed to provide the framework and techniques to do the following (**Figure 8-1**):

- Map business goals into actions and measures that deliver the goals
- Structure, plan, and reliably, safely, and efficiently drive the project
- Anticipate and avoid common issues
- Detect emerging risks and divergence from the planned path and support corrective actions
- Provide the right amount of flexibility and agility to respond to unexpected events and to adjust to the specific constraints of the customer's business or project

Fig. 8-1

When designing or reviewing your project governance model, you should examine how well it addresses each criterion. Furthermore, it's worth reexamining this in the specific context of the goals and constraints of your current project.

In this chapter, we explore what project approaches, governance models, disciplines, and techniques project teams should consider when preparing

for and implementing a Dynamics 365 project. Similarly, we use Microsoft's Success by Design framework to highlight critical questions to consider throughout the project lifecycle to help drive success.

Why is project governance important?

All system implementation projects, including Dynamics 365 applications, need good project governance to succeed. However, business application projects have specific needs and challenges, and aren't theeasiest of projects to implement. Business applications directly impact and are directly impacted by the processes and the people in the business. For a business application implementation to be successful, it's not sufficient to have good software and good technical skills; the project also needs to have good governance processes that include the business.

A Dynamics 365 project needs to understand the business requirements deeply and at a domain and industry level. It also needs significant and sustained participation from business users representing many different roles, disciplines, and skills. Many, if not most business participants, may not have previous experience implementing business application or business transformation projects, let alone Dynamics 365. This puts an additional burden on ensuring that the methodology, approach, and governance models are sufficiently robust to support and drive the Dynamics 365 business application project. It also requires business users to gain sufficient knowledge of the standard capabilities of the Dynamics 365 application to better map their underlying business requirements to the out-of-the-box solution.

It's perhaps worth reminding ourselves of the most common problems related to implementing business applications (**Figure 8-2**):

- Unclear or ever-changing project scope
- Late discovery of project slippage
- Disputed areas of accountability or project responsibility
- Low or sporadic user and business engagement
- Delays to go live (often at a late stage)

Fig. 8-2

Typical governance issues

- Unclear or ever-changing project scope
- Late discovery of project slippage
- Disputed areas of accountability
- Low or sporadic user and business engagement
- Delays to go live at a late stage
- Technical issues hiding underlying governance issues
- Mismatched expectations between customer and partner
- Stakeholders blaming each other

- Technical issues hiding underlying governance issues
- Mismatched expectations between customer and partner
- Stakeholders blaming each other

The list is long, and you may have seen other issues, but most of these issues result in project delays. However, the root cause of the issues tends to lie in gaps in the definition of the governance model, or in the effectiveness of operating the project governance processes. Even after the project goes live and meets most of the business requirements, if the project delivery isn't smooth, it can create stakeholder dissatisfaction and a lack of confidence in the project.

In the context of the implementation lifecycle, you should define the project governance model as early as possible, and certainly as part of the Initiate phase of the project, because it directly influences the project approach, oversight, resources, and project plan. During the Initiate phase, if the customer or the partner has an underlying governance model, you should review it and adapt it for this specific project, with consideration for any specific business requirements and constraints.

Next, we explore the various areas that you should think about as part of establishing your project governance model.

Project governance areas

Project governance is a wide topic and can encompass multiple different practices and disciplines. Various methodologies exist in the market, including those that are customized by partners and customers. Irrespective of the methodology, you should consider some fundamental principles when designing your governance, or when reviewing and revising it.

The vast majority of projects have some form of governance; we're not describing setting up the common governance disciplines here. Instead, based on our experience across thousands of projects, we look at the areas where we often see the need for reinforcement and expansion to drive a more reliable and successful project delivery experience:

Project governance

- Project goals
- Project organization
- Project approach
- Classic structures
- Key project areas
- Project plan

- **Project goals** We look at whether the goals deliver solid foundations for the implementation, are well defined, and provide the necessary direction to the project
- **Project organization** We explore how the right project organization and related roles and responsibilities help or hinder the efficiency of the project
- **Project approach** We examine the impact of the right methodology and the wider approach on project success
- **Classic structures** We review how we can examine the most common governance processes to ensure they don't hide the real issues and provide false reassurance
- **Key project areas** We look at how we can improve effectiveness in areas of the project that get a lot of technical attention, but frequently suffer from insufficient direction, planning, and oversight
- **Project plan** We examine the critical importance of good project planning and extract some of the planning-related lessons from the previous sections

Project goals

Well-defined project goals are essential for steering a project and for defining some of the key conditions of satisfaction for the stakeholders. Often, the project goals are described in the project charter or as part of the project kickoff. In any case, it's worth shining a spotlight on them as part of the Initiate phase of the project and regularly throughout the project. We recommend taking deliberate actions to reexamine the goals in the context of your latest understanding of the project.

When reviewing or crafting project goals, consider the following:

Are the goals clear and realistic?

If your goals are overly ambitious or vague, the project may expend unreasonable effort and time in the attempt to meet them, only to fall short of the expectations set. Of course, no project intends to create unclear or unrealistic project goals, yet there is a long history of such examples, so we recommend conducting an honest review to confirm these against the latest project scope.

Where possible, create objective goals, which are easier to measure, work with, and track. Some qualitative goals such as "the new system should work at least as well as the old system" are open to interpretation and can drive the wrong project behavior. They can be misinterpreted and force the team to design customizations in the new system to try and recreate the old system. Instead, consider the goals in terms of the overall business outcomes you desire; this framing allows the project to deliver them in a way that is most efficient in the new system architecture.

Are the goals aligned with the business priorities?

If the goals aren't well aligned with the priorities of the business stakeholders and the business units in scope, the project won't receive the necessary business ownership, participation, or attention. Consider whether the IT priorities and the business priorities expressed (or implied) by the project goals are complementary. We also recommend explicitly verifying that the priorities given by the organization are well understood and aligned with those of the operating business units. Sometimes group-led policies and requirements implied by the project goals, which are expected to deliver additional controls and value at the group level, may generate additional work at the operating unit level.

It's essential to have all the stakeholders pull the project in the same direction—conflicts at the level of misaligned goals are extremely hard for the project team to solve. For example, if Finance leadership is looking to improve compliance by adding more checks and approvals in a purchase process, and Procurement leadership is looking for a faster, less bureaucratic and more streamlined purchasing process, unless the goals are balanced, the project delivery will falter. The end result will probably disappoint both stakeholders. Another common example is when IT leadership has a goal of a single platform or instance for multiple business units, but the business unit leadership has no goals to create common business processes. This mismatch can remain hidden and undermine the feasibility and efficiency of a single platform.

Again, we recommend reviewing the goals with the stakeholders to confirm that they will promote the right project delivery and business outcomes. Successful projects have not only ownership of the project

goals from the business, but also ownership of the successful delivery of the project goals.

Are the project goals well understood by all the project members?
Some projects rely on a single kick-off meeting to communicate the goals. However, many goals would benefit from more in-depth discussion (especially with project members from outside the business) to better explain the underlying business reasons. Consider how you can reinforce this communication not just during the initial induction of a new project member, but also throughout the project lifecycle.

Have the project goals been correctly translated into project deliverables?
Once a project starts the Implementation phase, the necessary attention needed for the day-to-day management and delivery can sometimes overshadow the importance of the strategic project goals. This is one of the key findings from project post go-live reviews—the original aims of the project faded into the background as the project battled with the day-to-day challenges.

Once a project starts the Implementation phase, the necessary attention needed for the day-to-day management and delivery can sometimes overshadow the importance of the strategic project goals.

In many cases, projects are driven by the fit gap list, which takes a very narrow view of the business expectations. Consider specifically reviewing the initial scope of the project and initial solution blueprint (and project plan) with the business stakeholders to assess how well the goals are mapped to the project deliverables and take any corrective actions.

Is the governance process structured to formally monitor progress against the goals?
This is an often-neglected area; sometimes projects only seriously review the goals or success criteria as part of the final go-live assessment, which is too late. Try to avoid this by defining specific structured processes to ensure the assessment is ongoing. For example, look at how the solution design blueprint helps meet the business process objectives and how the data migration and reporting strategy meets the information access objectives. Monitor to confirm that the project priorities are aligned with the project goals and aren't being diverted by other considerations or day-to-day challenges.

Project organization

Projects implementing business applications tend to have common structures and roles (such as project manager, solution architect, subject matter expert, functional consultant, and technical consultant) that are recognizable across different projects. The difference in the effectiveness of project teams comes from the way in which the project organization functions in practice compared to the theoretical constructs (**Figure 8-3**).

When assessing your project organization, consider the following:

How well is the project team aligned to the business?
Teams that have good alignment between the business streams and project functional workstreams tend to have more effective and higher-velocity projects. A common model is for each key business stream to be matched with a corresponding project workstream. An experienced leader from the business stream is usually seconded to the project and takes on the role of the lead subject matter expert (SME). For larger projects, multiple SMEs may be appointed for a single workstream.

Fig. 8-3

The direct involvement of a business stream with the project is usually the most successful model because it engenders trust both ways—from the business to the project and vice versa. If you have multiple organizational steps between the business and the project team SMEs, the level of business ownership of the solution design diminishes, as well as the quality of the input from the SMEs.

Does the project organization include active and appropriate senior business stakeholders?
Projects are far more successful

when senior business stakeholders have clear roles and are active and deeply engaged in driving the project. A good sign for the project is when the senior stakeholders are keenly interested in the status of critical processes and are making a direct impact on the prioritization of tasks and making the corresponding resources available.

Projects in which the senior stakeholders are more passive and just occasionally asking "How is it going?" or "Let me know if you need something" tend to have poor outcomes.

Does the structure promote cross-team collaboration?

Examine if the project organization structure, reporting lines, team leadership and composition, communication channels, team objectives, and delivery approach naturally generate sufficient cross-team collaboration, or if they naturally encourage silos.

Danger signs are project organization structures with the following:

- Team members performing the delivery work on the ground have to navigate multiple layers of management to overcome issues
- Members of workstreams rarely work with other workstreams
- Responses to ad hoc questions on a team member's task status and issues are routed via the project manager—this also introduces mistrust within the team
- Workstream communication channels are all hierarchical, with few organic, cross-workstream objectives

A good way to determine the level of cross-stream collaboration is, for example, to scrutinize how closely, and how often, the data migration workstream works with the various functional and technical workstreams' design. How closely entwined are the ongoing delivery objectives of the data migration team and those of the functional team? Do they primarily communicate hierarchically or across to their teammates in other areas?

Is accountability and authority well defined at the project leadership level?

Projects where the wrong role is accountable for delivery or where the accountability is diffuse tend to have low velocity, with project

decisions stagnating. Projects where the accountability is given to someone too junior in the organization to have the corresponding authority may also languish without sufficient direction and resources.

How well are the team roles and responsibilities defined?

Every role on the project needs to be well defined, including roles that may be part-time on the project. The critical role played by business users that aren't seconded to the project is often ill-defined. They may typically be necessary for defining a detailed process or data requirements, for approving business processes, or for preparing and performing testing. These are critical activities, and the corresponding roles should be well defined so the resources can be planned and their tasks and status are visible to the project leadership.

How well are the team roles aligned to the solution complexity and system design requirements?

You should conduct an honest analysis of the experience and ability of the resources in key roles when compared to the complexity of the design and its constraints. Be wary of job titles that don't match the experience and authority that normally accompany such titles.

This is particularly important for the key roles of lead solution architect, lead technical architect, and project manager.

Are the levels of resources proportional to the level of effort and complexity?

By the end of the Initiate phase of the project, you should have a reasonably credible high-level solution blueprint, a reasonable estimate of the backlog and effort, and a high-level project plan. This should provide a good grounding for reviewing the level of resources planned for the key roles.

This honest review is especially recommended for the roles that are likely to be the most constrained, so that mitigation plans can be initiated in a timely manner.

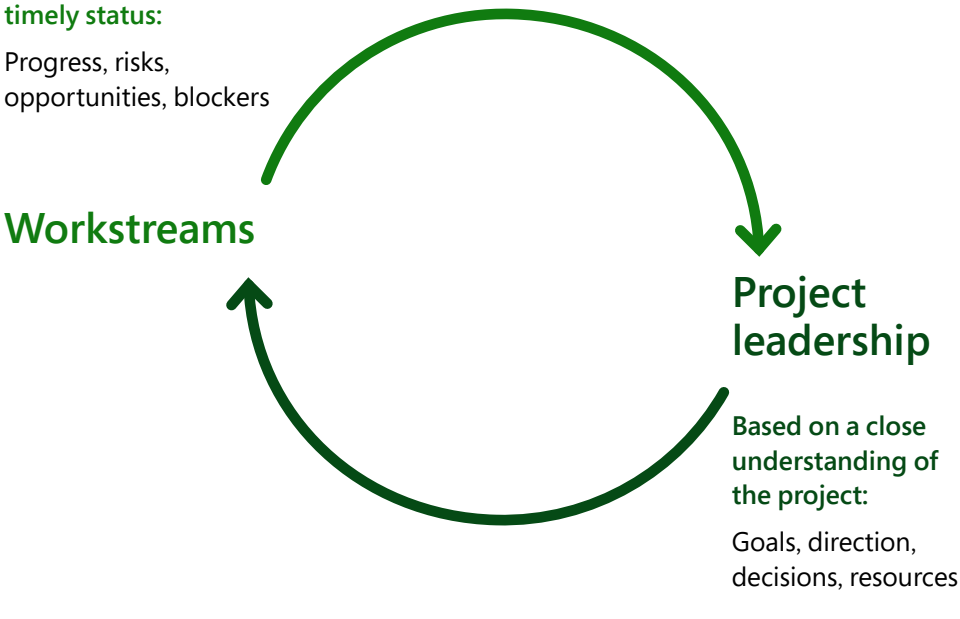
During the implementation, you should regularly assess how the day-to-day working of the control, communication, and feedback functions are being helped or hindered by the project team organization

(**Figure 8-4**). How well does the project organization structure facilitate or constrict the undiluted and timely flow of direction, guidance, and decisions from the business to the project workstreams? In the other direction, does the structure enable or hinder the flow of accurate, actionable, and timely feedback from the workstreams to the project leadership and business stakeholders?

A typical setback on projects is the leadership team not receiving timely and accurate feedback on project issues, therefore adding unnecessary cost and delays. This is often due to structural issues created by a project organization that doesn't help the flow of meaningful information. Projects with a lot of hierarchical communication and reporting-line hoops to clear (especially on large projects) tend to have difficulties in surfacing accurate data promptly. This can result in project challenges stagnating for too long, which leads to delays and additional cost.

Many projects start with the ambition of some model of partnership and shared leadership with the various parties involved. The projects that better achieve this ambition are the ones that meaningfully map the commercial and partnership arrangements (or expectations if informally agreed) and their respective expertise to the project organization. Projects tend to have better success when the customer has a clear leadership (and ownership) role in the organization structure driving the overall project. Similarly, unless the customer has experience and capability in implementing business applications, having the implementation partner in a leadership (and accountability) role for providing strategic guidance on the implementation methodologies, system design, and technical procedures allows for the best mapping of leadership roles to those with the best experience. Of course, depending on the specific expertise between the customer

Fig. 8-4



and partners, the actual balance of responsibility lies somewhere in the spectrum of these arrangements, and the organization structure should be mapped to that agreement.

In business application projects, some critical project roles have a disproportionate impact on the success of a business application project. For most projects, the role of the solution architects and the project manager is particularly critical to success. The project manager and solution architect roles are important both from the implementation partner and from the customer; the customer solution architect is usually described as the lead business SME or the business lead. Depending on which areas of your project are particularly significant or risky, you may want to add other roles to the list of the critical project roles. You should then pay extra attention to ensure that the individuals chosen for these critical roles have the right level of experience and capability. For example, an ineffective solution architect may not be able to direct the solution design blueprint such that the different elements of the solution design come together into a coherent and efficient solution that matches the business needs.

You should also confirm that the project organization allows all roles to be successful by enabling them to properly assert their expertise and strongly influence the design and implementation, thereby delivering the full positive impact of their roles.

Project approach

When talking about project approach, one of the dangers is that it can be assumed to be synonymous with the project implementation methodology. This can then leave a vacuum in the processes that need to be defined outside of the implementation methodology. This can be especially true if the implementation partner is providing a limited, technical implementation methodology. If you're the customer, you should consider the wider set of processes required to define your project scope, manage your resources, and manage the changes, tasks, and processes. Prioritize areas that aren't directly covered by the partner but are necessary for every business to perform in support of

a business application implementation. You should explicitly identify areas that would be the responsibility of the customer to manage. Then confirm that you have adequate definition of the approach and the right level of governance planned for each of these areas.

The areas excluded from the implementation partner's responsibility will vary, but the following areas are typically involved:

- Defining the scope of the project (the to-be business process and system requirements)
- Managing internal project resources and liaising with wider business teams
- Managing the process for the proper participation of all the relevant business units and roles
- Managing the source data quality for data migration
- Training the internal project team and the end users
- Data, system, and process validation by the non-project business users, such as user acceptance testing (UAT)
- Security and access definition and validation
- Implications on the wider IT enterprise architecture and wider business processes
- Interpreting, applying, and validating the requirements related to relevant regulatory bodies
- Cutover management
- Managing the non-Dynamics 365 systems involved in system integrations
- Managing communication within the business and with customers and suppliers
- Budget management
- Business and organizational change management
- Non-functional requirements such as performance management, system security, and business continuity planning
- Transitioning and operating the support model and other post-operational duties

This is not intended as an exhaustive list, more of an indicator that the overall project approach needs to consider a much wider set of functions than what may be covered by a typical implementation methodology or a commercial contract. Some projects define the

overall approach in a project charter or similar document. You should examine if the project approach adequately covers the missing areas from the implementation methodology.

One of the discussions that regularly occurs about project approach is the debate on waterfall versus agile. Methodologies based on either of these principles have their advantages and disadvantages. For more detailed discussion on implementation methodologies, refer to "Chapter 5, "Implementation strategy." From a project approach perspective, whichever methodology (or hybrid) is adopted, it's important to identify and record the specific risks that logically emerge from that methodology and have a mitigation plan for each one.

For example, if your chosen methodology doesn't include early analysis and defining a solution blueprint that reflects the whole design scope, the project may stumble from designing specific solutions from sprint to sprint. You may find that the designs in later sprints can't be built upon the previous sprints due to the constraints of not having analyzed and considered the high-level design for the whole end-to-end process. We often see this in pure agile projects.

Similarly, if the chosen methodology is strictly linear and mostly document-based, you risk spending an enormous amount of time and effort in analysis, design, and coding phases with little exposure to the working solution and limited feedback on how the solution performs in the system. This is a risk we see in many pure waterfall projects. You need to identify such risks in your chosen methodology so you can discuss, accept, understand, and specifically address them in the project approach.

Sometimes when a methodology is being selected, there is an assumption that a methodology that works well for a bespoke software development project will apply with minimum changes to a highly configurable Dynamics 365 cloud-based business application implementation. Similarly, methodologies that suited the old world of on-premises business application implementation with long implementation durations aren't best adapted to today's more rapidly evolving cloud world. Consider the very different

circumstances of a modern, cloud-based Dynamics 365 business application implementation:

- The implementation is for a cloud-based, packaged SaaS software application
- A significant (if not a majority) of the activities aren't coding, but business process design, setup, configuration, data migration, validation, and more
- Some critical activities for a Dynamics 365 application implementation (such as extracting a fit gap report from a fit-to-standard analysis) aren't covered by more generic software development implementation methodologies
- There is ever-increasing functionality in the Dynamics 365 business application of powerful configuration options or low-code/no-code options and the ability to use the Dynamics 365 Power Platform in citizen-developer and professional developer modes
- The methodology must address the fact that a very significant part of the implementation process revolves around the understanding and configuration of business processes and working closely with business users throughout the project lifecycle
- Although Dynamics 365 applications provide a platform for custom development, the chosen methodology must address the fact that the development process is building on, or extending, the standard business process flows designed in the system
- Dynamics 365 applications are a cloud-based, SaaS business application with a regular update rhythm that needs to be recognized as part of the new way of working

All of these factors (and more) mean that the implementation methodology needs to be directly relevant to the nature and needs of a Dynamics 365 business application project.

When looking to determine what the overall project approach should address, consider the specifics of your whole project solution build lifecycle and make any additions, adjustments, or highlights to the standard methodology. Confirm that the key processes and controls you expect are adequately addressed as part of the governance of the project:

- Does the methodology and approach enable a good definition of the scope and a solution blueprint?

- Is the testing strategy fit for purpose?
- Do controls such as regular and appropriate reviews by the project team and the business exist?
- Is a good process for project status analysis and reporting in place?

You may want to define a set of critical processes, deliverables, and controls to ensure you have the right coverage from the project approach (just as you would for functional and technical requirements).

Once you are in project implementation mode, and the focus is on day-to-day tasks, it's easy to lose sight of the ideals agreed in the governance model at the start of the project. You should establish regular checkpoints to go through the formal discipline of honestly comparing the actual ways of working on the project with the governance model and taking corrective actions as necessary.

For example, analyze a random sample of how a given feature, requirement, or user story is actually progressed through the lifecycle, or take a scope change item and trace how it was handled to resolution. Did they progress through their lifecycle as you expected, and did your governance model help deliver a fast and quality result? Taking a (preferably independent) regular pulse of the real, practical implementation of the processes and controls helps keep the project approach relevant and reduces risk.

Classic structures

Most, if not all, business application projects have the common, classic governance structures in place. This section doesn't give a general overview of historically well-known disciplines around these common governance areas; instead we look at how to assess the true effectiveness of these processes and controls in practice. The mere presence of these classic governance structures such as steering groups, program boards, or risk registers can sometimes lull projects into thinking that they have adequate active governance. Let's dig deeper into these areas to explore how we can get better insight into their function and evaluate their effectiveness.

Steering groups

Most projects have some form of steering group in which the project sponsor, senior business stakeholders, and project leaders meet regularly to discuss and review the project. A lesson learned from participating in multiple such steering groups across multiple projects is that the effectiveness of these meetings varies hugely. Common factors that can impact the effectiveness of a steering group are when the purpose of steering group meetings is diluted or unclear, and when project status reporting isn't easily understood, accurate, or actionable (**Figure 8-5**).

Steering group meetings

Steering group meetings can negatively influence project success if the project manager presents the status and the steering group is simply checking in with very little understanding of the details of the project, no direct engagement, and a lack of knowledge to challenge and explore issues more deeply. Projects that are considered able to manage themselves despite some issues are often surprised by a sudden crisis, resulting in delays and dissatisfaction.

An efficient and effective steering group has leadership team participants that understand and perform their primary function of steering the project. This requires them to invest the time to understand the content of the project at a sufficient depth so that they can read between the lines of the project status report, ask the right questions to understand how and where they can help, and proactively direct the project.

Fig. 8-5

Diluted purpose

- ✗ Little understanding of project details
- ✗ No direct engagement
- ✗ Lack deep knowledge

Clear purpose

- ✓ Understand primary function
- ✓ Time invested
- ✓ Comprehensive knowledge

Apply similar, regular assessments of the effectiveness of other such boards, like the change control board or the design authority board.

Project status reporting

Accurate, meaningful, and actionable project status identification

and reporting is the lifeblood of a project. Business application implementations often involve complex, multiple strands of work, all of which need to come together seamlessly and on time. Multiple parties are often involved in the project delivery—business SMEs, business IT, business project managers, and their equivalent roles from the implementation partner, as well as other third parties. In addition to the main functional and technical delivery, there are multiple other disciplines, like data migration and security. All these roles have their own specialist tasks to perform, and managing and monitoring all these different types of tasks to provide a coherent project status isn't easy.

Additionally, specialist teams may provide an overly technical view of the status in terms that only that team understands. Often, project-level status reporting is driven by fit gap progress, which isn't necessarily the critical part of the project. Consider the following when looking for effective project status reporting to the steering group:

Is the project status meaningful and actionable for the intended audience?

For example, when presenting internally to your own specialist team, a detailed and technical view is very appropriate; however, when presenting the status to a steering group or to non-project team members, consider how to express the status in terms that are understood and actionable.

Is there evidence to believe that the project status is accurate?

Consider how the project status data has been gathered, analyzed, summarized, and presented. Understand the level of uncertainty associated with the data. Is there a history of the project status data and related estimates versus actuals that can provide some idea of certainty? Steering groups have to consider the degree of uncertainty associated with the status when assessing risks and making critical decisions on moving or adding resources, approving budgets, and setting timelines. This means testing the accuracy of the status by looking at it from multiple angles to ensure that it provides robust and actionable data.

Is there a comparison between the planned and actual status?

Project status reports only have meaning when they show an easily understood comparison between the planned status and the actual progress—the report must be able to answer questions like "Are we

on track?" "What is the size of the remaining work to completion?" "What do we, as the steering group, need to do to get the project on track and keep it on track?" The absence of comparing planned work to actual and remaining work doesn't allow the steering group to take necessary actions to help recover.

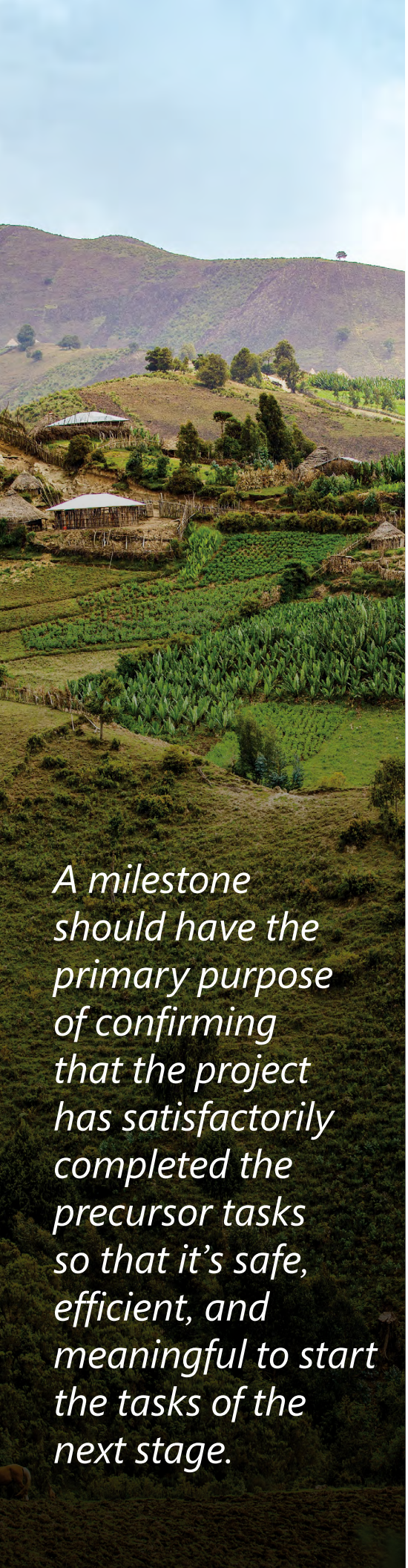
Risk register

Most projects have some form of a risk register. When used well, this register is a good way to communicate risks and help teams focus their attention on removing barriers to success. The following are examples of ineffective use of risk registers:

- Risks that are of little practical value to the project are being used to shrug responsibility or provide cover against blame.
- Risks remain on the register for a long time with just new comments and updates being added weekly at each risk review meeting, but with no resolution. This implies that either the risk isn't getting the attention it deserves or it's difficult to resolve and consuming more and more resources without results. In any case, you should either treat risks stuck in this loop urgently with focus or accept them with a mitigation so they don't drain the project over time.
- Risk priorities don't reflect the project priority at that stage of the project. You should take care to ensure that the risk register doesn't create a parallel project effort with its own priority and path. Risks and issue resolution should be incorporated as part of the main project delivery.
- The risk register has a very large number of risks, many of which are stagnant. Consider how many risks the project can realistically work on at any given time and trim the register according to real project priority.

Stage gates

Stage gates or milestone-driven planning and reviews are a common feature of the majority of business application projects, including more agile projects. These milestones are regarded as important checkpoints spread throughout the project timeline, which the project can only pass through if they have met certain criteria. The reality of many



projects is that the checkpoints don't always perform their intended function. There may be several reasons for this, and projects should examine if they are suffering from the following limitations:

- **Criteria for the milestone are unclear** You should strive to create very explicit criteria for entering and exiting a milestone. If the criteria are difficult to measure, it's difficult to take unequivocal decisions based on data.
- **Exit and entry criteria aren't respected** Projects are typically under great pressure to get past one milestone and move into the next phase, either because of resource usability or commercial payment triggers, or because the project optimistically believes that it will somehow catch up with the stragglers that didn't meet the milestone criteria. Sometimes a project is also pressured to move to the next phase because it implies progress. This self-delusion can create additional risks in the project; uncontrolled disturbances in project deliverables can delay the ultimate goal of a successful go live. The debt incurred by skipping the criteria has to be paid back at some point—the later it's resolved, the higher the cost. When considering allowing a milestone to pass when it hasn't met all its criteria, project leadership should strongly consider the nature and size of the incomplete tasks, precisely how these tasks will be addressed, what impact the lagging tasks will have on dependent tasks. This allows for a managed transition between milestones with the risks fully understood and a clear and realistic action plan for the stragglers.
- **Milestones don't reflect true dependency** From a project lifecycle perspective, a milestone should have the primary purpose of confirming that the project has satisfactorily completed the precursor tasks so that it's safe, efficient, and meaningful to start the tasks of the next stage. There may be other perspectives, such as commercial ones to trigger stage payments or just periodic points for the project to review progress or cost burndown, but we're focusing on a project lifecycle point of view. Some common examples of stage gate criteria are as follows:
- **To-be business processes defined** An approval step from the business to confirm the business process scope helps ensure that prior to spending a lot of effort on detailed requirements and design, the process scope has been well established and agreed upon.

- **Solution blueprint defined** This helps ensure that the requirements have been analyzed and understood sufficiently well to be able to define an overall, high-level design that is confirmed as feasible. Additionally, the key design interrelationships and dependencies are established before working on the individual detailed designs.
- **Formal user acceptance testing start agreed** Starting formal UAT with business users tends to be the final, full formal test phase before go live, with the expectation that go live is imminent and achievable. Prior to starting this phase, it makes sense to validate the readiness of all the elements to ensure that this test phase can complete within the allocated time period and meet the necessary quality bar.

Consider the various milestones that help with a true understanding of readiness, especially those that help you better understand when to start dependent tasks, such as feature complete, code complete, code frozen, business data validated in Dynamics 365, or solution ready for user acceptance testing, and determine what entry and exit criteria would help.

Design and change boards

A project with multiple parties involved in the design and delivery tend to have some form of design review board that ensures compliance with the enterprise IT policies and standards. This board also ensures that the proposed designs from the project (and especially design changes) are fit for purpose.

Effective design review boards (also called architecture review boards or design change boards) tend to have a good grasp of the overall solution blueprint. In addition to reviewing any individual design, they can ensure new designs and changes will operate within the boundaries of the solution blueprint and won't adversely affect other designs.

The design review board should also have representatives, not just from the project, but also from the wider business and IT, to help ensure the business case for the design is sound and the impact on related systems (integrations) and overall enterprise architecture are considered. Another characteristic of a successful design review board

is that they have minimum bureaucracy and their focus is on helping the project to move as securely and as fast as possible.

The process by which a design review board actually functions is as important as its constituents and roles and responsibilities. Design review boards should communicate the solution blueprint boundaries and design standards expected so projects can proactively make sure that they're working within the expectations.

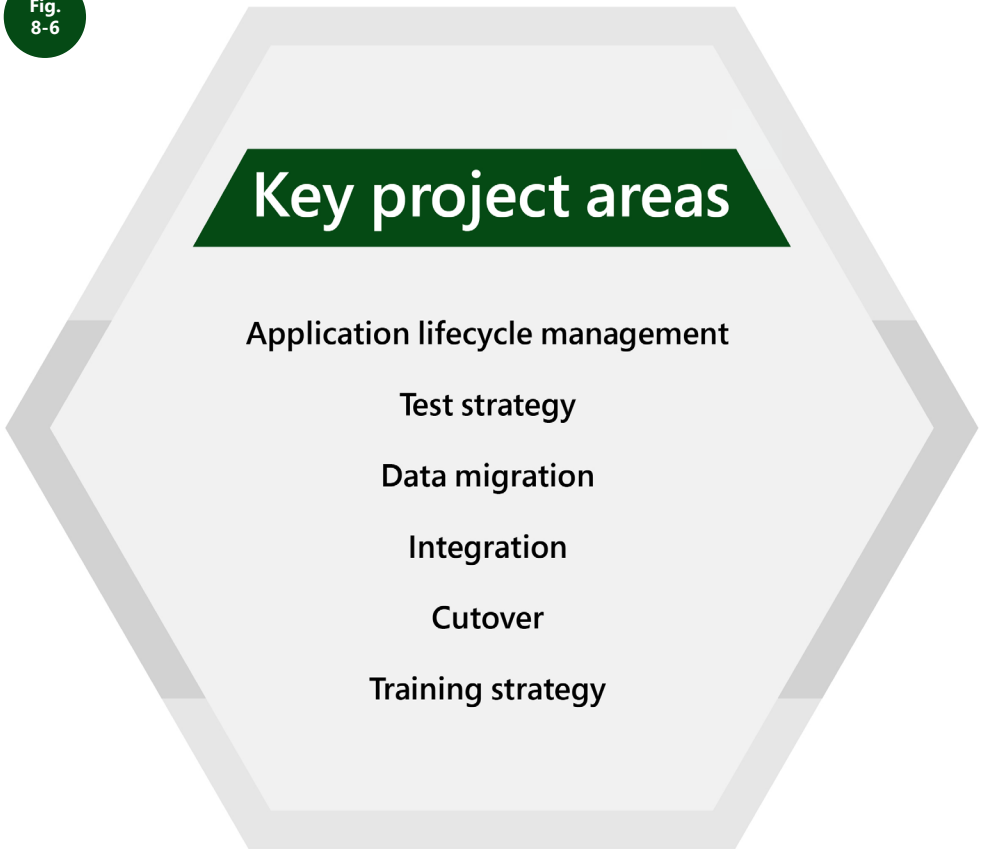
For the best project velocity, you should set an expectation that the reviews will be interactive and expected to be resolved within a single iteration. This requires the design review board to be pragmatic and well prepared. It also requires the project to have the discipline to provide sufficiently detailed proposed design information. This process needs to be organized and communicated as part of the project governance in the Initiate phase so you're not trying to establish these practices during the intensive pressures of a running project implementation.

Key project areas

Project governance is often thought to be confined to the classic areas we discussed in the previous section, but in Dynamics 365 business applications, we should consider the key activities that also need to have governance processes embedded. We see many projects where the strategies for the more specialist project delivery areas have their own processes and disciplines that don't always work in concert with the project approach or to the benefit of the overall project. These areas are often managed and driven by specialists with specialist processes and internal goals.

It's worth considering what governance is appropriate for these areas to ensure the project (and hence the business) derives the best value. In this section, we discuss the key areas of application lifecycle management (ALM), test strategies, data migration, integration, cutover, and training strategies (**Figure 8-6**).

Fig. 8-6



Application lifecycle management

Does the defined application lifecycle match the project methodology? For example, how will the configuration data be analyzed, defined, approved, secured, managed, and distributed to the different environments? Is this process also governed by the requirements of the project approach, and will it be designed to meet the expectations of the project timetable? The risks here are that ALM processes are sometimes considered as a technical topic and not rolled into the overall project governance, which creates mismatched delivery and project planning.

The build process (configuration, data, code) in a business application can be presented as a complex, technical, and jargon-filled process. Business stakeholders may retreat and let specialists take complete oversight. However, project leadership can get meaningful insight into the (necessarily technical) build process by devising a layer of governance that provides the necessary understanding and helps keep the project on track. For example, ask the following questions:

- Are the build requirements directly mapped onto the end-to-end business processes, so that any discussions related to the requirement can be well understood in that context by non-technical stakeholders?
- Are the right controls in place so that progress can be tracked and reported in a way that is meaningful outside of the technical teams?
- Do controls exist to ensure the right quality, standards, and compliance needs will be applied?
- Does the project have a practical and efficient method to raise

business process questions with the business area owners, and a way for the business to understand the system delivery of a business process?

A more in-depth exploration of the ALM and build process is available in Chapter 11, "Application lifecycle management."

Test strategy

A well-defined test strategy is a key enabler for project success. A high level of governance is required to ensure that the right test strategy is created and implemented, because multiple cross-workstream parties, business departments, IT departments, and disciplines are involved. When evaluating the suitability of the test strategy, in addition to the technical angles, consider how well some governance areas are covered:

- Does the right level of governance exist to ensure that the testing strategy is mapped to, and proportionate with, the project scope and the business risks?
- Is the business sufficiently involved in the test coverage, test case development, and approval of the testing outcomes?
- Is the project approach in line with the test strategy and test planning?
- Does the right governance exist to make sure the business processes are tested end to end?
- Is the data to be used in the various testing phases and test types sufficiently representative to fulfill the test objectives?
- Is the test coverage across the functional and non-functional areas adequate to help assure the business of safe and efficient operation in production use?

More details on defining a successful test strategy are in Chapter 14, "Testing strategy."

Data migration

For data migration, examine the related governance coverage to ensure that this process is well understood at a business level:

- Do you have the right level of business ownership and oversight on the types and quality of data being selected for migration?

- Is a data stewardship process in place to make sure that the cleansed and migrated data is kept clean during the project and in operational use?
- Will the data from the existing source systems be faithfully transformed so that it's still meaningful and fit for purpose in Dynamics 365?

Review the data migration strategy from a governance view to ensure the workstream is delivering to the needs of the project and the business and isn't becoming a silo with only technical oversight.

Integration

A common area missing governance is the management and ownership of non-Dynamics 365 systems involved in the integrations, because they tend to be managed by those outside of the project.

The cutover from the previous system to the new Dynamics 365 system is a time-critical and multi-faceted process.

Make sure that the business impact on either side of the integration is understood and managed. Confirm that the data exchange contracts are defined with the business needs in mind. Examine if the security and technology implications for the non-Dynamics 365 systems are properly accounted for.

You also need project-level governance on the definition and management of non-functional requirements, such as expected volumes and performance, and clear processes to resolve issues that may stem from the other systems in the integration.

Cutover

The cutover from the previous system to the new Dynamics 365 system is a time-critical and multi-faceted process. It requires coordination from multiple teams for the related tasks to all come together for a go live. You almost certainly need to include business teams that aren't directly part of the project. Therefore, cutover needs to be shepherded with a deep understanding of the impact on the wider business. Preparation for the cutover needs to start early in the project, and the governance layer ensures that the cutover is a business-driven process and not a purely technical data migration process. For example, early

definition and agreement on the related business calendar sets the right milestones for the data migration to work with.

The legacy systems shutdown window for the final cutover is typically short, perhaps over a weekend. For some cutover migrations, that window may be too short to complete all the cutover activities, including data migration. In such cases, the project team may perform the data migration as a staggered, incremental migration, starting with slow-moving primary data and frozen transactional data. This leaves a smaller, more manageable remainder to address during the shutdown. This needs careful governance, and the strategy needs to be decided early because the data migration engine needs to be able to deliver incremental data loads. You should also carefully consider what business activities you may need to throttle or perform differently to reduce the number and complexity of changes between the first migration and the final cutover. The changes need to be meticulously recorded and registered (automatically or manually) so that they can be reflected in the final cutover.

Training strategy

Most projects have plans to train at least the project team members and business users. All too often though, training is seen as a lower priority. If any project delays put pressure on timelines or budget, training can be one of the first areas to be compromised.

This trade-off between respecting the go live date and completing the full training plan can be easier for the implementation team to rationalize because the team is aiming for a go live and the risk of poor training can be seen (without sufficient evidence) as manageable. The worst consequences of poor user training are felt in the operational phase.

You can mitigate this (mostly unintentionally) unbalanced trade-off with the following actions:

- Clearly defining and communicating the implications on project success in operational use of poor training delivery.
- Making the review and approval from the business reflect people readiness. This check should be part of the entry criteria into user acceptance testing and be a meaningful part of the go/no go for production use.

The test for people readiness needs to be meaningful; it should be an evaluation of the effectiveness of the training, and not just that it was made available. The test should be equivalent to assessing whether enough people in key roles can conduct their critical day-to-day business process safely and efficiently using the Dynamics 365 application and related systems.

More details on how to put together a good training strategy are in Chapter 19, "Training strategy."

Project plan

Project plan analysis is where the outcomes of a lot of governance topics become visible. The effects of good governance are felt on a project by noting that the project plan is resilient to the smaller changes and unknowns that are a reality for all business application projects. Poor governance, on the other hand, manifests itself as continuously missed targets, unreliable delivery, and repeated re-baselining (pushing back) of the project plan milestones. The key is for the project planning process to have its own governance mechanisms to avoid poor practices, detect them early, and provide the agility and insights to fix and adjust quickly.

When determining how the project plan should be constructed, the project plan should be able to demonstrate the following:

Is the project plan designed to cover all the critical areas?
Does the plan cover (at least at a milestone level) all the key areas of activity? If not, key dependencies and effort will be missed.

Is the project plan actionable?
A project plan should be able to fulfil one if its primary purposes of directing a project. If that isn't the reality on the project, you should look at how to restructure it so that it's the main driver of project activity at least at the milestone level.

If a plan is thousands of lines long, it's unlikely to be directly actionable, and will probably be in a state of continuous update and adjustment.

Similarly, if the project workstreams aren't paying any attention to the plan, it's a clear sign that the project plan may not be actionable. Actionable project plans help direct the project teams to deliver the right activity at the right time and reduce uncontrolled activities.

Projects using agile practices may be using some form of prioritized backlog management for much of the build and test activities, but there is normally still a high-level milestone plan to allow for the communication and management of related tasks that aren't covered by the backlog.

Whatever methodology is in place, the plan should be regularly evaluated to confirm that the teams are taking their direction from the plan.

Is the project plan up to date, accurate, and realistic?

Compared with the traditional on-premises business application projects of a few years ago, today's cloud-based Dynamics 365 implementations tend to move faster with shorter durations. A project plan that isn't up to date is ineffective.

Projects should institute thresholds for how out of date a project plan is allowed to become. A project plan that is inaccurate in terms of the estimated effort (and remaining effort), duration, and dependencies will promote the wrong decisions and allow risks to remain hidden, and ultimately give an inaccurate picture of the project status.

For a project plan to remain accurate, it needs a robust task estimation process and a reliable and easy means of tracking tasks in progress. Consider how to make this possible structurally and with efficient processes.

Does the project plan give a clear indication of what tasks are critical?

Business application projects have many moving parts and many specialist areas. The project must be able to accurately derive the critical path so that project leadership can focus their attention on those tasks. For agile projects, agile crews prioritizing the backlog and managing the critical path within short-duration sprints and iterations can provide the necessary insight.

Accountability for outcomes

Project planning should promote the clear identification of accountability for the outcomes (both at the workstream level and overall). Often, projects create a RACI (responsible, accountable, consulted, informed) matrix at the start of the project, but this may not provide the right level of accountability for all areas. Additionally, this matrix isn't always referenced during the project implementation. Asking the question via the project plan if the accountability for the outcome (not responsibility) is clear helps keep the accountability fresh in everyone's mind. It keeps the project more agile with faster decision-making.

Status reporting

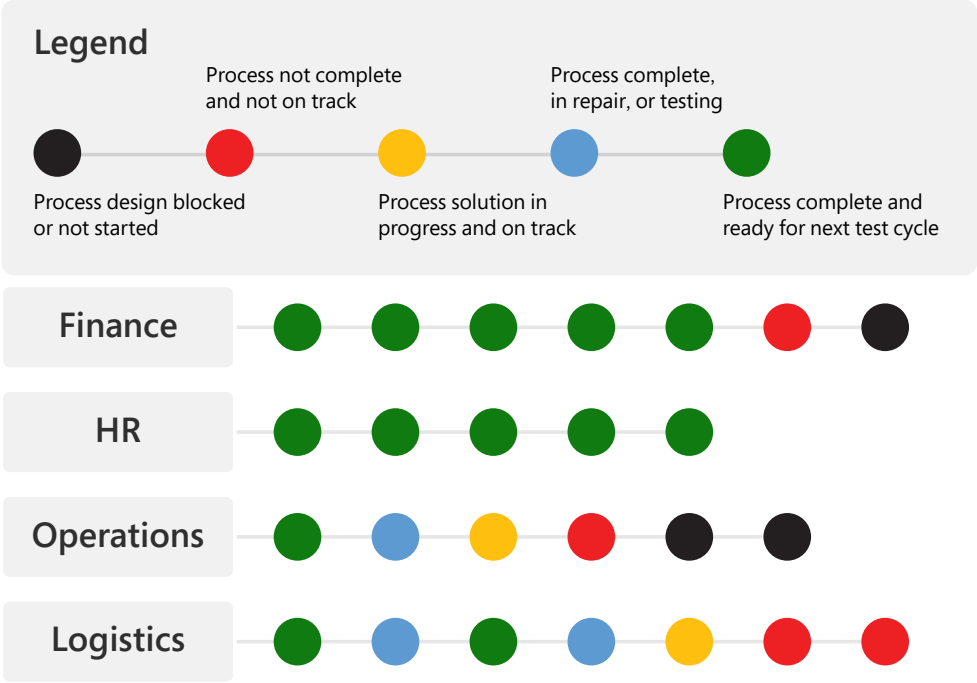
A well-constructed project plan facilitates accurate project status reporting. However, it needs to be deliberately designed into the project plan with the right dependencies, estimated effort, and milestones so it can be easily extracted from the plan. This means that the project should be deliberate in the detail to which tasks are scheduled so dependencies and milestones are created with this in mind. Because status reporting often has milestone-level status as a key indicator, the meaning of a milestone's completion must be explicitly defined so the implications are clear to the intended audience.

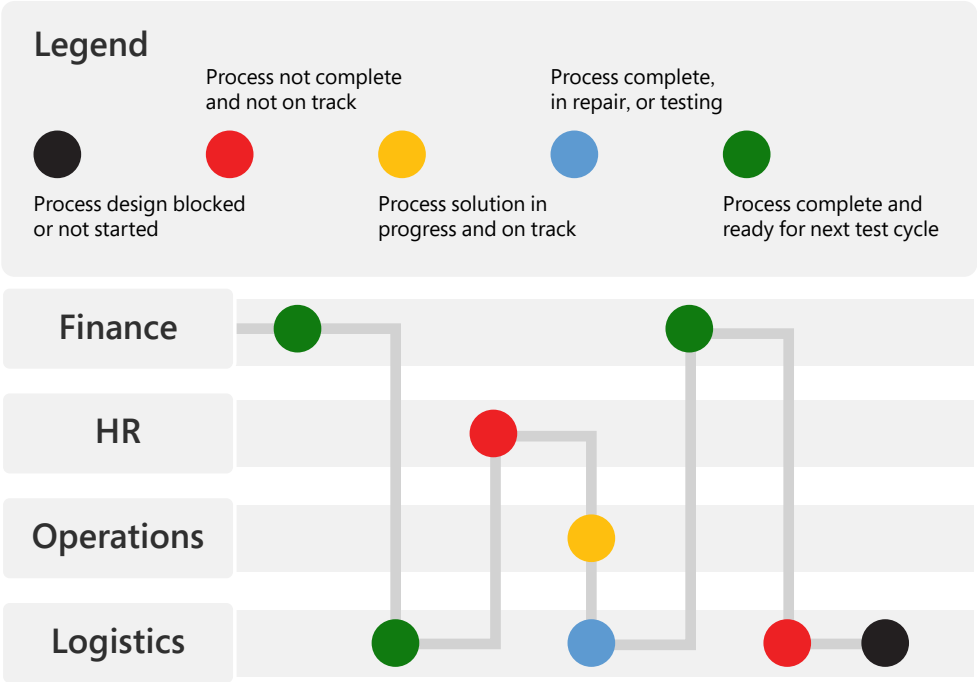
Project plans also need a regular and easy-to-use process to keep task progress updated with an accurate status. You should also define a uniform and reliable method for units of progress so you get the same quality of updates from all teams. For example, tasks defined as 80 percent complete often under-estimate the effort to complete the remaining 20 percent; instead, consider reporting the remaining effort in hours for a given task.

Some projects, especially ones using more agile practices, may use alternative or additional analysis and presentation methods such as a backlog burndown, remaining cost to complete, or earned value, but these principles apply nevertheless and they all rely on having accurate underlying data on actual versus expected progress to date and the remaining effort expected.

Fig. 8-7

Business process heat map example





A useful way to report the status of a business application project is to present a functional heatmap view (**Figure 8-7**), which shows the readiness of the key processes for the next major milestone (such as sprint review, SIT, UAT, or go live). The functional heatmap is a business process perspective of the project plan status.

Status reports can also include status from other project controls such as risks and issues, sentiment surveys, or resource availability in prior months. These aren't directly derived from a project plan and can provide a useful supplementary view. However, it's important to ensure that actionable conclusions are extracted from these alternative views; these conclusions should generate tasks that can be measured, planned, allocated, and subsequently reflected at some level in the project plan.

Project feedback loop

Project status should provide a feedback loop to the project stakeholders, project management, and the project team to adjust the project based on the findings. This should be a defined process in the overall project governance to create the discipline to regularly

review the status and generate explicit actions for the project that directly address the status findings.

The project status should not be the only source for this feedback loop—consider all the other controls and procedures in the project that can generate actionable information, if only there were a systematic mechanism defined to diligently listen and extract the actions. For example, updates from daily stand-ups, design reviews, sprint playbacks, or even informal risk discussions can help provide useful feedback.

In summary, create a culture from your governance processes that actively recognizes the feedback loop and translates feedback into actions.

Conclusion

The primary intention at the start of this chapter was to provide an overview of the importance of good project governance and provide guidance on how to assess the effectiveness of your existing or proposed governance model.

We then moved on to describe how many of the critical issues on a Dynamics 365 project that initially may manifest themselves as technical issues actually have their roots in failures of project governance. We also saw that the governance requirements for a Dynamics 365 business application needs to address the fact that the project is essentially delivering the capability to perform business processes using the application. So the governance model must include strategies and techniques that can help realize the project goals through the right engagement with business users and business processes.

We discussed how defining a project organization model that has an engaged leadership and facilitates rapid and accurate communication is more aware of the reality of the project. Similarly, project organizations that enable the right accountability, naturally encourage cross-collaboration, and reflect the complexity and effort of the project are much more likely to have good velocity and be successful.

We also discussed how the approach is much more than a technical methodology, and that the approach must be practical and tailored to the specific needs of a Dynamics 365 business application and to the customer's project and business needs and constraints.

We showed that the mere presence of the usual governance structures doesn't assure project success. We can evaluate the effectiveness of the governance model in very practical ways and with a focus on the specific needs of Dynamics 365 projects.

We also brought awareness to how a governance perspective on the specific technical implementation strategies (such as test and build strategies) can help a Dynamics 365 project achieve better, faster results. We proposed that governance should be an organic part of the strategy. These strategies shouldn't be considered as being purely subject to technical oversight and control.

Lastly, we discussed how a good project plan and planning process will expose the outcomes of a lot of these governance topics. Project planning shouldn't just be an overall status reporting tool, but be deliberately engineered to gain early and deep insights into the effectiveness of the different elements of the project work and allow rapid responses to the deficiencies.

You can use the key objectives of good project governance to judge the overall suitability of a given governance model. Ideally, you should test a proposed governance model against these objectives during the Initiate phase of the project and throughout, but it's never too late.

In summary, project governance for Dynamics 365 projects should be a framework that permeates all aspects of our projects, not just the typical areas that were historically or conventionally regarded as the domain of governance. Good project governance isn't a set of generalized bureaucratic or administrative procedures to follow—it's about creating a Dynamics 365-specific governance culture of driving the project to reliably and efficiently deliver the project goals.

Checklist

Project goals

- [ ] Ensure goals are clear, realistic, mapped to actions and measures, correctly translated into project deliverables, and regularly monitored throughout the project lifecycle.
- [ ] Align goals and ensure they are sponsored by relevant stakeholders across the organization, and properly communicated and understood by the implementation team.

Project organization structure

- [ ] Align business streams with functional workstreams for better efficiency and structure, and make sure the business stream is directly involved in the project.
- [ ] Secure a strong executive sponsorship and active engagement from senior business stakeholders within each business stream.
- [ ] Ensure cross-team collaboration in which members of each workstream are involved in other workstreams to avoid working in silos.
- [ ] Plan and budget the project resources in proportion to the effort and complexity of the project.
- [ ] Define accountability and responsibility at the project leadership level. Each stream identifies an owner with autonomy who is empowered to make decisions.
- [ ] Identify relevant project roles and areas of ownership and assign them to team members with the right expertise and experience.

Project approach

- [ ] Analyze, review, and confirm that your chosen implementation methodology works well with the business and project constraints and circumstances.

Classic governance

- [ ] Establish an effective steering group that understands enough of the project details to actively steer the project based on meaningful, actionable, and evidence-based information.
- [ ] Create a risk register with meaningful and actionable risks that align with project priorities and are actively addressed.
- [ ] Implement stage gate or milestone-driven planning and reviews as checkpoints to better track and communicate implications of the project status.
- [ ] Implement design review boards to ensure new designs and changes operate within the boundaries of the solution blueprint and don't adversely affect other designs.

Case study

Project governance as a critical success factor

A large enterprise that was using legacy home-grown applications for managing their sales and customer service business processes decided to embark on a new cloud application, Dynamics 365. This was a mission-critical and complex implementation for this company; it would affect their key business of providing customer service to their large enterprise customers, and was the first cloud deployment within a traditional IT managed landscape. The plan was to release a minimum viable product (MVP) within a timeline of six months and follow a phased approach for their users to adopt the new solution smoothly and without too many challenges.

Although the implementation team ensured that the scope was aligned with business priorities and clear project goals were agreed upon, no clear processes were defined for the following key aspects of the project:

- Realistic timelines for the activities planned within the engagement
- Understanding of technical and business complexity
- Well-defined project organization structure
- Assignment of adequate and qualified resources
- Effective project updates for the steering group
- Full understanding of the product's out-of-the-box capabilities, thereby avoiding unachievable product requirements

As the project began, complexities surfaced early in the cycle. However, the team didn't revisit and realign their activities. This resulted in the following impacts:

- Poor quality of deliverables
- Mismatch in customer versus partner understanding of requirements
- High number of defects identified during initial testing cycles
- Incomplete implementation of requirements due to lack of skilled resources on the project
- A continuously shifting end date for final go live, resulting in a delay of 6–8 months
- A complete lack of trust between the customer and partner

After several discussions between customer and partner stakeholders and a high impact on cost and morale, the team agreed to clearly define a project organization structure and an approach to continuously monitor and manage the overall project. The following activities were incorporated as project governance processes were defined:

- Considering the new technology, training needs for both the customer's business and IT teams were identified so they could understand both the SaaS world and Dynamics 365 product capabilities.
- An architecture board was established that was responsible for diligently reviewing architectural decisions coupled with the gap analysis. For any gaps identified, this board provided input to the change control board to ensure that a change management approach was followed without any further impact on timelines.
- Both industry and product architects were assigned to the project to avoid any further lack of domain understanding and perform a clear fit gap analysis.
- The steering committee was empowered by the customer's senior management to take the relevant and necessary decisions through the project.
- A key challenge that impacted the deliverable quality was that several activities were planned almost parallel to each other and were to be done by the same project team. As part of the project reorganization, activities such as performance testing and vulnerabilities testing were outsourced. This allowed the existing team to stay focused on their core functional scope.

As the project team implemented these various controls and processes, they were able to redefine their project timelines and have now gone live within that timeline. This has led them to ensure that they can apply and tailor these learnings to other projects to avoid similar high-cost impacts.

In conclusion, project governance must not be thought of as an afterthought. It must be set up from the beginning to work as a flexible backbone for the entire project.

As illustrated by this and other case studies, project governance is the most critical factor that entails all the key elements to make a project successful.

9

Guide

Environment strategy

Introduction

Defining your environment strategy is one of the most important steps in the implementation of your business application.

Environment-related decisions affect every aspect of the application, from application lifecycle management (ALM) to deployment and compliance.

At a fundamental level, a good environment strategy is about obtaining the right balance with multiple layers of compliance and security, productivity, collaboration, isolation, performance, and maintainability. A strategy to manage environment provisioning and access—and controlling resources within them—is key to:

- Securing data and resource access
- Organizing and structuring solution components
- Governing and managing capacity

In this chapter, we explore several deployment and transition scenarios. By the end of this chapter, you should understand the factors to consider when defining your environment strategy, and the available patterns that will enable you to confidently deliver an environment plan for your applications.

Environment strategy

Environments are containers that store, manage, and share your organization's data. They also store the data model, application metadata, process definitions, and the security constructs to control access to data and apps.

A successful environment strategy defines the core principles and policies for creating, granting access to, and decommissioning environments of different types, along with the necessary governance process. These policies should provide a consistent approach for centrally deployed, IT-driven implementations and for citizen-developed applications for individuals or small business teams.

A data strategy is likely to inform or impose constraints on an environment strategy. For example, an organization could build a unified data platform that different apps and services consume, or there could be a fragmented legacy data estate that must be accounted for in the cloud environment strategy. Chapter 10, "Data management," delves into the patterns and scenarios that are relevant to environment planning.

To get started with creating an environment strategy, ask the following questions:

- **Where** are my users physically located?
- **What** are my organization's data compliance and residency requirements?
- **What** types of environments are required, and when?
- **What** are the different roles (such as developers, testers, admins, business analysts and trainers), and which environments do they need access to?
- **What's** the access procedure for an environment?
- **Which** apps will be installed in the environment?
- **Will** the solution require more than one production environment?
- **Which** services and third-party applications do I need to integrate with?

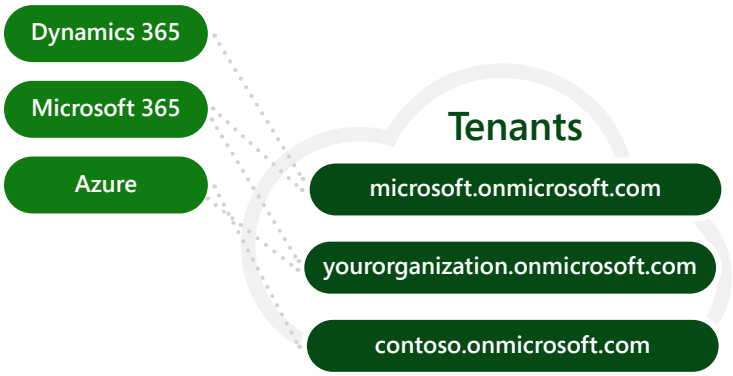
It should be apparent from the sidebar questions that environment strategy is not the responsibility of a single person or team. It can involve IT admins, the architects responsible for the design and implementation of the solution, the information security and compliance teams, and the business stakeholders.

Environment-related decisions are hard and expensive to change. As we explain throughout this chapter, environment strategy can affect how solutions are implemented. This exercise needs to be done at the start of the project—even before beginning implementation—and is not something that should be left to the end as a part of a go-live plan.

Tenant strategy

Understanding tenants is fundamental to defining an environment strategy for your deployment. A tenant is a logical structure that represents your organization. Every cloud service to which an organization

Fig. 9-1



subscribes will be associated with the customer's tenant. A tenant is not dedicated to a specific service; it has subscriptions for Microsoft services, such as Exchange, Dynamics 365, Power BI, and Azure. Every tenant is given a Microsoft domain—such as yourorganization.onmicrosoft.com—but admins can also associate their own custom domains with the tenant. For example, as shown in **Figure 9-1**, contoso.onmicrosoft.com can be associated with contoso.com and other domains owned by the organization. You can also federate access to your tenant via your organization's on-premises Azure Active Directory (Azure AD) to enable single sign-on (SSO) across on-premises and cloud services.

A tenant provides the highest level of isolation for a cloud service, as it represents an organization and differentiates one customer from another. Tenants never share resources or licenses because they're meant to represent an organization and not a department or business unit. (In some cases, subsidiaries could share a tenant with the parent organization, depending on the organization structure, and the tenant model might mirror that structure.)

Isolation approaches also may differ between services. For example, as shown in **Figure 9-2**, you could have different subscriptions in Azure within the same tenant—and each subscription could host different applications and have a different set of users—because Dynamics 365 and the Microsoft Power Platform have environments that isolate data and manage user access.

Fig. 9-2

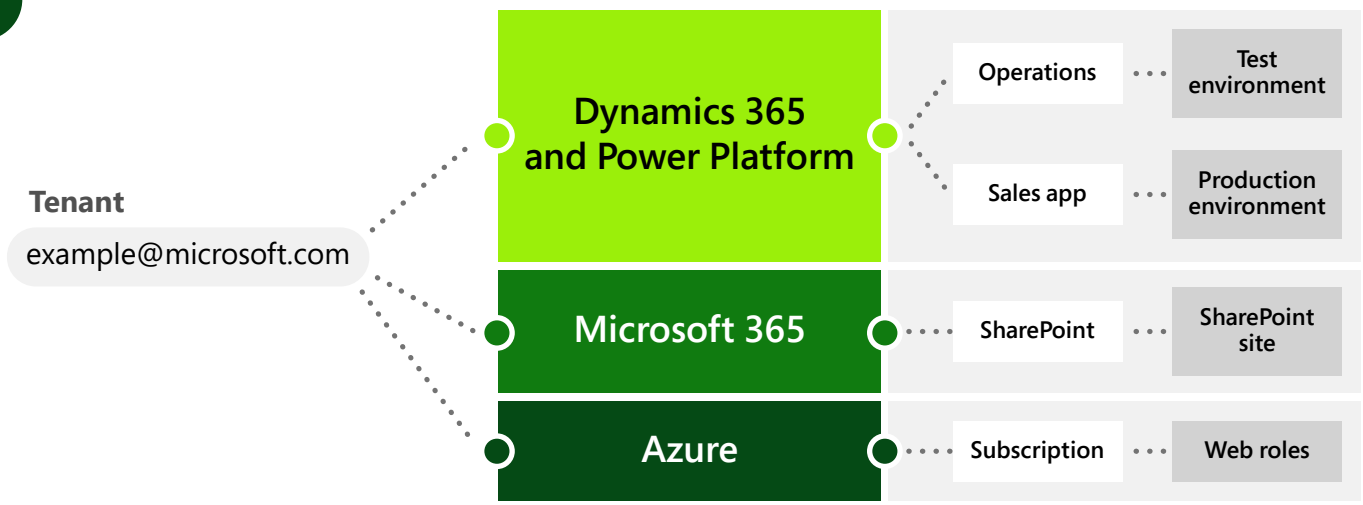
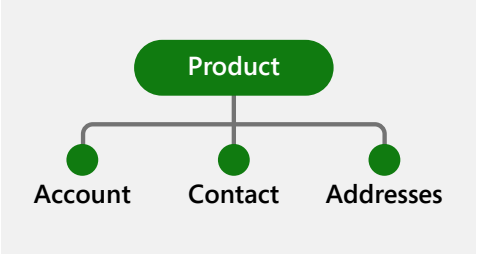


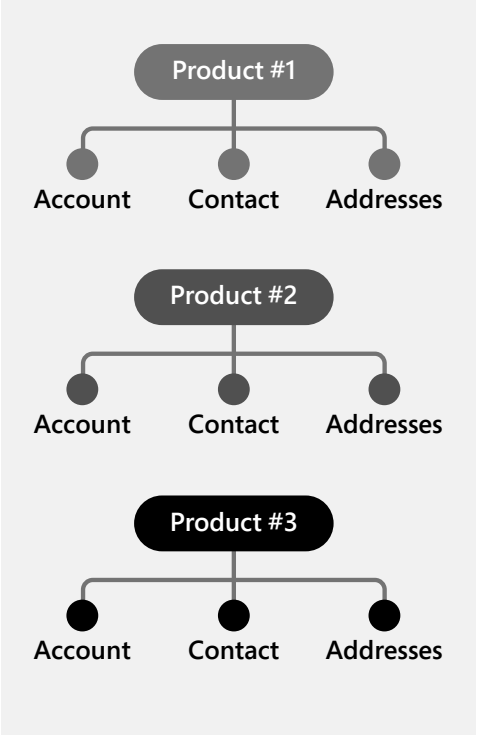
Fig. 9-3

Global single tenant vs. global multitenant setup

Global single tenant



Global multitenant



Your environment strategy should provide a balanced level of isolation to meet the security and compliance needs of your organization. It also should take into consideration the administration, ALM, and collaboration needs of the project. To define the right tenant and environment strategy for your organization, it's necessary to understand the controls that are available for each service to isolate code, configuration, data, and users. It's also important to note that the isolation provided by services doesn't necessarily reflect the underlying infrastructure or design of a cloud service. A separate tenant doesn't mean it's a separate server farm—and having a separate environment doesn't give you a different front end.

Global single-tenant setup

Using a single Microsoft tenant to represent the organization in the Microsoft cloud (**Figure 9-3**) is a common setup. It provides unified administration for user access and licenses, enables seamless integration between services, and lets the organization share tenant resources.

All Dynamics 365 and Power Platform environments will be a part of the same tenant. There could be different apps deployed in different environments, or they could have different users for each app and environment, but all would belong to the same Azure AD that is associated at the tenant level. Using the Multi-Geo setup, you could create environments in different countries or regions to meet your compliance and application needs, but they would remain a part of the same organization-wide tenant. Sovereign cloud deployment requires a separate Azure AD and might have additional regulatory restrictions that limit your ability to create environments in different countries or regions.

Let's examine some of the pros and cons for a global single-tenant setup.

Pros of a global single-tenant setup:

- It aligns to the concept of tenant and the way cloud products are licensed.
- It allows sharing of licenses and quotas across the tenant per service. (Microsoft Dataverse storage is one such example.)

- It provides a central place for IT admins to manage the services, users, and licenses.
- It is easier to deploy organization-wide IT policies such as data loss prevention (DLP) and conditional access.
- It is simpler to manage service account configurations and connections when moving from one environment to another in the same tenant.
- It facilitates easier and safer integrations between tenant services, without the need to elevate privileges to cross service boundaries. (For example, Power Platform and Exchange Server-side sync is only possible within the tenant.)

Cons of a global single-tenant setup:

- A tenant can only be associated to a single org Active Directory forest. If there are independent subsidiaries using a separate Active Directory forest, they can't be added to tenant, but can use alternatives such as Azure business-to-business (B2B) for guest-user access.
- In a single-tenant setup, you can create tenant-wide policies. All environments, including the production environment, are governed by the same policies. Creating exceptions for trying out new services or testing might require the team to go through multiple information security hoops. (For example, a proof of concept or pilot requiring a different Active Directory conditional access policy will attract scrutiny from security.)

Global multitenant setup

Usually, the tenant structure for the Microsoft cloud software as a service (SaaS) mirrors the structure of the organization. But some organizations operate as separate business entities in a country or region—even though they might be part of a single umbrella brand. In such cases, an organization might adopt a multitenant setup (**Figure 9-3**), as each business entity might have its own contractual needs, regulatory constraints, and licensing agreements.

Alternatively, organizations can have separate tenants for development, testing, and live systems. This primarily supports the information security requirements to isolate data and keep users who aren't involved in production from accessing production systems. Still, using separate tenants is generally not recommended because there could be data and access isolation at the service level to meet information security requirements.

(For example, you might use security groups on environments to limit access.)

User accounts, identities, security groups, subscriptions, licenses, and storage can't be shared among different tenants. So, using separate tenants might require replication of all other services or creation of stubs for external systems—per tenant—for all the integrations. Also, duplication of licenses and quotas across tenants may be needed to support ALM and performance testing. Admins would have the additional responsibility of ensuring that all the policies and settings between tenants are in sync to avoid deployment failure between tenants. Some organizations do go down this path because of their historical setup or a lack of understanding of the controls available to restrict access at the service level.

Let's examine some of the pros and cons for a global multitenant setup.

Pros of a global multitenant setup:

- It works when different subsidiaries are fully independent, with different systems to manage user identities in separate domains, and have separate licensing needs.
- It provides the highest level of isolation that will simplify the security approvals but is not generally required.
- It is easier to manage and monitor the cost per tenant without the need for internal cross-charging and tracking usage of shared resources on a single tenant. (For example, storage is shared on a single tenant.)

Cons of a global multitenant setup:

- Licenses can't be shared across different tenants, which means that storage, API add-ons, and other tenant-level quotas must be purchased for each tenant, and you may have to purchase separate licenses for the same user to access separate tenants.
- Overhead to maintain tenant-level Active Directory and DLP policies consistently across tenants can lead to surprises during production deployment.
- Service-level admin actions, such as ability to copy an environment, may not be available across tenants, which can make testing or troubleshooting difficult.
- The build automation and continuous integration and continuous

delivery (CI/CD) pipelines that span multiple tenants can be more complicated and might require manual intervention, especially when managing connections to the service.

- You may have to purchase a significant number of licenses for conducting testing. With load testing, for example, you can't reliably simulate a load from 1,000 users using five test-user accounts.
- If you're using capacity add-ons, you will have to make duplicate purchases for each tenant to develop and test your solutions.
- Integrations with other services, such as Exchange, can't be done across tenants, which means potentially purchasing licenses for other Microsoft services for each tenant.

Overall, managing your ALM process across several tenants brings unnecessary complexity to licensing and technology, and delivers little value in terms of security, because you might already have the necessary controls at the service level to limit access to business data.

Key factors affected by an environment strategy

In this section, we explore the key factors directly affected by an environment strategy, including security and compliance, ALM, operations, and functional and nonfunctional aspects of the solution. Considering these key factors will help you define an environment strategy that meets the needs of your organization, and set the direction for the future growth of these applications to meet evolving business requirements.

Compliance

Security and compliance are critical considerations for an environment strategy. Each organization needs to ensure that data is stored and processed in accordance with local or regional laws, such as data-residency requirements for a specific region. For example, the European Union (EU) enforces the General Data Protection Regulation (GDPR) for EU residents, as well as its citizens outside of the EU.

At the onset of the project, your organization must determine your environment's compliance requirements. These can vary widely depending on the industry, regulations, business type, and user base, and will need to be approved by your internal security and compliance teams.

The most common elements affecting compliance are:

- The physical location of the environment(s), which usually determines data residency
- The specific features and apps in the scope and their data-processing requirements
- The encryption (at rest and in transit for confidential information)
- The disaster recovery procedures to restore the service and data
- The data-retention policies, and the ability to back up and restore data
- The personally identifiable information (PII) standards for data protection, and ensuring the service is compliant with the regulatory requirements for the industry and region

Application design

The environment strategy can affect the application design. Conversely, the needs of an application can drive the environment requirements, and it's not uncommon for IT systems within an organization to reflect the actual structure of the organization. Depending on your organization's strategy for data isolation, collaboration, and security between different departments, you could choose to have a single shared environment or create isolated environments. For example, a bank might allow data sharing and collaboration between commercial and business banking divisions while isolating the personal banking division, which reflects the bank's application design and environment strategy.

Environment policies should not be created in isolation. Creating a generic organization-wide policy, such as "Every application should use a separate environment" or "All applications should share a single environment"—without considering the business requirements or understanding the underlying data and integration dependencies—could lead to unnecessary complexity and fragmentation.

The data store for an application and the supporting business process

plays a key role in the environment decision. If multiple apps for different departments can benefit from leveraging each other's data, a single environment with integrations can improve consistency and collaboration. The user experience can be tailored via dedicated apps for different personas and secure data access using the security model.

For sophisticated systems that need to use multiple environments and potentially sync data between them, it could be technically possible, but such patterns require careful consideration of performance and API capacity—and are best avoided.

App integrations also play a key role, as multiplexing with different environments may not work. For example, a user may not be able to simultaneously sync emails to several environments.

Performance

Microsoft cloud services provide a high degree of scalability and performance. Based on considerations such as network latency, firewalls, network traffic monitoring, organizational proxies, and routing by internet service provider (ISP), globally distributed users can experience higher latencies when accessing the cloud. This is why we recommend creating a latency analysis matrix (**Figure 9-4**) for solutions that have a globally distributed user base.

This exercise gives a fair idea of how the network latency will affect the user experience based on the location of your environment. This information can be used to make a balanced choice so most users have an acceptable level of latency, and application design can be optimized for users in high-latency locations. For example, using

Fig. 9-4

Environment location	User location	Device	Network	Latency	Bandwidth
North America (NAM)	Canada	Browser	Corporate network	80 ms	6.4 Mbps
North America (NAM)	Amsterdam	Tablet	Corporate Wi-Fi	120 ms	8 Mbps

lightweight forms or reducing the number of records shown on a page can enable faster loads.

Also, environment types should be chosen carefully, based on the workload and the capacity needed. Using trial environments, for example, will not deliver the same level of performance as live production environments, so it may not be suitable to develop or test the solution on a trial.

Scalability

Scalability of the SaaS platform is a critical consideration for business applications. Traditionally, scaling up in on-premises deployments was about adding more servers or more CPU, memory, or storage capacity to existing servers. In a cloud world with elastic scale and microservice architecture, the server could be replaced by an environment and the compute and data transfer units by the API capacity. (This is just used as analogy—it's not a one-to-one mapping where one environment corresponds to a server in the SaaS infrastructure.)

The scalability parameters for a SaaS could be:

- How many parallel connections or concurrent users can you support with one application or in a single environment?
- How many API calls is a user entitled to make?
- What are the limits on workflow or code executions?

The following is a potential SaaS cloud interpretation of vertical and horizontal scalability.

Vertical scalability

Organizations commonly operate single environments supporting thousands of users, and each user has a defined API entitlement based on the license type assigned. The environment's storage grows as more users and applications store their data. Dynamics 365 SaaS services are built on a scalable cloud infrastructure that can store terabytes of data to meet the requirements of large enterprises. When it comes to workflows automation, each environment can have any number of Microsoft Power Automate flows, each with thousands of steps

performed per day, which can be further scaled using appropriate license types and add-ons. There are no hard limits on the number of apps you can have per environment, but you can't have multiple first-party apps of the same type in the same environment.

Horizontal scalability

With horizontal scalability, organizations can have several separate environments, with any number of Power Automate flows on the tenant. There are no native sync capabilities between environments, and you still need to take license entitlements into consideration, especially when it comes to tenant-wide storage and API entitlement.

The effort to maintain the solution is directly proportional to the number of environments involved.

Maintainability

Maintainability measures the ease and speed of maintaining a solution, including service updates, bug fixes, and rolling out change requests and new functionality.

If you have several applications sharing common components in the same environment, you should consider maintainability when upgrading or deploying a new release. It's also important to invest in automation testing so you can run regression tests and quickly identify any dependencies that could cause issues.

The effort to maintain the solution is directly proportional to the number of environments involved. For example, testing a release wave or analyzing the impact of deprecations is easier when there is just one production environment with the Dynamics 365 Sales app and the Dynamics 365 Customer Service app, compared to a solution that uses different production environments for the Sales and Customer Service apps.

ALM

ALM includes the tools and processes that manage the solution's lifecycle and can affect the long-term success of a solution. When following the ALM of a solution, consider the entire lifespan of the solution, along with maintainability and future-proofing. Changes to your environment strategy will directly affect the ALM (and vice versa),

but it's important to be clear that environments are not the repository of your code and customizations.

Environments can be used to separate business functions or for different purposes, such as development, testing, and production. Typically, at least three environments are necessary to support a mature release-management process.

Types of environments

Production environments are meant to support the business. By default, production environments are more protected for operations that can cause disruption, such as copy and restore operations. Sandbox environments can be used to develop, test, and delete as required.

Purposes of environments

- **Development** One or more development environments are usually required, depending on the customization requirements and time available. Development environments should be set up with proper DevOps to allow for smooth CI/CD. This topic is covered in more detail in Chapter 11, "Application lifecycle management."
- **Quality assurance (QA)** Allows for solution testing from both a functionality and deployment perspective before the solutions are given to the business teams in a user acceptance testing (UAT) environment. Only managed solutions should be deployed here. Depending on project requirements, there can be multiple testing environments, including regression testing, performance testing, and data-migration testing.
- **UAT** Generally the first environment that business users will have access to. It will allow them to perform UAT before deployment.
- **Training** Utilized to deliver training. It allows business users to practice in a simulated environment without the risk of interfering with live data.
- **Production** The live system for business users.

Citizen development

One of the key value propositions of the Power Platform—the underlying no-code/low-code platform that powers Dynamics 365 Customer Engagement apps—is that it enables people who aren't professional developers to build apps and create solutions to solve their own problems.

Central IT still has governance responsibility and creates guidelines to secure data with DLP policies. But business users should be able to build apps and automate their work while accessing data in a secure way. This organic growth of applications by business users facilitates digital transformation at a massive scale. Thousands of organizations have adopted this "citizen developer" philosophy to quickly roll out hundreds of apps across the organization, creating a community of engaged employees who are helping realize the vision of an agile business that can evolve to meet customer needs in days or weeks instead of months or years.

The environment strategy that enables this kind of organic growth will be different from a traditional IT-developed solution. It's crucial to deliver an agile and automated process where business users can request environments to enable the maker experience and connect to data in a secure way while conforming to the standards for data modeling and application design set by the organization.

When planning your environment strategy, consider any future phases and rollouts of the solution, as well as changing requirements.

Future-proofing

Future-proofing is the process of developing methods to minimize any potential negative effects in the future. It can also be referred to as resilience of the solution to future events.

Your environment strategy needs to take into consideration any future phases and rollouts of the solution, and allow you to deal with changing requirements and build a solution that will not limit the business or take away necessary control. As an example, consider country-specific needs, as well as variations to and deviations from the standard process.

Environment app strategy

If you're deploying multiple business apps on the Dynamics 365 platform, you will need to decide on an environment app strategy. Should all apps be deployed in the same environment? Or should each app have an environment of its own?

There isn't a standard answer or blanket approach that will work for all apps within your organization. The best approach for you is the one that facilitates collaboration and cross-pollination of information between apps—while also reducing data silos and meeting your organization's security, compliance, and data protection requirements.

Multiple-app environment

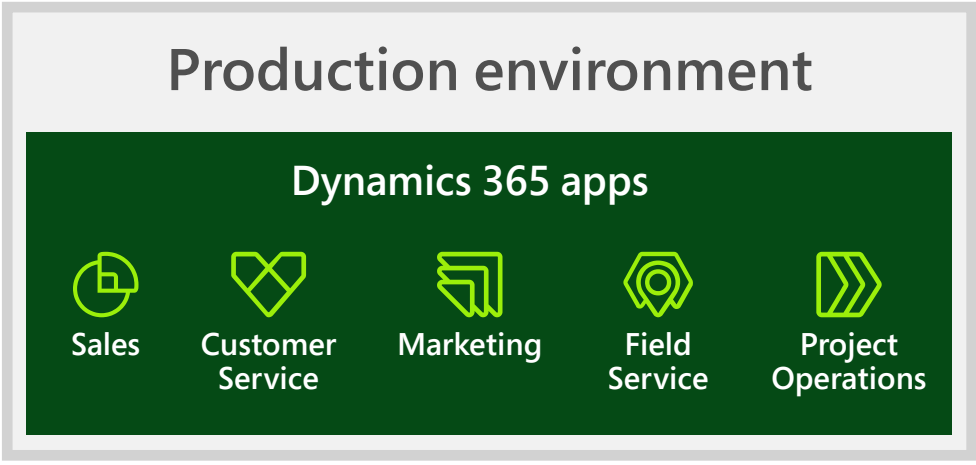
In a multiple-app environment scenario (**Figure 9-5**), a single production environment is used for multiple apps. For example, the production environment might have the Dynamics 365 Marketing and Sales apps to enable collaboration between the marketing and sales teams, and facilitate a quick transfer of qualified leads to sales representatives. Similarly, having the Sales and Customer Service apps in the same environment gives the sales team insights into customer support experiences, which could affect ongoing deals.

With multiple-app deployment, the app data, security models, and data models are in a common repository, allowing the apps to reuse any integrations. The security model design will be used to limit access to data for each app, with the user experience defined by the individual app. For example, the Sales and Marketing apps might use the same lead table, but security roles and the app definition will control access to records, fields, and processes.

Let's examine some of the pros and cons for the multiple-app deployment model.

Fig. 9-5

Multiple-app environment



Pros of a multiple-app deployment model:

- It enables stronger collaboration between business teams by decreasing data silos.
- It reduces the number of environments to manage, as well as the amount of effort needed for platform updates.
- It allows reuse of integrations, and its simpler design lowers API consumption while

avoiding the need for data sync across environments.
- It simplifies reporting and the business intelligence architecture.

Cons of a multiple-app deployment model:

- Its security model could become complex, depending on access restrictions.
- Its ALM and release process needs to be coordinated for each app and will require full regression testing when changing shared components, making automation testing even more critical.
- Its capacity utilization for individual apps is more difficult to track.
- Its security, data models, or integrations, if poorly designed on one app, can affect other apps.
- It might have limitations on deploying several apps of same type in a single environment. (For example, you can't have several Power Apps portals with the same template in a single environment.)
- It can't be used if you need to segregate the environments for a globally distributed user base due to performance, compliance regulations, or process differences.

Per-app environment

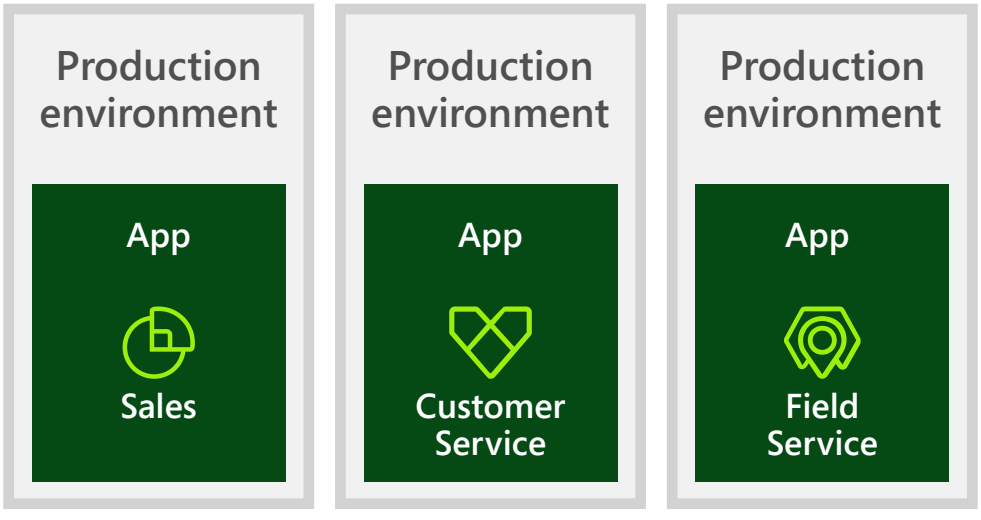
In a per-app deployment model (**Figure 9-6**), every application gets its own production environment, with a separate set of environments to support the underlying ALM and release process. There is complete isolation of the data, schema, and security model. A per-app environment might seem simpler from a deployment perspective, but it can create data silos such that one business process cannot easily benefit from sharing information with another, leading to additional effort in building complex integrations.

Also, the security model is defined per environment, so you can't use the platform security constructs to define which data from one environment will be accessible to a user in a different environment.

Consider an example where the Sales and Customer Service

Fig. 9-6

Per-app environment



apps are deployed in separate environments, requiring core customer records to be synchronized across the two. In this scenario, providing sales representatives with visibility into support experiences will require some level of integration, either by copying over the data periodically or integrating the visual user interface (UI).

This approach might be more relevant for isolated functions within an organization where there is little need for data sharing and collaboration. For example, an internal HR ticketing system for employees has little to do with the sales process. Network latency, compliance, and region-specific requirements could mandate the use of per-app environments, but will introduce data-replication challenges, integration redundancy, and increased costs for storage, integration, and maintainability.

Let's examine some of the pros and cons for the per-app deployment model.

If you're deploying more than one app, you will need to carefully consider these factors in coordination with the multiple-app and per-app environment strategies covered in the previous section.

Pros of a per-app deployment model:

- Its security model is separately managed for each app, which simplifies configuration when there is no need for data sharing between environments or development of custom security components.
- Its ALM and release process for each app can be independent.
- Its capacity utilization for individual apps can be tracked easily for cross-charging within the business.
- Its design issues in one app will not directly affect other apps.
- It is the preferred approach if there is a need to segregate the environments for a globally distributed user base due to network latency, compliance regulations, or process differences.

Cons of a per-app deployment model:

- It can create data silos between business units and departments, potentially reducing collaboration and value to the business.
- It means a linear increase in the number of environments to support ALM for each app and the administration effort for platform updates.
- It largely prevents reuse of integrations and is a potentially complex solution requiring data sync and complex custom security logic. In some cases, integrations are not possible across several environments.

(For example, you can't sync users' Exchange mailboxes to several environments at the same time.)

- It potentially brings higher API and storage costs due to data and integration redundancy.
- Its reporting and intelligence architecture will be more complex and require data unification in an external data store, such as a data lake.

Global deployment scenarios

This section will focus on three common app-deployment models you could use—along with the tradeoffs, pros, and cons when choosing one approach over the other—to help you find the best option for your deployment.

We're discussing global deployment scenarios in the context of a single-app deployment. If you're deploying more than one app, you will need to carefully consider these factors in coordination with the multiple-app and per-app environment strategies covered in the previous section.

Global single environment

A global single environment is just one environment deploying an app that's accessed by users across the globe. All the data, processes, code, customizations reside in a single environment. Based on the app's security requirements and regional data-sharing policies, your organization will have to manage access via security roles. This is the most common approach, as it enables strong global collaboration and a unified view of data and processes across different regions, business units, and legal entities.

While a global single environment simplifies deployment and maintenance of the solution, centralized process governance is needed to ensure that different regions conform to a unified process, with deviations kept to a minimum. If processes are not unified globally,

handling region-specific requirements will require a complex solution—likely with poor performance—which ultimately will affect usage and adoption.

Your organization might have to adjust the security model for each region to meet local regulations and accommodate cultural differences around sharing and collaboration.

Your organization might have to adjust the security model for each region to meet local regulations and accommodate cultural differences around sharing and collaboration.

Let's examine some of the pros and cons of a global single environment.

Pros of a global single environment:

- Data unification across regions and countries enables greater collaboration and helps standardize processes
- Settings, updates, configurations, and ALM are easier to manage
- Costs of storage and integrations are reduced by avoiding duplication
- Intelligence and reporting architecture, as well as master data management, are simplified

Cons of a global single environment:

- Change management and governance could be harder when trying to make a process work for everyone while avoiding political challenges
- Poor adoption is possible in regions where users aren't engaged or if their requirements are deprioritized
- Security model configuration could become complex, as each persona might have a different regional flavor
- Some regulatory and data-residency requirements would be hard to meet in a global single environment, and could slow down change and business agility if the effort isn't well coordinated
- High network latency for users in different countries or regions can affect performance and adoption
- Maintenance schedules and downtime are difficult to manage

Global multiple environment

Multiple-environment scenarios support different business units, departments, teams, countries, or other structures inside an organization.

Each environment has its own database, and data isn't shared across environments. This builds a physical security aspect on top of logical,

out-of-the-box security based on security roles. In separating environments by region or country, solutions included in each environment can support local business and compliance requirements.

If users are distributed over large distances, particularly for global deployments, multiple environments may be more suitable because of the implications (such as latency) associated with the connection infrastructure, which can significantly affect the user experience. Distributing environments to provide users with more local access can reduce or overcome network-related issues, as the access occurs over shorter network connections.

By creating multiple production environments, it's possible to easily implement specific business requirements and have different teams delivering the solution within different time frames.

In terms of ALM, the solution-release process can be unique for each environment, which will support specific business units, departments, and countries. However, it will increase the number of environments needed to support development, testing, and training activities, as each production environment needs its own environment to support the solution-release process.

Because the data is not unified in a global multiple-environment setup, the reporting and analytics strategy is more complex, and typically there needs to be a data-consolidation exercise for reporting on key performance indicators (KPIs) and insights about the overall business.

This approach is common when companies have different requirements across subsidiaries, business units, or countries. By creating multiple production environments, it's possible to easily implement specific business requirements and have different teams delivering the solution within different time frames. Each subsidiary, business unit, or country operation has control over its own environment.

This model makes it easier for local operations to run and maintain their solutions, but it also increases the costs, and may lead to siloed work and deployment.

Let's examine some of the pros and cons of a global multiple environment.

Pros of a global multiple environment:

- Individual environments are only responsible for fulfilling their

own requirements, making them simpler and smaller
- Local administration minimizes language and time-zone barriers
- Local environments may perform better when closer to the end user, reducing latency
- Compliance with local requirements, such as GDPR, is easier
- Data can be physically separated, and there's flexibility in managing data residency
- Maintenance time windows are easier to adjust

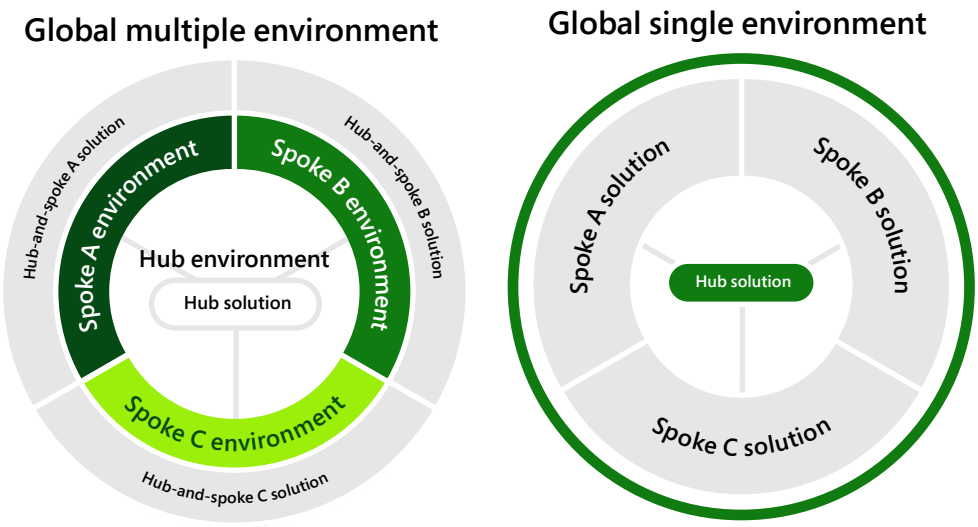
Cons of a global multiple environment:

- Data must be unified in a central repository for reporting and analytics
- Data isn't updated across environments or may become specific to each environment
- There's no single 360-degree view of a customer
- It's more costly
- Users may need to access different environments for different purposes, affecting the user experience
- Multiple instances must be managed

Hub-and-spoke model

A hub-and-spoke model (**Figure 9-7**) depicts the company's core unified business solution as the hub. The core is then distributed and extended by the subsidiaries, business units, or country operations, which are depicted as spokes.

Fig. 9-7



The hub-and-spoke model could be a variation of the multiple-environment model where each of the regions have an independent environment, but share the core data model.

Alternatively, this could also be achieved with multiple apps in a single environment, where each region is able to independently manage the variations.

There are multiple factors that could trigger environment transitions, ranging from the overall ALM process to changes in business structure.

Environment lifecycle scenarios

Environments transition through various states before they are decommissioned. Some of these transitions are natural to support ALM, but there could be business, compliance, and technical reasons that trigger these transitions. As described earlier, environment-related changes are complex, and they require significant planning and (usually) downtime.

There are multiple factors that could trigger environment transitions, ranging from the overall ALM process to changes in business structure. Let's explore the various types of transitions that you might see throughout the lifecycle of an environment. These transitions could require admins with appropriate roles to self-serve and, in some cases, seek assistance from product support teams.

Creation

An organization might create a new environment for several reasons. The purpose of the environment—and considerations such as the environment type, country or region, apps required, access security groups, URL and agreed-upon naming conventions, and tiers and service levels—should be well defined and understood before its creation.

Environment transition

An organization might have multiple types of environments, each targeting specific use cases, including trial environments, default environments, sandboxes, and production environments. Be aware of possible transitions between types, as there could be limitations and implications to the service. For instance, changing a production environment to a sandbox might change the data-retention policy and applicable service-level agreements (SLAs).

Copy

Most often used in troubleshooting scenarios, environment copy lets you create a copy of an existing environment with all its data or only

the customizations and configuration. Be aware of the effect of storage capacity and compliance requirements, which could restrict copying of customer data. Sometimes copy is also used to support ALM instead of using the source control. This pattern should be avoided, and environments should never be used as a repository for code and customizations.

Restore

Sometimes, you might need to restore an environment. Depending on the type of environment and service, there could be several restore points or a feature allowing you to restore an environment to a precise time. Restore could lead to data loss, so it's not really an option for production environments unless performed in a limited window.

Geo-migration

Geo-migrations could be triggered by changes in regulations, or simply because of a need to move an environment to a lower-latency region for a better user experience. This would normally involve creating a copy of an environment and then physically moving it to a different region. It almost certainly will involve downtime for users, and requires approval from the compliance and security teams. It also could lead to change in service URLs and IP ranges, affecting integrations and network security.

Tenant-to-tenant move

Any change in tenant strategy might trigger a request to move an environment from one customer tenant to another. This is not a common request and will require several steps before the actual move. Most importantly, users must be correctly mapped between tenants and the record ownerships must be restored. It might require a regional migration first and could involve several hours of downtime.

Merged environments

Merging multiple environments into a single environment takes substantial planning and could be extraordinarily complex. Merging involves the data model and the processes in the app, followed by the actual data with necessary transformations and deduplication, then the security model and, finally, the integrations.

Split environments

Splitting an environment could be required if an organization transitions to a multiple-environment strategy. This could involve provisioning new environments and migrating all previous customizations, along with the relevant data subset. Alternatively, an organization could create the split by copying the environment and moving it to another region.

Migration from on-premises to cloud

Many organizations have on-premises infrastructure, but the evolving needs of businesses and the potential and benefits of the cloud lead them to migrate to a SaaS cloud solution. Traditionally, this has involved creating new cloud environments, running a cloud-compatibility assessment, moving the solution components and data, and redesigning parts of the solution to take advantage of the latest cloud features. To help speed up this process, Dynamics 365 services offer a "lift and shift" solution to move data into the cloud. It's fast and cost-effective, but by moving a legacy solution into the cloud, organizations may miss an opportunity to optimize their processes.

Administration mode

Administration mode can be used for maintenance when only selected users can access the environment. Before you do this, assess the systems that will be affected by putting an environment in administration or maintenance mode, such as a public-facing portal that connects to the environment.

Deletion

Deleting an environment removes all the data and customizations, as well as the option to restore and recover. The process of decommissioning or deleting an environment needs gated approvals.

Governance and control

Establishing a center of excellence (CoE) for business applications that is responsible for defining guidelines, creating best practices, and nurturing organic growth plays a fundamental role in driving business innovation.

A CoE helps enforce security and policies, but also fosters creativity and innovation across the organization. It empowers users to digitize their business processes, while maintaining the necessary level of oversight.

Good governance is critical during deployment and for long-term system operation. But the governance and control processes for a centrally deployed and managed IT solution might be vastly different from processes used to deliver a secure ecosystem of business-user-developed applications.

A CoE helps enforce security and policies, but also fosters creativity and innovation across the organization. It empowers users to digitize their business processes, while maintaining the necessary level of oversight.

With the changing technology landscape and evolution of security mechanisms, organizations should review and update policies on a continuing basis. The better organizations understand the service architecture, the greater the opportunity to fine-tune the controls, rather than creating overly restrictive policies.

To define a robust environment governance policy:

- Examine your organization's usage, application types, user personas and location, data confidentially, and access control
- Evaluate your capacity needs in terms of storage, API call, and other factors
- Understand the release and rollout strategy, and how it affects the environment lifecycle
- Enforce audit requirements at the environment and application levels.
- Ensure DLP policies apply tenant-wide
- Define admin roles and processes to obtain admin privileges
- Monitor deployment and usage

Product-specific guidance: Operations

Now we'd like to offer some guidance specific to the Dynamics 365 Finance, Dynamics 365 Supply Chain Management, and Dynamics 365 Commerce apps. A team implementing a project with these apps requires environments to develop, test, train, and configure before production. These nonproduction environments come in a range of

tiers with pre-defined sizing, topologies, and costs. Understanding these basics is key to charting out a well-designed environment plan.

What is environment planning?

Environment planning is the process of describing the details of all the environments required during and after implementation. It includes procuring and preparing all environments that the team will need to adequately develop and test the Dynamics 365 solution.

To learn more, check out the Environment Planning for Finance and Operations Implementations Tech Talk.

For Finance, Supply Chain Management, and Commerce projects, the FastTrack team provides a sample environment plan in a matrix format that lists the type, topology, deployment option, and due date for each environment. This plan is a living document that should be revisited after each milestone of the project.

The importance of environment planning

Implementing an enterprise resource planning (ERP) and a customer relationship management (CRM) project typically requires collaboration among multiple teams that will need to access the apps for different reasons—to develop, build, test, and train. So, part of your plan should cover making environments available for different teams to ensure they can deliver their respective tasks on time. Also, if you plan for all needed environments early, you will know what the costs are going to be throughout the project and at which stage(s) you need the funds, and you will avoid project delays from last-minute discoveries of missing environments.

A note on LCS projects

In the case of Dynamics 365 Finance, Supply Chain Management, and Commerce, all environments are contained in one project on the Microsoft Dynamics Lifecycle Services (LCS) portal. This LCS project is hosted on one Azure AD tenant, and includes all the environments used in the implementation, testing, and for going live, such as the production environment.

Typically, an implementation project has only one LCS project per Azure AD tenant. In rare cases, an organization may decide to have multiple LCS projects. Reasons to have more than one LCS project or instance include times when a global implementation's requirements for data residency, latency, or data volume can't be met by one instance. Or perhaps different business units are implementing the product separately as independent applications.

Each LCS project must meet the minimum licensing requirement. Currently, you need to open a support ticket to request additional project, but soon this will be a self-service activity.

Environments and project complexity

When you buy the Finance, Supply Chain Management, and Commerce apps from Microsoft, you get two environments. One is the production environment, which is where your company business will be conducted. It's a robust environment, with high availability (HA) and disaster recovery (DR) capabilities.

In addition to the production environment, you will receive one sandbox environment that's a standard acceptance testing (tier 2) instance for the life of the tenant. It's a nonproduction, multibox instance that can be used for testing, UAT, integration, and training needs.

(Because it's a basic tier 2 instance, it may not be enough for every type of testing. For example, performance testing needs a bigger instance to handle larger data volumes and more users.)

Depending on the project complexity, these environments may not be sufficient. **Figure 9-8** depicts a sample chart comparing the additional requirements of two projects.

For less complex projects, you can use the out-of-the-box tier 2 instance for testing. You still need to consider development environments for development activities, with one development environment per developer. These environments have Microsoft Visual Studio installed.

You should also think about the build environment. Source code developed in the development environment is synched to Azure DevOps. The build process will use the build environment, which is a tier 1, to produce code packages that can be applied to sandbox and production.

Alternatively, you can use Microsoft-hosted build agents to do the builds instead of having a dedicated build environment. The limitation of this approach is that you can't run unit tests, and there is an overall

Fig. 9-8

Environment plan sample

Environment purpose	Finance and Supply Chain Management	Dataverse	Standard subscription	Less complex project	More complex project
Testing or training	2	Sandbox	● ●	● ●	● ●
Production	Sized	Production	● ●	● ●	● ●
Development	1	Development		● ●	● ●
Build	1	n/a		●	●
Golden configuration	1 or 2	n/a		●	●
Data migration	2+	n/a		●	● ●
Performance testing	4 or 5	Sandbox		● ●	● ●
Training	2 or 3	Sandbox			● ●
System integration testing	2 or 3	Sandbox			● ●
Other/ad hoc	n/a	Trial or sandbox			● ●

● Finance and Supply Chain Management environment ● Dataverse environment

time limit per month. Assess the limitation and decide whether you need a dedicated build environment.

Next is the "golden configuration" environment, for configurations that are fully tested and those you want to retain and transfer to other sandboxes and production environments.

You can consider a data-migration environment to move data from legacy systems, and a performance-testing environment—which is a higher tier (4 or 5) environment—to conduct performance tests.

When it comes to more complex projects, you may need more environments. For example, you may wish to conduct testing in parallel if multiple teams are developing and testing parallel workstreams.

Production environment and go-live assessment

When you're preparing to go live, the first thing you need is access to your production environment. Project-wise, a production environment is deployed after all customizations are code-complete, UAT is finished, the customer has signed off, and there are no blocking issues.

For a Finance, Supply Chain Management, and Commerce project, you won't have access to this environment on day one, as Microsoft needs to complete few prerequisites—including a go-live assessment—before we can release the production environment.

In a go-live assessment, a solution architect is assigned to review the project and provide an assessment of potential risks, best practices, and recommendations for a successful go live. In some cases, the solution architect might highlight risk factors and ask for a mitigation plan. When the assessment is completed, the solution architect will indicate that you're ready to request the production environment in LCS.

Self-service deployments

For Finance, Supply Chain Management, and Commerce environments,

Please note that the production environment is used exclusively for running your business operations and shouldn't be used for testing. You will be able to perform the cutover and, if planned, "mock" the cutover in production.

there is a new type of self-service deployment that's rolling out incrementally across the globe. The benefits are easier and faster deployment times, and there are a few differences from the earlier infrastructure as a service (IaaS) architecture. For example, there is no remote desktop access; to access the SQL server database, you will have just-in-time access.

General recommendations

- Plan for environments early in the project and revisit the plan at regular intervals.
- Define a consistent naming standard for your environments. For example, a gold environment should have "gold" in the name.
- Have a regular schedule to deploy updates and import fresh data (if needed).
- Keep all environments in the same region if your business is in one region. For example, avoid having a test environment in one geographical location and production in another.
- Deploy environments by using an unnamed account, such as dynadmin@your_organization_name.com. Assign the environments to an owner who will be responsible for their status and maintenance. We strongly recommend using the same dedicated admin account on all environments.

Assign the environments to an owner who will be responsible for their status and maintenance.

Test environments

- Consider the number of testing environments you will need throughout the project. A lack of environments may prevent concurrent testing activities and delay the project.

Training environments

- Make sure all users have access with appropriate roles and permissions, which should be the same roles and permissions they will have in production.
- Plan for downtime and have a communication plan to alert users. (Zero downtime is the eventual goal.)
- Ensure all integrations are set up and working, so users can experience the end-to-end cycle of a business process.

Gold environment

- Make sure no transactions happen in a gold environment. There should be a process to bring tested configurations into the gold environment.

Data-migration environments

- Assess whether you need a dedicated environment for data migration, which is a disruptive task that can't generally coexist with other types of test activities. Multiple data-migration environments may be needed to avoid conflicts if multiple parties are migrating data concurrently.
- Account for data-migration performance in environment planning. Depending on the size of the data-migration task, it may be necessary to use a tier 3 or higher environment to perform data-migration testing. (You can also use an elevated cloud-hosted environment.)

Pre-production environments

- Assess whether there is a need for a separate environment to test code or configuration changes before they're applied to production.
- If there will be continuing development of the solution after you go live, you may need a separate pre-production environment to support concurrent hotfix and hotfix test activities. (This environment should have the same code base and data as production, so a like-for-like comparison can be performed for any new changes before they're applied to production.)

Performance-testing environments

- Plan a specific environment for performance testing, or you won't be able to do performance testing activities in parallel with other test activities
- Avoid doing performance testing in a cloud-hosted environment, the production environment, and tier 1, tier 2, or tier 3 environments
- Use of tier 3 or smaller environments typically won't provide the resources required to perform a performance test

- Limitations on the number of threads and database transaction units (DTUs) available in these environments won't allow for modeling of most concurrent workload profiles
- Turn off these environments when not in use to save costs

Developer environments

- Ensure each developer has an independent development environment

Production environment

- Raise production environment requests through support, as you don't have direct access to this environment

Product-specific guidance: Customer Engagement

Throughout this chapter, we have discussed concepts related to environment strategy that are product-agnostic and apply to most cloud deployments. Now we're going to focus on product-specific resources that apply to Power Platform and customer-engagement apps.

Environments and Dataverse in Power Platform

For each environment created under an Azure AD tenant, its resources can only be accessed by users within that tenant. An environment is also bound to a geographic location, such as the United States. When you create an app in an environment, that app is routed only to datacenters in that geographic location. Any items that you create in that environment—including connections, gateways, and flows using Power Automate—are also bound to their environment's location. Every environment can have up to one Microsoft Dataverse database (**Figure 9-9**), which provides storage for your apps.

Dynamics 365 products and Power Apps

Microsoft business applications are a set of intelligent solutions that

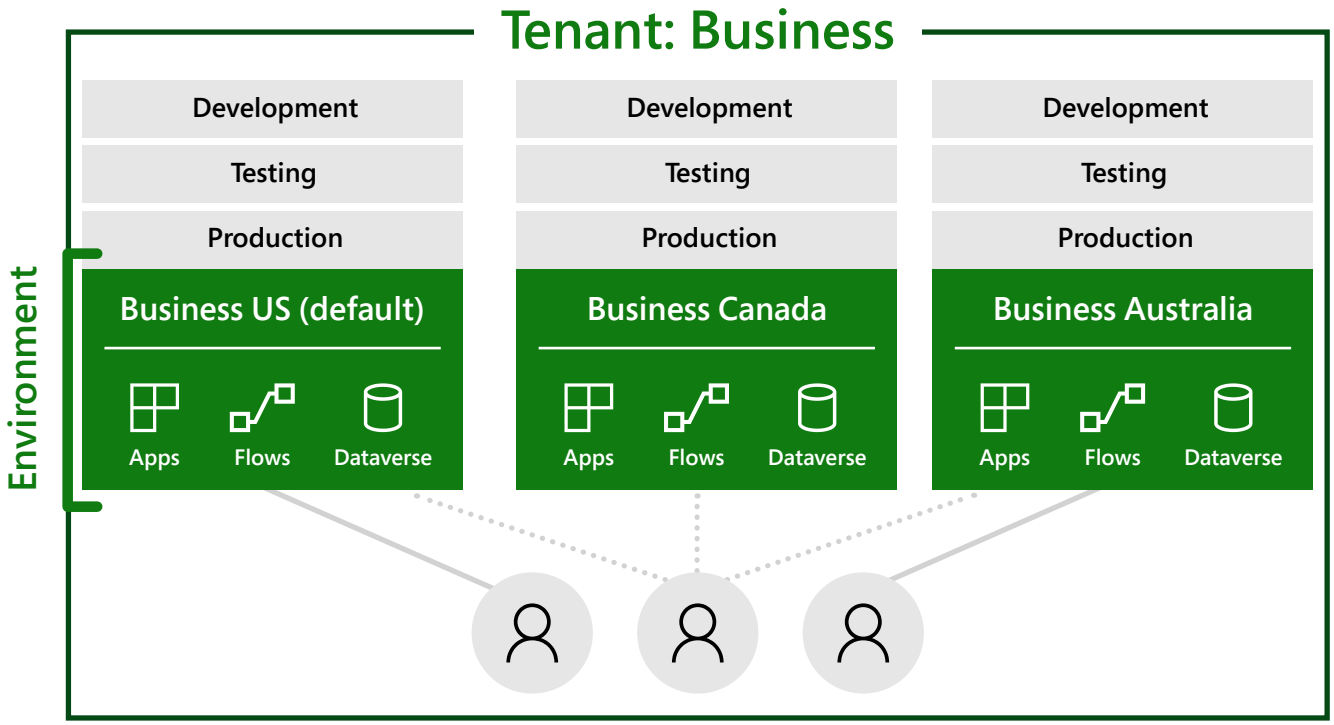
supply a comprehensive view of an organization's business. These solutions include the Dynamics 365 products that are connected by data and intelligence and supported by the Power Platform.

Microsoft unified CRM and ERP within Dynamics 365 and the Power Platform to make it easier to create apps and share data across all Dynamics 365 applications. The combination also creates a set of purpose-built apps with threaded intelligence to connect front-office and back-office functions through shared data. Rich analytical capabilities provide organizations with deep insights into each functional area of their business.

Power Apps is a suite of apps, services, connectors, and a data platform that provides an app-development environment where you can quickly build custom business apps that connect to your business data stored in either the underlying data platform (Dataverse) or online and on-premises data sources.

Customer-engagement apps such as Sales and Customer Service are first-party Power Apps developed by Microsoft.

Fig. 9-9



Administration and governance

The Power Platform admin center provides a unified portal for administrators to create and manage environments, including sandbox environments, backing up and restoring environments, and copying an environment. Admins can also manage settings for Power Apps and Power Automate.

The Microsoft Power Platform Center of Excellence (CoE) Starter Kit is a collection of components and tools that are designed to help you get started with developing a strategy for adopting, administrating, and supporting the Power Platform, with a focus on Power Apps and Power Automate.

On-premises to online migration services

The Dynamics 365 migration program enables on-premises customers to simplify and accelerate their move to the cloud. The migration program also features access to a dedicated team of migration advisors, no-charge migration assessments, pricing offers, tools (including a CRM platform assessment tool), and migration support for qualified customers.

Conclusion

As we have seen throughout this chapter, defining an environment strategy for your organization is a critical step when planning for deployment. Decisions made about environment strategies are hard to change and could be very costly later in the program lifecycle. The wrong environment strategy can create unnecessary data fragmentation and increase the complexity of your solutions. Early environment planning aligned to your organization's digital roadmap is fundamental to success.

Checklist

Organization environment and tenant strategy

- [ ] Define environment and tenant strategies and obtain agreement for them at the program level with all key stakeholders, including business, IT, security, and compliance.
- [ ] Create a strategy that considers the future growth of the solution.
- [ ] Create an environment strategy that can support the ALM processes and necessary automations.
- [ ] Create an environment strategy that considers the short- and long-term impact on licensing, compliance, application design, performance, scalability, maintainability, and ALM of the solution.
- [ ] Create a strategy that considers the potential need for citizen development scenarios or collaborative development with both IT and business users.
- [ ] Create an environment planning matrix with key considerations of the pros and cons to help plan and visualize the impact.

Environment app strategy

- [ ] Create an organization environment strategy that defines the guidelines for onboarding additional apps.
- [ ] Assess the impact on the overall data estate. Avoid excessive fragmentation and promote reuse of existing integrations and data flows.

Global deployment

- [ ] Account for global deployment scenarios; additional coordination and agreement may be required to meet regional requirements and compliance needs.
- [ ] Assess the latency to choose the optimal location—global deployments are prone to network latency related performance issues.

Governance and control

- [ ] Establish governance processes for provisioning, monitoring, managing the lifecycle, and decommissioning the environments early on.
- [ ] Ensure the different security personas involved in the environment management are understood and appropriately assigned.
- [ ] Use the CoE Starter Kit to make necessary adjustments to citizen development use cases as needed because they require further governance consideration.

Case study

Timely environment planning is a critical factor to success

A multinational retailer with stores in Southeast Asia, Australia, and New Zealand implemented the Dynamics 365 Finance, Supply Chain Management, and Commerce apps in a series of phased rollouts spread over eight months.

The management team was not clear that the environments, services, and default capacities included in the SaaS subscription license—mainly the production and sandbox environments and hosted build automation services—were the minimum needed to run the solution on an ongoing basis, but were insufficient on their own to complete all the implementation activities required for the moderately complex rollout.

When the FastTrack team was engaged early in the implementation process, this issue was addressed as part of the Solution Blueprint Review (SBR). In discussions with the solution architect, the retailer and partner members of the implementation team gained a better understanding of the deployment plans and risks posed by their current environment strategy (or lack thereof). After the review, the team agreed to conduct a deep-dive workshop on environment strategy with FastTrack, which was one of the recommendations based on the findings from the SBR.

In the workshop, the solution architect explained that environment planning is not a one-time, set-it-and-forget-it event. Rather, it's a continuing activity that requires close collaboration between key stakeholders. Careful planning aligned with the project schedule ensures that the team can be productive, conduct important activities concurrently without environment contention, and achieve milestones on time, within budget, and with the desired level of quality.

Next, the project team charted out an environment plan to document the different environments, their purpose, the exact project phase for which they were needed, and how long they would be needed. For example, they acquired a high-grade testing environment for three months just to conduct their performance testing. They also identified key activities that might affect the environment plan and incorporated recurring reviews of the environment plan into the project plan. This simple but important exercise helped the team feel confident that they understood and accounted for all important implementation requirements, and that they were prepared to procure the needed environments on time to reach all milestones.

The development team also worked on configuring DevOps for multiple release branches, with a separate pipeline of sandboxes to facilitate go-live phases and ongoing support for each one.

Ultimately, this retailer successfully performed the implementation and rolled out the solution across multiple countries—and realized that timely environment planning was a critical factor in that success.

Section

Implement

10 Guide Data management

Introduction

Data surrounds us every day, like a blanket. Whether you are posting on your social media, scheduling a doctor's appointment, or shopping online, the information collected is one of the most valuable assets of any business.

With the right data, organizations can make informed decisions, improve customer engagement, and gather real-time information about products in the field.

In this chapter, we discuss the many ways data plays a part in defining a solution. Data plays a vital role in the success of every deployment.

You learn about:

- Data governance
- Data architecture
- Data modelling
- Data migration
- Data integration
- Data storage
- Data quality

This chapter aims to break down the various functions within data management that collectively ensure information is accessible, accurate, and relevant for the application's end users. We focus on the most common discussion points surrounding the lifecycle of data within a project.

Regardless of your role, take the time to consider what is important to each person interacting with the data. For example, users of a system focus on their data quality, ability to search, data relevance and performance, while architects and administrators are focused on security, licensing, storage costs, archival, and scalability.

Make sure to always ask these questions:

- How is the data used?
- What use cases require the data?
- Where is the data coming from?
- Where is the data stored?
- Is there a cost (internal/external) to access the data?
- Do users need their data on a mobile phone?
- Do customers need access to their data?

We use the Success by Design framework to highlight critical questions like these that must be asked throughout the project lifecycle to ensure the desired results for everyone.

Data governance

Let us start the discussion with data governance before we start unpacking different principles of data management.

While data management is usually seen as dealing with operational issues of data, data governance is about taking a high-level strategic view of policies and procedures that define enterprise data availability, usability, quality, and security.

With artificial intelligence (AI) and machine learning (ML) taking center stage in most digital transformation projects, and the fact that the success of these initiatives is highly dependent on data quality, it is prudent that executives start considering data governance seriously.

You may have heard of instances where different departments in the same company do not agree on the definitions of business terms and the values reported. For example, a sales report generated last month may experience two departments reporting different values. This could be happening because the two departments are using two different systems, which is caused by there not being a common definition of business terms and values that everyone in the company agrees upon.

With advanced data governance in place, the walls that create data silos effectively crumble as data is shared in a trustworthy way with a consistent understanding of business concepts and definitions.

Sharing data within the company is crucial for two economic reasons—growth and distribution. Data is a non rival resource. It is not a material resource that if one person uses it, others cannot use it. If data is shared with everyone in your company, people can start becoming self-sufficient and build on top of it increasing the overall value as it becomes available to all.

Though data governance is a broad topic, for our discussion we want to focus on key aspects that can help drive a successful implementation.

Consider the following key pillars when analyzing data governance within your implementation.

Data stewardship

A data steward is a role within an organization responsible for the management and oversight of an organization's data assets, with the goal of providing that data to end users in a usable, safe, and trusted way. By using established data governance processes, a data steward ensures the fitness of data elements, both the content and metadata, and they have a specialist role that incorporates processes, policies, guidelines, and responsibilities for administering an organization's entire data in compliance with policy and/or regulatory obligations.

Data quality

The quality of the data cannot be thought of as a second or third process. Data quality is at the front of data governance policies. It should be thought of as high quality and fit for the purpose of whatever the intended use is. Driving data's accuracy and completeness across the organization is typically managed by dedicated teams who may use a variety of tools to scrub the content for its accuracy. Although these tools aid the process more and more, this is still typically a human responsibility.

Without data governance in place, organizations struggle to control corporate assets needed to ensure data quality.

The users that are most familiar with their data should typically be the gatekeepers for cleansing, including standardization and adherence to the policies outlined by the data steward.

Master data management

Most enterprises keep a primary set of data used across the organization to supplement systems and reporting. The data elements are typically stored within a master data management (MDM) solution. This includes customers, accounts, products, and other areas according to business needs. There are typically rules of engagement that internal teams requesting access to the master data must follow, which is documented within the data governance plan and managed by the data steward.

Proper use cases

A data governance plan cannot be set up without understanding the proper use cases. Every line of business (LOB) has a level of ownership of their data along with the use cases that drive the solution design. The same holds true for data governance. The use cases aid in naming the primary data elements the organization is taking proactive control of.

For example, one retailer wants to better target its customers through email campaigns, which in the past failed to deliver expected results due to incorrect contact information being captured. While defining this use case, the company also set up the right data governance principles that ensure what data quality is required for email campaigns. They were challenged to define “good” data to satisfy the use case. This simple use case uncovered other issues. The company found that online customers, who buy as guests, could enter any value in the email field and there was no validation. This led to data stewards and LOB process owners setting up new validation processes.

Without data governance in place, organizations struggle to control corporate assets needed to ensure data quality. During the requirements gathering stage of your implementation, start paying particular attention to the availability of data required for your solution. Early discussion and identification goes a long way in defining your use cases.

For an example of a proper use case, see "A Show-Don't-Tell Approach to Data Governance."

Data architecture

After data governance is understood within the organization, the next step is to look at data architecture and the different types of enterprise data.

According to Data Management Book of Knowledge 2 (DMBoK 2): "Data architecture defines the blueprint for managing data assets by aligning with organizational strategy to establish strategic data requirements and designs to meet these requirements."

Data architecture is a high-level concept, traditionally part of enterprise architecture. In simpler words, data architecture describes how data can be provided to meet a business's strategic objectives.

The right data architecture is central to the success of your business's data strategy. It focuses on where and how you use your data across various sources and what technologies are available to harness. It is important to know where the data is coming from, if it is accurate, and if it provides the insights needed by the business within the solution.

If your organization has an enterprise-level data architecture in place, make sure to research policies, procedures, and restrictions that are in place for accessing the information you need.

For example, an organization may have a number of disparate applications to support different business units, and one of the legacy applications (not Dynamics 365) is traditionally used as a central repository for customer master data. The enterprise data strategy may be to support a master version of the customer data in order to supply better analytics and visibility of customer interactions across all lines of business. However, interactions with a customer, like sales opportunities, sales transactions, support cases, or general activities, may be managed within your Dynamics 365 solution.

When there is no enterprise data architecture, if you simply draw out all the existing business applications, their interfaces, and connections, you come up with a spaghetti diagram. That is a good start to getting an overview of your integration and data architecture. In fact, the resulting data flow diagram is an important input into defining your integration strategy.

Read more about integrations in Chapter 16, "Integrate with other solutions."

The key point to data architecture is to create a holistic view of the data repositories, their relationships with each other, and ownership. Failure to implement data architecture best practices often leads to misalignment issues, such as a lack of cohesion between business and technical teams.

During your design and analysis phases, it is important to capture the data owners, systems, and conceptual flow between systems. This information helps to meet the integration requirements, as well as providing a reference for the broader project team.

When you prepare your data architecture, use the questions below to guide you.

- How do you use your data?
- Where do you store your data?
- How you manage your data and integrate it across your lines of business?
- What are the privacy and security aspects of the data?

Before you can start working with data, it is important to know what a typical enterprise data is made up of, as illustrated in **Figure 10-1**.

Types of enterprise data

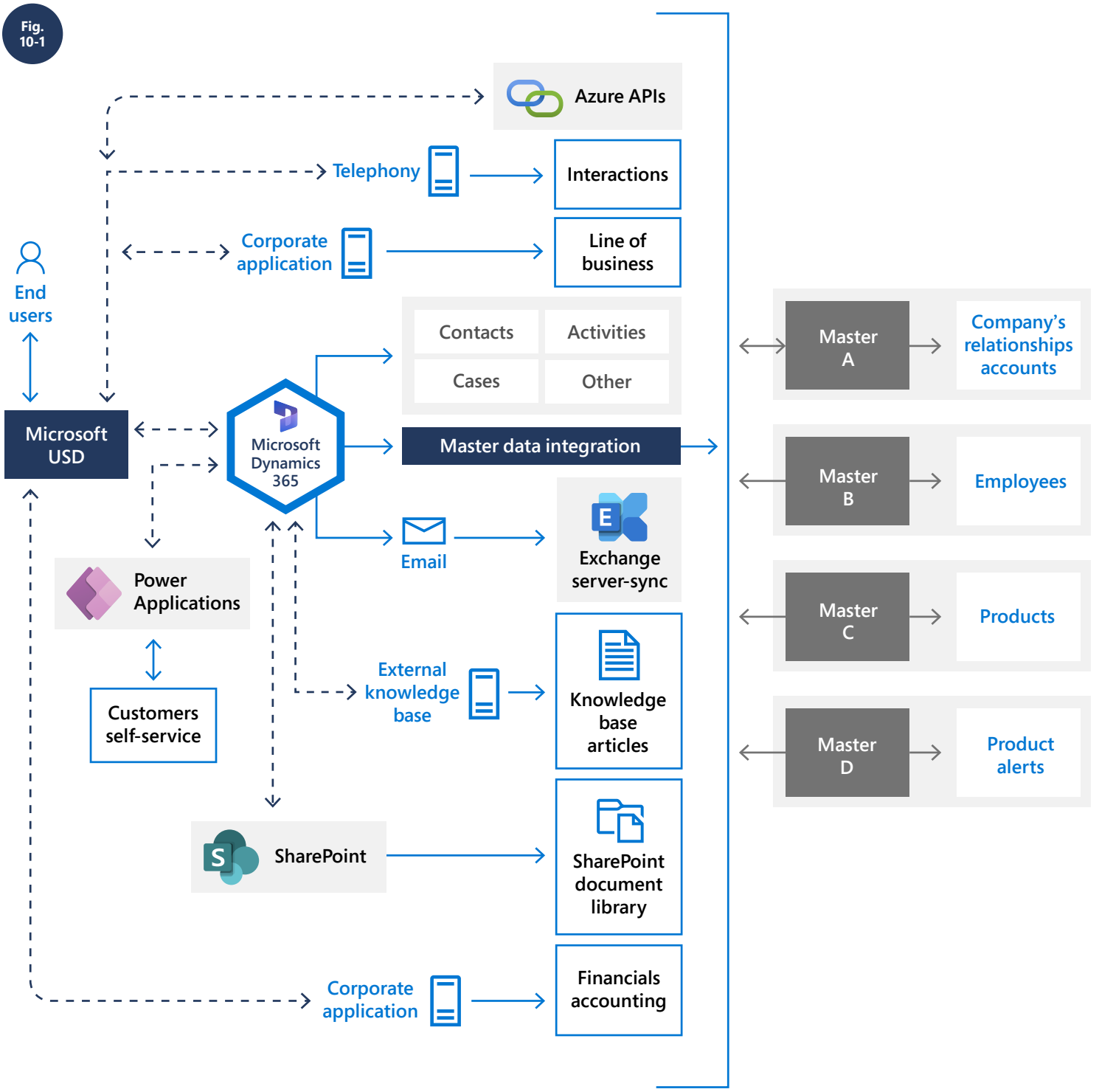
There are several distinct types of physical data to be aware of. This type of data typically falls into one of four categories: master data, configuration data, transactional data, and inferred data.

Master data

Master data is your source of common business data and represents a non transactional, consistent, and uniform set of tables. The details are maintained and controlled for accuracy. For example:

- Customers
- Vendors
- Products
- Company data
- Contact data
- Account holdings

Fig. 10-1



Configuration data

Configuration data is data about different setups needed to prepare your production environment for operational use. Besides importing master data like customers and products, you need to import the configurations data. For example:

- Currencies
- Tax codes
- Modes of payment
- Address (countries, states, postal codes, etc.)

Transactional data

This type of data is generally high in volume due to the nature of its use. Transactional data typically refers to events or activity related to the master data tables. The information is either created automatically or recorded by a user. The actual information could be a statement in fact (like in banking), or it could be a user interpretation, like the sentiment of a customer during a recent sales call. Here are a few other examples:

- Communication history
- Banking transactions
- IoT transactions
- ERP transactions (purchase orders, receipts, production orders, etc.)

Inferred data

Inferred data is information not collected by the business or users. Typically, this information is automatically generated based on other external factors, which adds a level of uncertainty. For example:

- Social media posts
- Credit score
- Segmentation

Data modeling

With a base understanding of the data governance and data architecture, we can now focus our attention on how we plan to store the information we collect. Data modeling should always be completed before any configuration begins. Let us start with a basic understanding of data modeling, recommended practices, and how it is related to a project.

According to the DMBoK 2, “The process of discovering, analyzing, representing and communicating data requirements in a precise form is called the data model.” While data architecture is at an elevated level and involves a data architect looking at the business requirements broadly, data modeling is at a lower level and deals with defining and designing the data flows in precise detail.

Data architecture and data modeling need to work together when designing a solution for a specific business problem. For example, if your organization wants to implement an e-commerce solution, you cannot do that unless somebody knows and has defined the existing data architecture and data models, different integrations in play, existing data imports and exports, how customer and sales data is currently flowing, what kind of design patterns can be supported, and which platform is a better fit into the existing architecture.

Likewise, if your organization wants to implement a data lake, somebody needs to answer questions like what the existing data flows are both in and out of the business applications, what is their format, and what are the existing data warehouses we need to migrate to data lake?

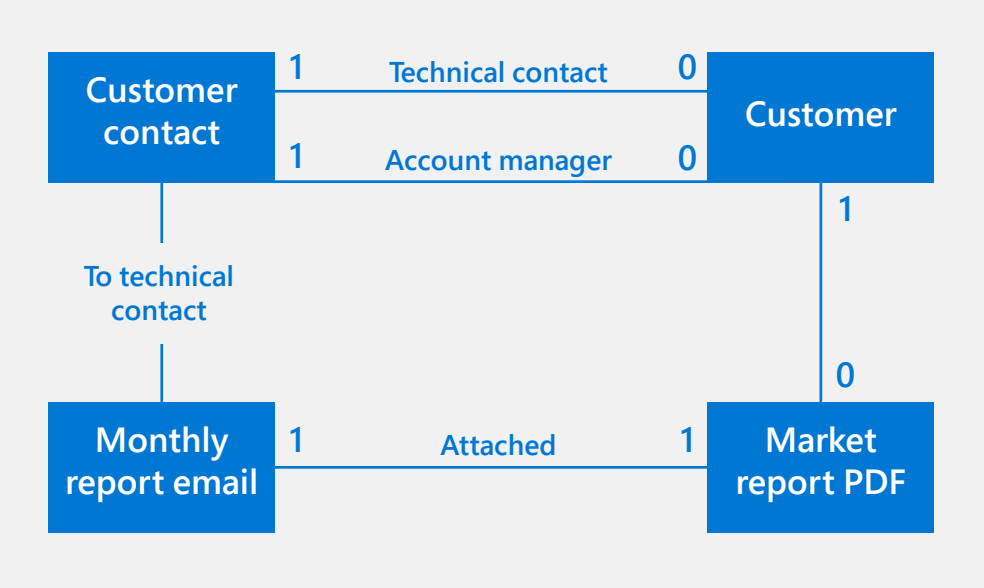
It is also recommended that the data model conforms to the Common Data Model standard, which does not deviate from or repurpose existing out-of-the-box tables, to ensure cross application compatibility and future readiness. Common Data Model provides a shared data language for business and analytical applications to use and makes it possible for data and its meaning to be shared across applications and business processes such as Microsoft Power Apps, Power BI, Dynamics 365, and Azure. Refer to For more information, refer to **Common Data Model**.

What is data modeling?

A data model is a visual model showing how data flows through your system and how different tables relate to each other. Data models define the relationship types between tables and abstract a database to a visual representation that is easy to understand.

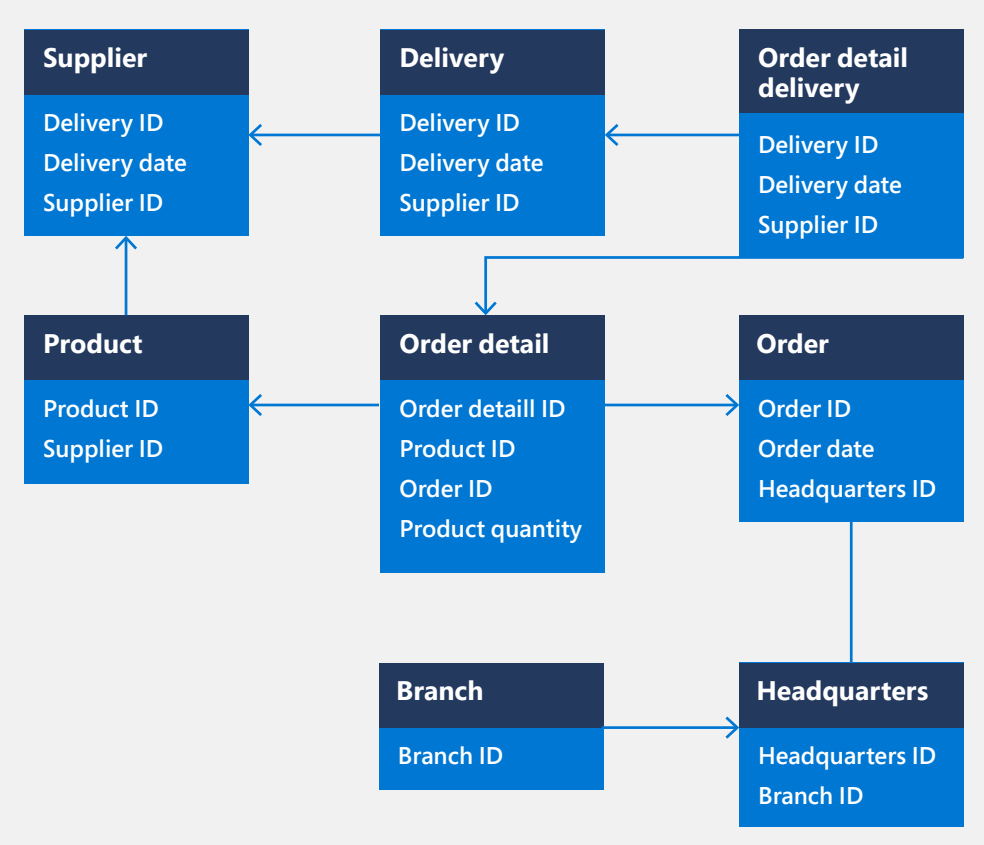
If your project requires creation of new tables to support storing data for some specific business processes, it is recommended that you create a data model to define and understand the schema and data flow. But if yours is largely a vanilla implementation or has minimal customizations, you may not need to do this. It is recommended to discuss with your system integrator and solution architect to see what works best for your case.

Fig. 10-2



There are multiple types and standards for data modeling, including Unified Modeling Language (UML), IDEF1X, logical data model (**Figure 10-2)**, and physical data model (**Figure 10-3)**. A common type of physical data model is an entity relationship diagram (ERD).

Fig. 10-3



Data storage

Next, we need to talk about data storage and what you can do to manage the growth and costs involved. The reality is no matter where your data is stored, there is always a cost involved. Whether you are managing data on-premise or in Microsoft's cloud, you should be aware that storing and moving your data can impact the overall cost of the solution.

There are several factors to consider when defining a data storage strategy.

Forecasting capacity

The capacity your environment consumes should be part of the overall environment strategy. Every environment should be assessed for storage requirements based on usage. For example, the development environment does not require as much storage as a production environment to operate.

First, you need to calculate the approximate volume and size of the data to properly come up with an estimate. You can gather this detail from your existing data sources. You should come up with a size either in megabytes (MB) or gigabytes (GB) needed in production.

The reality is no matter where your data is stored, there is always a cost involved.

Based on the estimated size for production, you then need to allocate a percentage for each environment. **Figure 10-4** provides an example.

The best practice is to build a data storage forecast for a minimum of three years, including an average increased annual volume. We recommend that when designing and configuring your different environments you discuss with your system integrator, and Microsoft, the amount of storage you anticipate you need and when you need it by. The business needs to track the data growth to make sure the overall size stays within your allotted storage entitlements.

Fig. 10-4

Environment	Use	% of Product
Production	Contains all data required for release	100
Training	Contains a sampling of data for training	15
QA	Contains a sampling of data for testing	15
SIT	Contains a sampling of data for testing	15
DEV	Contains limited data only for development needs	5

Environments should be managed with a well-planned environment strategy. For example, a developer should not be able to simply go ahead and create new environments on the fly. Doing so could create opportunities to go over entitlements, which leads to enforcement, which in turn could limit your ability to deliver on the application lifecycle management (ALM) strategies.

Read more about environment and ALM strategies in Chapter 9, “Environment strategy,” and Chapter 11, “Application lifecycle management,” respectively.

Configuration data and data migration

Once you have a good understanding of your business requirements, project scope, and have set up proper data governance, data architecture, and data models, you need to start the preparations for importing data to prepare for your solution to be ready for go live(s).

There are two types of data we are talking about: Configuration data and migrated data.

Managing these two data imports is often the most challenging part of Dynamics 365 implementations. We often see customers underestimating the efforts required and projects suffering as issues surface later. In this section, we talk about plans for configurations, plans for data migration, different environments, identifying data sources, ETL process, and finally, staffing the right resources who can manage these efforts.

Configuration data

Configuration data is data about different setups needed to prepare your production environment for operational use, for example, currencies and tax codes. Besides importing master data like customers and products, you need to import configuration data. Managing configurations is an activity that needs to happen throughout the lifecycle of the implementation. As the team understands the business requirements and designs required, they understand different setups and parameters needed to enable the business processes required by the business.

Since the Dynamics 365 offerings are highly configurable, you need tight control over the different configurations required to be set. With

so many development and testing environments and different team members, there is a high chance of incorrect configurations being set, leading to errors in processes. In the case of Dynamics 365 Finance, Supply Chain Management, and Commerce, we recommend setting up a dedicated golden configuration environment to capture and maintain the correct configuration data. A gold environment can help ease the pain in tracking, storing, updating, and controlling the configuration data. Gold environment settings should be kept pristine, have tight access control, and no one should be allowed to create any transactions. You can use the golden configuration database to restore other environments for testing, data migration, and even initial go live.

Use a configuration plan

Next, it is important to discuss the idea of using a configuration plan. This is especially important in a phased rollout scenario when you need to plan to setup a number of environments, legal entities, or business units.

You can use a golden configuration environment to restore into production for the first phase of go live, but for every subsequent phase of go lives you do not have that luxury. The double whammy of a large number of setups and slight but crucial variations of each go live can be highly daunting, so doing it manually is fraught with danger. We highly recommend that organizations use a configuration plan.

Configuration plans are a structured list of different master and configuration data that you want to import for go live and any subsequent go lives. It is organized in the sequence required by the system and manages the dependencies between entities. As an example, if you need to enable purchase orders you need vendors first, if you need projects, you need accounts first, etc.

This plan also represents the functional scope for the solution. As described in Chapter 7, “Process-focused solution,” you start with your process definition and scope for the implementation. This process requires enabling functionality to be enabled in the application, and this functionality requires configuration and master data and eventually some transactional data in connection to the first use of the

application, like some opening balances. The configuration plan represents this scope in terms of system enablement and the correlation between all of them.

Another benefit of the configuration plan is to provide early visibility of how those configurations and master data play a role between the different business units (legal entities) or apps that share the same database foundation. You can determine if that data is natively shared, or if it is unique for an entity. Having this visibility prevents you from having redundant data replication and improves the speed to enable the solution.

For example, flagging the setups that only need to be imported once as they are shared across business units or legal entities versus setups that need to be imported for each phase as they are not shared. A configuration plan can help you be organized and consider the requirements for all the subsequent business units or legal entities. Another example, in the case of Dynamics 365 Finance, Supply Chain Management, and Commerce, can be whether you want to use data sharing features to copy the configurations across companies instead of manually importing those configurations into each company.

The configuration plan is your configuration playbook that shows you scope, dependencies, and data redundancy and on top of that, it can help you to track the progress of enabling the solution in terms of configurations and master data required to be loaded.

Note that managing a configuration plan is an iterative process. As you make progress in the project and discover new requirements, and thus new setups, you should come back to the plan and update it. The list is your one-stop shop to tell you what you are going to do for each go live for each configuration, what data is already available, and what data needs to be imported. Just like your golden configuration environment, it's crucial to keep your configuration plan up to date.

A proper well-maintained configuration plan makes sure the team considers all the important configurations for all the planned rollouts and does not miss them causing unwanted errors in production after go live.

This plan also helps the team be prepared for cutover and calculates the required downtime window to import all necessary data. The team should test the configuration plan a number of times, note the time it takes for all the configurations, and confirm that the total time required is within the cutover window.

We strongly recommend that teams put some thought in and create a configuration plan to manage all their imports as the benefits of reduced errors and faster imports far outweigh the efforts needed to create and manage the plan.

Since the configuration plan is essentially a list, many customers prefer to use Excel sheets as most organizations are familiar with the program. There are also customers who prefer to maintain the list in DevOps. For example, you can have one work item for currencies and another work item for tax codes with exported data entities attached to the work items. It does not matter which option you choose, as long as your team members can access and update the list when necessary.

A sample plan for managing configuration imports for Dynamics 365 Finance, Supply Chain Management, and Commerce can be downloaded from Sample Configuration Plan using Data Entities.

Data migration

The second type of data we are dealing with is migrated data. Data migration, in simple terms, is the process of moving data from one data model to a different data model for future use. In the case of Dynamics 365, data migration refers to moving data from a legacy system to a Dynamics 365 application. Whether your legacy system was another application or disparate Excel spreadsheets, you may have been capturing data to run your business and you need that data when you move to your new Dynamics 365 application.

Migrated data is either master data such as customers, products, and vendors, or open transactions such as open sales orders, open purchase orders, on hand stock, and open balances.

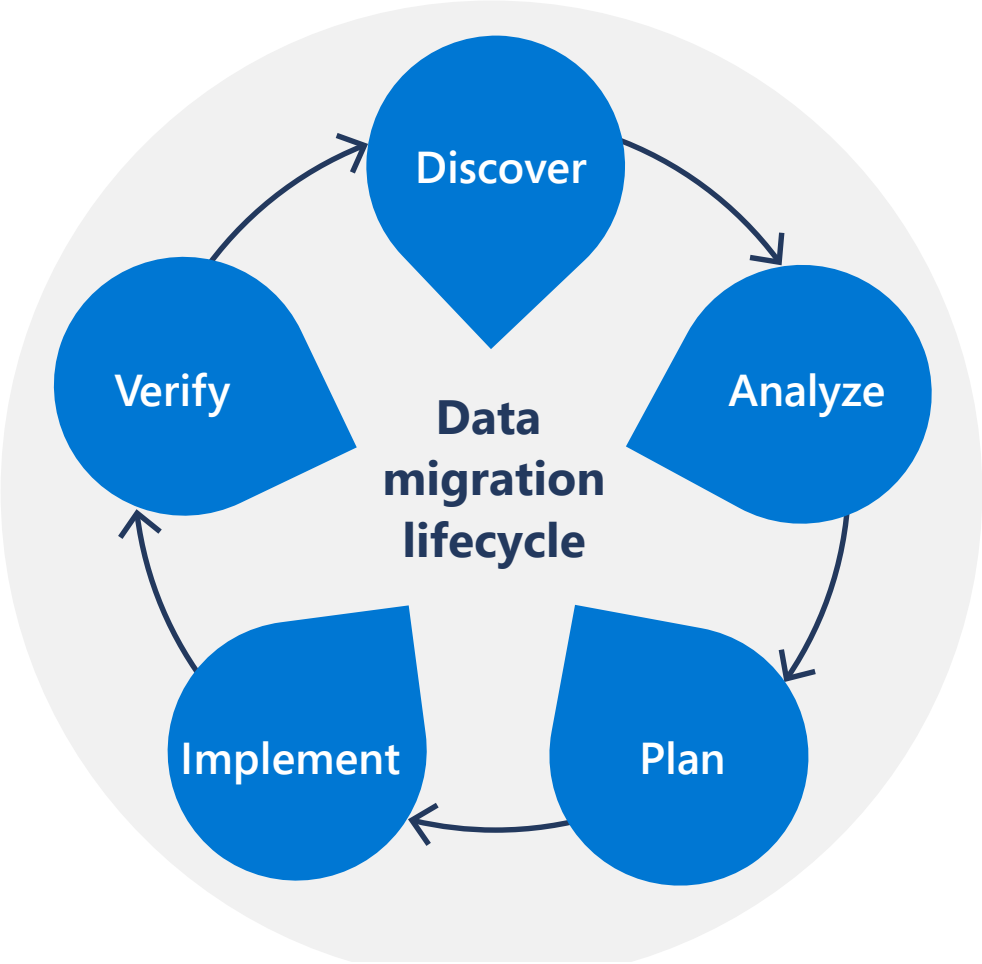
Migration planning

When deploying a solution, data is particularly important. Different departments within the organization can find it challenging to support certain activities when there is no data in the system.

Typically, when replacing another business application, relevant data is migrated into the new system during the deployment so the users can see relevant business information when they start using the application. It is especially important to manage the timeline of data migration activities in the project plan so you can allocate time and people to the activities. For example, you need to track activities like migration planning, identifying data sources, data mapping, ETL (extract, transform, and load), importing data for UAT, and importing data for go live.

Explore the parts of the data migration lifecycle in **Figure 10-5**. Take time to **discover** the mandatory data required by your solution and **analyze** the data types and systems that source the data to be imported.

Fig. 10-5



Migrating data is a time-consuming and expensive process that often lacks full visibility and understanding of the use cases. The use cases help define how the data is used within your solutions processes.

When building a **plan**, keep in mind that data migration activities can be a disruptive task and should not co-exist with other testing activities, so it is advised to procure a dedicated high tier data migration environment.

It is recommended to **implement** at least one round of system

integration testing (SIT) and user acceptance testing (UAT) with migrated data. During testing, developers and users are able to **verify** the migrated data and ensure it meets the requirements.

Defining a migration strategy

Data migration is a major undertaking and is often run as a parallel activity to the development and testing activities and spans multiple phases. Several decisions need to be made regarding the scope of the migration, environments, source databases, and the tools used to extract and transform.

There needs to be clear business requirements defining the data migration scope, timelines, and expected cutover plans before a project kickoff. You should raise concerns immediately if these details are not clarified.

- Identify the source for where the data needs to be pulled and migrated.
- Define the environments that are needed (staging environment).
- Is there a mapping document for the tables/fields between old and new systems?
- Do you have the right analysts and developers resourced?

Multiple sources data

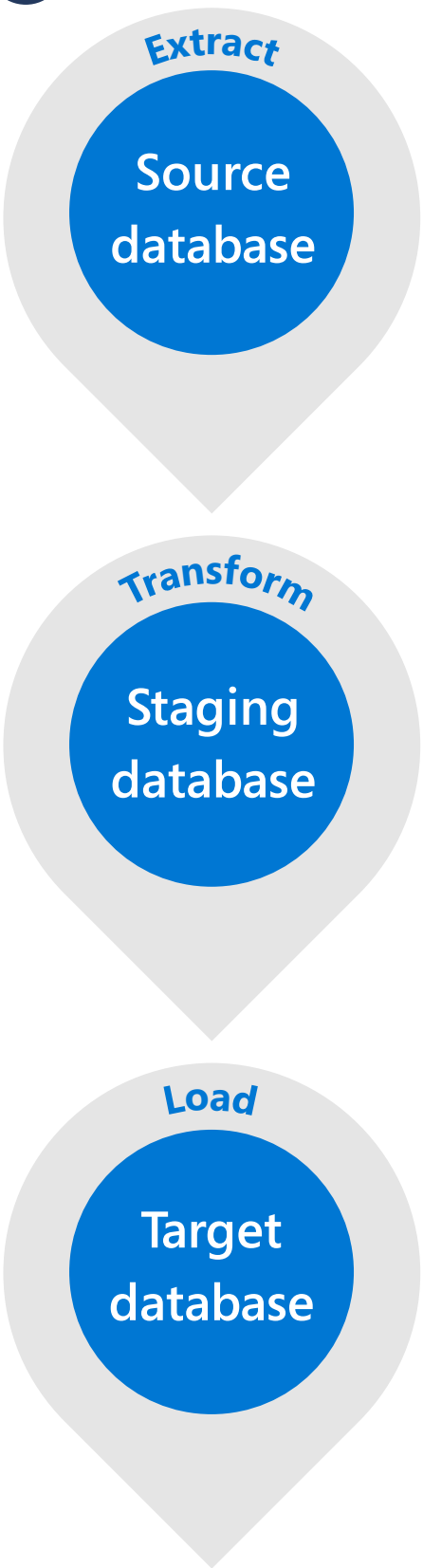
The source data can come in all shapes and sizes. From a salesperson's Excel spreadsheet to a legacy systems SQL database. Knowing all the sources allows time to analyze the options for extracting the information.

- SQL database
- Third party database
- External webservices
- Access
- Flat files/Excel

In advance and during the project, it is recommended that the LOB begin the process of identifying, consolidating, deduping, scrubbing, and cleansing the data. The goal is to limit the amount of transformation

needed during migration. The more logic required to transform, the slower the migration takes to load.

Fig. 10-6



Environments

Another missed factor is sizing of the import and staging databases required for running migration tooling and cleansing of data. You need to make sure environments used are sized appropriately to handle the volumes of data in scope. It is recommended practice to have all databases and environments running under the same location and region. During environment planning, the appropriate environments should be sourced for data migration.

Data mapping

The process of data mapping can start once the solutions for data modeling have been defined. You should be organized to keep the process systematic and simple.

- Use an Excel spreadsheet with each tab representing a system table.
- Capture all the fields being mapped together along with any transformation requirements.
- Pay close attention to the target system you are loading data into. For example, data types should match between systems or transformed accordingly.

Extract, transform, load (ETL) tooling

Once you have identified the sources, set up the environments, and documented the data mappings, then the logical next step is to **extract** the data from the source, **transform** the data for target, and **load** the data into the target, as illustrated in **Figure 10-6**. You can use many types of tools and approaches when migrating data to your Dynamics 365 solution. Some of the very standard options include:

- Data import/export wizards
- Azure Data Factory
- SQL Server Integration Services (SSIS)
- Third-party integration products

Refer to the product-specific section later in this chapter for a list of some of the most common options.

Roles and responsibilities

Customers and partners should staff the project team with the right resources who understand data and the tools in Dynamics 365.

Examples of some important roles and responsibilities in **Figure 10-7**.

Data integration

Integration is the connecting of one or more parts or components of a system to create a more unified experience or to ensure a more consistent outcome of a process. Integration allows leveraging of existing services both internal and external without having to rebuild or migrate existing functionality.

Data integration is done to bring data into the system or out to other systems. Typically, this happens through an event or in a batch on a schedule. For example, when a record is created or updated it would be event driven and the nightly scheduled transmission of data would be a batch.

Refer to Chapter 16, "Integrate with other solutions" for more details.

Fig. 10-7

Role	Responsibility
Data migration analyst	Assist in designing, planning, and managing the data migration process. Work with subject matter experts and project team to identify, define, collate, document, and communicate the data migration requirements.
Data steward	Maintain data and manage it according to data properties as required by administration. Coordinate with stakeholders and provide all definitions for data.
Data migration architect/ developer	Design and develop the environments required for migration. Develop data packages as required to move the data between systems. Provide initial testing and validation.

Data quality

Once data is migrated and integrated to your solution, it is critical that information remains accurate, complete, reliable, and, most importantly, up to date.

You can use various techniques to manage the quality of the data, including data profiling, data cleansing, and data validation. The most important aspect of data quality to understand is that data quality is the responsibility of everybody in an organization. We see many customers who overestimate the quality of their data and underestimate the effort it takes to get it into shape. Keeping data quality top notch requires following all the principles we highlighted in this chapter and equally strong leadership to drive the habit of managing data on an ongoing basis.

Make your data quality a habit!

Conclusion

Data has taken center stage in all enterprise transformation projects. Businesses want to be data driven. Having the right data flowing in your business means you have set up a well-functioning business in which you have the most up-to-date and accurate information about your customers and products and can provide a superior experience to your staff and customers.

The benefits of data do not just stop there. Having the right data means you can start leveraging ML and AI today to predict what your customers need tomorrow.

Having proper data governance, data integration, data security, and data quality are not just best practices; they are necessary for your business's survival because every business wants to be data-driven and compete for customers' attention. These concepts can help lay the foundation on which you can start tapping into data to transform your processes, gain insights, and predict customer behavior.

Data management covers so many different areas, it is easy to get overwhelmed. By starting with the right people, process, and data, you cannot go wrong.

Product-specific guidance

Up to this point in the chapter, our data related guidance has applied to Dynamics 365 Finance, Supply Chain Management, Commerce, as well as Customer Engagement application projects. While both applications live in the Dynamics 365 ecosystem and customers frequently adopt both systems, often simultaneously, there are differences between the two. This can mean differences in how each application should manage its data.

In this section we highlight some of the different approaches for both applications.

Customer Engagement

This section includes a number of recommendations and resources provided in Customer Engagement to help manage modeling, storage, migration, and archival.

Data modeling

Data modeling is a science, and there are data modeling professionals and established standards for data modeling. To be effective with Dynamics 365 data modeling, you do not have to be a professional data modeler or use any special tools. Popular tools like Microsoft Visio can be used to quickly create a basic ERD diagram that visualizes the relationships and flow of data between tables. In this section, we discuss some general best practices for data modeling for Dynamics 365 deployments.

- Data models should be updated continuously during a deployment. It is common for a data model to be designed at the beginning of a project, but it is very important that it does not stop there. As you go through the deployment, new fields and tables are added—it is important to capture these in the data model to make it a living data model. Recommend to customers that they continue to update it in order to enhance the system.

- Do not include every table. Some core tables, such as activities, notes, and users (record owners), are related to nearly every entity in Dynamics 365. If you include every relationship with these tables in your data model, the result is unreadable. The best practice is to include only the primary tables used in your configuration in your data model diagram, and include only custom relationships with the user and activity tables to maximize readibility.
- Data models should include tables outside of Dataverse. If you are integrating with other systems via Dataverse data connectors or virtual tables, or if data flows outside of the Dataverse via an integration, this data should also be represented in your data model diagram.
- Start simple with the standard tables, then add custom entity relationships to your data model.
- User experience should influence your data model. Sometimes it is easy to overnormalize your data, but the process could make the application more cumbersome to use.

Start with what you need now but design the data model in a way that supports what you are going to be doing in the future. For example, if you know that down the road you need to store additional details about sales territories, using a text field for territory now makes it more difficult to implement than if you use the territory entity relationship. Plan for what is coming.

Data storage

This section provides product-specific guidance you can use while implementing or maintaining your solution.

Storage capacity

The storage capacity is a standard calculation within the Power Platform that is easily managed by the system administrator. The Power Platform admin center is the tool you should use to maintain visibility of storage and consumption. Within the Power Platform admin center, go to Resources > Capacity > Dataverse for more details about your capacity entitlements, as shown in **Figure 10-8**. You can access this by going to the Power Platform admin center.

Storage entitlements

Dataverse capacity (database, file, log, and add-ons) is pooled across the tenant and shared among all environments and workloads. The first subscription provides a one-time default capacity entitlement for the tenant. Learn more about the Dataverse storage capacity.

Storage segmentation

To better understand how capacity is calculated within Customer Engagement, the following provides the breakout based on storage type and database tables.

Dataverse database: All database tables are counted for your database with the exception of logs and files below.

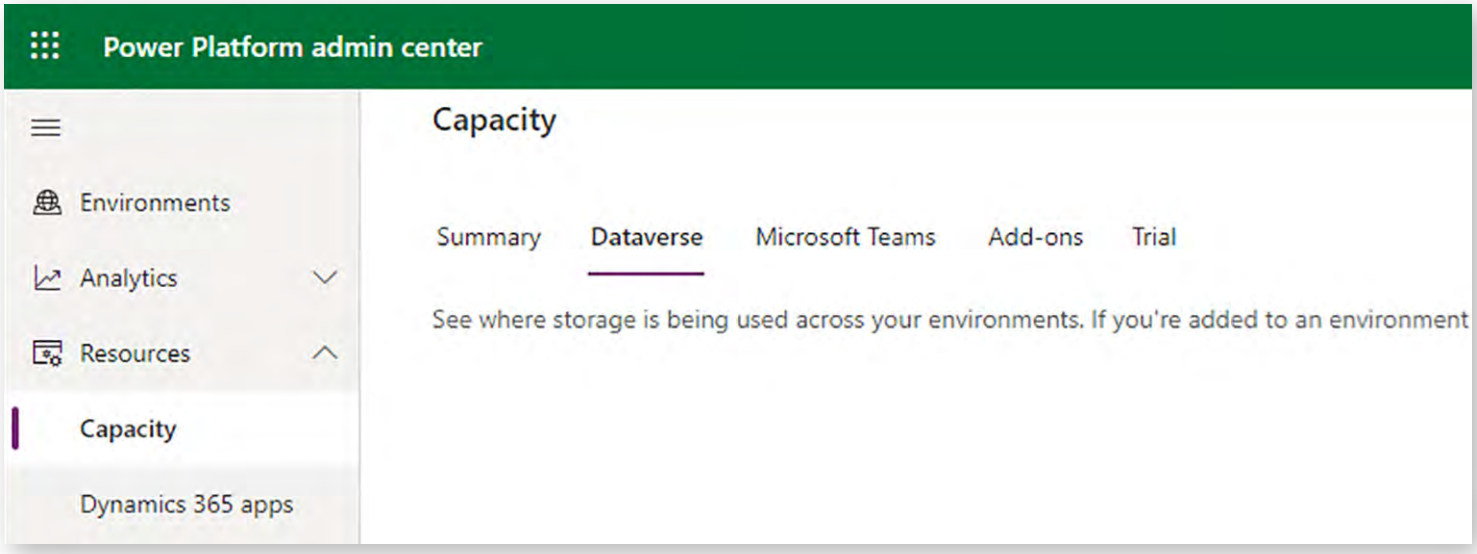
Dataverse files: The following tables store data in file and database storage:

- Attachment
- AnnotationBase
- Any custom or out-of-the-box entity that has fields of datatype file or image (full size)
- Any entity that is used by one or more installed Insights applications and ends in "-analytics"

Dataverse logs: The following entities are used:

- AuditBase
- PlugInTraceLogBase

Fig. 10-8



Data migration

This section provides product-specific guidance you can use while implementing or maintaining your solution.

Approaches and tools

You can use many types of tools when migrating data to your Dynamics 365 solution. The following is not an exhaustive list, but it includes some of the most common options.

Do not migrate data It is often assumed that large amounts of data must be migrated into Dynamics 365 to provide a complete view of the customer. However, just because you need to see data in Dynamics 365 does not mean that the data must live in the Dynamics 365 Dataverse database. Considerations should be taken for the value of the data weighed against the quality of data, the volume of the data, the amount of time required to load the data, and potential storage costs before migrating data. Not physically migrating data can frequently be replaced with embedded Power BI reports, virtual entities, embedded canvas apps with connectors to the source data, and other options. For example, the email auto-capture option in Dynamics 365 Sales Insights displays activities to salespeople from their Exchange inbox rather than having to load the data to Dynamics 365. This can be a viable option to show activity data rather than loading large amounts of data. Virtually displaying data can also significantly reduce the time and cost of deployment, as you do not have to take time to load data and test your data migration.

Power Apps Excel add-in This add-in can be used to open entities directly in Excel and create and update records. Records are updated or created directly in the Dataverse. Not all entities support updates from Excel, and lookup fields must be manually edited to correctly match.

Get data from Excel At make.powerapps.com, you can choose "Get data" when viewing an entity and import data from an Excel or .csv file. This option gives additional control over the mapping of data than the Excel add-in, and lets you import data from one file into multiple entities. To help ensure that the format of fields and mapping

is correct, you can export a template from a CDS entity, populate your data, then import it. Alternatively, you can import a custom file and provide mapping. Lookup fields must include the primary key or entity alternate key valued to match correctly.

Power Platform dataflows At make.powerapps.com, you can select "Dataflows" from under the Data menu and configure an import from several data sources. Data sources includes common external services file sources as well as generic web APIs. Data from these data sources can be transformed prior to import using Power Query.

Legacy Dynamics 365 data import utility You can import data to Dynamics 365 entities from CSV, .xls, .xml, and .zip While the Dataverse API Get Data option is recommended for most flat file imports, the legacy data import option has several unique capabilities that might be useful in some cases.

- Legacy data import can be used to create new entities and fields and option set values in Dynamics 365. While this is convenient, the best practice is to add these items from a solution rather than from data import.
- Legacy data import can import multiple files at the same time when multiple files are added a zip file.
- Legacy data import can resolve lookup fields using values not contained in the primary or alternate keys.

Dynamics 365 Configuration Data Migration Utility This is a tool provided by Microsoft that is designed to move data between Dynamics 365 environments. If you are migrating data from sand-box environments to production, this tool is very useful as it allows you to keep the record IDs consistent between environments. Consider this when you have entities like country and specific records are referenced in workflows or flows, if the ID changes between environments, the process does not work. Configuration Data Migration Utility was updated in early 2020, so you can now provide filters to tailor the records included in the Configuration Data Migration Utility data packages.

Extract, transform, and load (ETL) software For more complex migrations, such as migrating an entire legacy CRM environment's data, manual import from flat files is not efficient and can be error prone. For more complex migrations, commercially available ETL tools like SSIS, Azure Data Factory, or a number of third-party ISPs offer services and tools that can be used to create a migration data transformation specification (DTS) that can migrate legacy data to Dynamics 365/CDS. This approach can have the following benefits:

- Reusability of migration, allowing the data migration developer to test and refine the migration multiple times before go live.
- Delta sync loads when moving the data from an in-production system to a new system. Users still use the legacy system until the switchover to Dynamics 365 happens. If the data is loaded into production several days before go live, there is additional data created in the legacy system after the initial load is finished. ETL tools allow the data loaded to be filtered to only include data changed since the last load, ensuring that the migrated data is up to date when the users start using the new system.
- Consistency with data integration. Sometimes data is migrated in an initial load, and then updated via an ongoing integration. In these cases, it is optimal to use the same tooling that you use for the ongoing integration to also migrate the initial data load, and in those cases the same DTS and field mappings may be used for the data migration as well.
- More complex data transformation. When moving data via flat files, you can choose what data is mapped to the new system, but if you wish to change or transform the data, it must be done manually in the source flat files. With an ETL based migration, you have flexibility to transform the data during the migration. For example, say there are five different types of accounts in the source data, but you wish to consolidate them to three, ETL tools allow for this transformation in the data translation specification.
- Updates and upserts. The flat file and Excel imports support updating records that match record keys, but sometimes your matching logic is much more complex. Say you want to both insert and update records (upsert). This is complex and tedious to do with the out-of-the-box data import options, as you don't always have the record IDs. ETL tools allow the data migration

developer to define record matching logic and update, insert, or upsert records based on any matching criteria. This is also helpful for making sure that duplicate records are not being created in the target.

- More flexibility in mapping lookup fields. Lookups to other entities can be challenging for data imports, especially fields like "customer" that are polymorphic, or fields like "owner" that have special properties. If your legacy data has owning users that have a different format, the ETL tool can transform the values into a format that can be imported.
- Many-to-many (N:N) relationships. ETL based imports can easily import data to N:N relationships and entities like marketing list member, which is not available from some of the flat file import options.
- Faster import of large data sets. Good ETL tools can use multi-threading to import data quickly into the common data service.

Power Automate and Azure Logic apps These can be used to import data and provide connectors to over 300 services. These options provide many of the same benefits as an ETL tool. In addition, since Power Automate includes more than 300 connectors to many leading services and application platforms, Power Automate can be a great option to migrate data from these services. Although we have these options, they are not really meant to perform large data migration jobs, they could be considered a great option for low-volume or low-rate delta migrations that are not too demanding on throughput.

Custom API migrations These can be used to migrate data to Dynamics 365. This option gives you more control over the migration and does not require any extra tools, but it requires that you have a developer to develop and support the data migration.

Data archival and retention

The challenge for most organizations is not necessarily defining a strategy but rather implementing one. Depending on your solution, there is most often a need to be a technical solution required to satisfy the strategy of moving the data.

As mentioned earlier, over time the volume of your data grows as well as the cost. Implementing a data archival and retention strategy allows regular shifting of data from one storage location to another. The fact is, over time your data grows and the SQL server comes under strain. This impacts SQL server's performance and thus degrades user experience. Even though Dynamics 365 is a SAAS application, and you do not bother with managing SQL server, it is a clever idea to have periodic data maintenance routines in place to archive and delete unwanted data to keep database nimble.

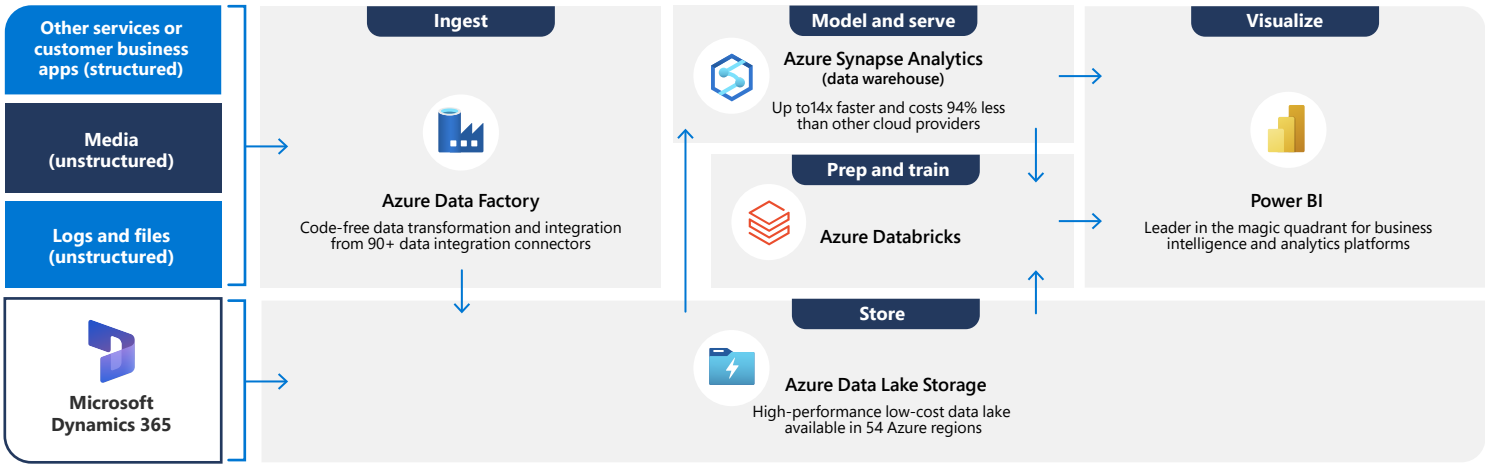
For example, you can set up data export schedules to replicate data to Azure Data Lake, which is comparatively cheaper than Dataverse. From a best practice's perspective, you do not want to keep old historical data in Dynamics 365, data that the business does not need for day-to-day operations. Once data is moved to the data lake, you can setup Azure Data Factory to create dataflows, transform your data, and run analysis and you can use Power BI to create business reports and produce analytics.

Figure 10-9 shows a modern data warehouse architecture by using Azure Data Lake Storage.

For more information, read [Ingest Microsoft Dataverse data with Azure Data Factory] and [Analyze Microsoft Dataverse exported to Azure Data Lake Storage Gen2 data with Power BI].

More information about reporting strategy can be found in Chapter 13, "Business intelligence, reporting, and analytics."

Fig. 10-9



Operations

Dynamics 365 Finance, Supply Chain Management, and Commerce have a number of data management tool sets and data maintenance cleanup schedules, which we describe in more detail here.

Data maintenance

When it comes to data storage, an important factor to consider is that not all data needs storing. You have lots of logs and staging data generated that can be safely truncated after a certain time.

This is where you need to set up a plan for data maintenance.

In Dynamics 365 Finance, Supply Chain Management, and Commerce, cleanup routines are available across various modules, which can keep the tables tidy. It is important to note that these cleanup routines should be completed only after detailed analysis and confirmation from the business that this data is no longer needed. Always test each routine in a test environment prior to running it in production. This article provides an overview on what is available today: Cleanup routines in Dynamics 365 for Finance and Operations.

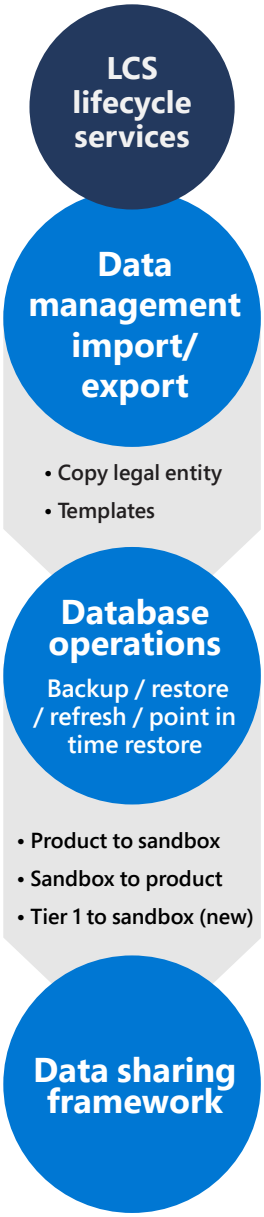
Best practices with handling PII data

- Avoid storing unnecessary PII data in Dynamics 365 Finance and Supply Chain Management if possible.
- Identify, tag, and classify PII data that you need to store.
- Dynamics 365 Finance, Supply Chain Management, and Commerce uses Azure SQL database that allows data encryption at rest and transport.
- X++ APIs/patterns to encrypt and decrypt data at columns level for added security.
- Build Edge applications/integration to help store and mitigate data residency and PII requirements.

Data management toolsets

Dynamics 365 Finance, Supply Chain Management, and Commerce has a rich toolset and processes available to support customers' data movement

Fig. 10-10



and migration requirements. Customers can use the features in app and LCS to combine different approaches. **Figure 10-10** highlights the options.

Data management workspace

In the case of Dynamics 365 Finance, Supply Chain Management, and Commerce, first and foremost is the data management framework. This is the primary tool, and you access it through the data management workspace.

Here you have an administrator managing all data-related activities through data projects using concepts like data entities, data templates, and data packages.

You can use this workspace to run a number of data-related scenarios like copying configurations between environments or loading data as part of data migration.

For more information, refer to:

- Data entities overview
- Configuration data projects
- Configuration data templates
- Configuration data packages

Database operations

Though you can use data entities and data packages to move small configurations, this may not be practical always.

You may often find it handy to move entire databases.

Database movement operations are a suite of self-service actions that can be used as part of data application lifecycle management (DataALM). These actions provide structured processes for common implementation scenarios such as golden configuration promotion, debugging/diagnostics, destructive testing, and general refresh for training purposes.

You can use database movement operations to perform refresh, export, import, and point-in-time restore.

References

Data Management/Data Warehousing information, news and tips - SearchDataManagement (techtarget.com)

Insights-Driven Businesses Set The Pace For Global Growth (forrester.com)

DMBoK - Data Management Body of Knowledge (dama.org)

For example, once your golden configuration environment is ready and all testing cycles have been completed and signed off in SIT and UAT, you can choose "Sandbox to Production" type database request to restore this database to production for go live.

For more info go to:

- Database movement operations home page
- Submit service requests to the Dynamics 365 Service Engineering team

Data sharing framework

Cross-company sharing is a mechanism for sharing reference and group data among companies in a Dynamics 365 Finance, Supply Chain Management, and Commerce deployment.

This framework is introduced to allow sharing setups and master data across multiple legal entities. This facilitates master data management when you are dealing with multiple legal entities and want to designate one legal entity as master for some setups and parameters data. For example, tax codes may not change from company to company so you can set up in one legal entity and use cross company data sharing framework and its policies to replicate the data across rest of the legal entities.

For more information, go to Cross-company data sharing.

Copy company configuration within an existing environment

If the deployment is a phased rollout involving multiple companies within one Dynamics 365 instance, a very good strategy can be to use the "Copy into legal entity" feature. This allows you to designate one base company as the source template from which you can seed multiple other companies or legal entities. All the setups are automatically copied into those destination companies. This way you can templatize your rollouts, allowing you to quickly stand up companies when needed. This strategy needs some upfront planning and efforts in doing the setups, but depending on the scale of a company's rollouts can be quite an effective strategy savings time and reducing errors.

For more information, see Copy configuration data between companies or legal entities overview.

Checklist

Data governance and architecture

- [ ] Establish data governance principles to ensure data quality throughout the business processes lifecycle, focusing on the data's availability, usability, integrity, security, and compliance.
- [ ] Appoint a data steward to ensure data governance principles are applied.
- [ ] Define proper use cases and make data available to support the business processes.
- [ ] Define proper rules and select applications for master data management.
- [ ] Define a proper data architecture that depicts a holistic view of the data repositories, their relationships with each other, and ownership. Capture the data owners, systems, and conceptual flow between systems during your design and analysis phases.

Data modeling

- [ ] Define, clearly document, and keep a data model up to date to serve as a blueprint.
- [ ] Conform the data model to the Common Data Model standard without deviations to ensure cross-application compatibility and future readiness.

Data storage

- [ ] Estimate and forecast data storage needs across different environments and types of data stores used in the solution.

Configuration data and data migration

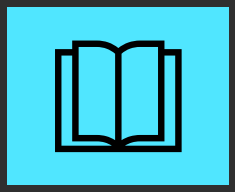
- [ ] Create, maintain, update, and test a configuration plan throughout the project lifetime. It accounts for all the required configuration data you import to support go live.
- [ ] Ensure the data migration analyst, data migration architect, and data steward create a plan for data migration that includes identifying data sources, data mapping, environments, ETL, testing, and cutover planning.
- [ ] Focus on maximizing the data throughput during migration by following Customer Engagement apps best practices.
- [ ] Optimize for network latency by staging in the cloud and batching requests.

Data integration

- [ ] Only store necessary data in the app for key processes that actively interact with it and choose the right type of data store based on the usage.

Data quality

- [ ] Create a realistic view of your data quality and estimate the efforts required to perform the necessary cleanup.
- [ ] Ensure that the apps have the necessary validations and controls to enforce data quality and that you have processes to measure and report on it.
- [ ] Maintain high-quality data by following the principles in this chapter, and have leadership drive the habit of managing data on an ongoing basis.



Case study

Building a rock-solid data foundation, step by step

A global travel company with thousands of staff in the UK, Dubai, and ANZ had been a user of a legacy homegrown IT system that had expanded over the years.

The system was hard to maintain and fraught with issues as customer data was divided in a number of applications and databases.

In order to target productivity, achieve compliance, and reduce costs, the business started a project to move to Dynamics 365 Finance, Supply Chain Management, and Customer Engagement.

The first phase of the project targeted the UK business and involved moving over 150 of their business processes to Dynamics 365.

Since the travel business handles people's personal data, including sensitive data like passport information and medical certificates, they had to be GDPR compliant. The team worked with an experienced system integrator and took a proactive approach to data governance and data security starting with charting out the architectural blueprint of their "to be" design. They wanted to be on top of all data outflows and inflows to make sure there is consistency, integrity, and security of data. There is one source of truth, so they defined apps that held the master data and a companywide policy formulation to let every department know what to find where and the exact process to request, update, and delete a customer's record.

They also built in workflows to manage data retention responsibly, like disposing of information when requested by a customer.

The previous systems were disjointed with data existing in silos. Different departments had sometimes widely different views of customers, which caused confusion and poor service for customers. Call centers were constantly busy as a customer's call was routed from department to department as no one person knew the whole picture.

One of the senior managers remarked that taking a proactive approach to data governance and coming up with a design that put data at the heart of every decision, helped the company to become truly data driven.

With the first phase of the UK deployment successful, the company is now actively working on going live for Dubai and ANZ businesses.

The system integrator also recommended an approach of managing configuration data for deployments. With businesses in three countries, there is lot of commonality in setups in the three legal entities that could be reused. But at the same time, there are peculiarities that need to be maintained separately. Instead of managing all this complexity manually, system integrators had good experience using data templates and packages in Finance and Supply Chain Management. They worked with the company and came up with a configuration plan checklist that identified each of the three businesses' specific setup requirements. This planning exercise paid back multi-fold when the business discovered how smooth, quick, and error free their subsequent deployments became.

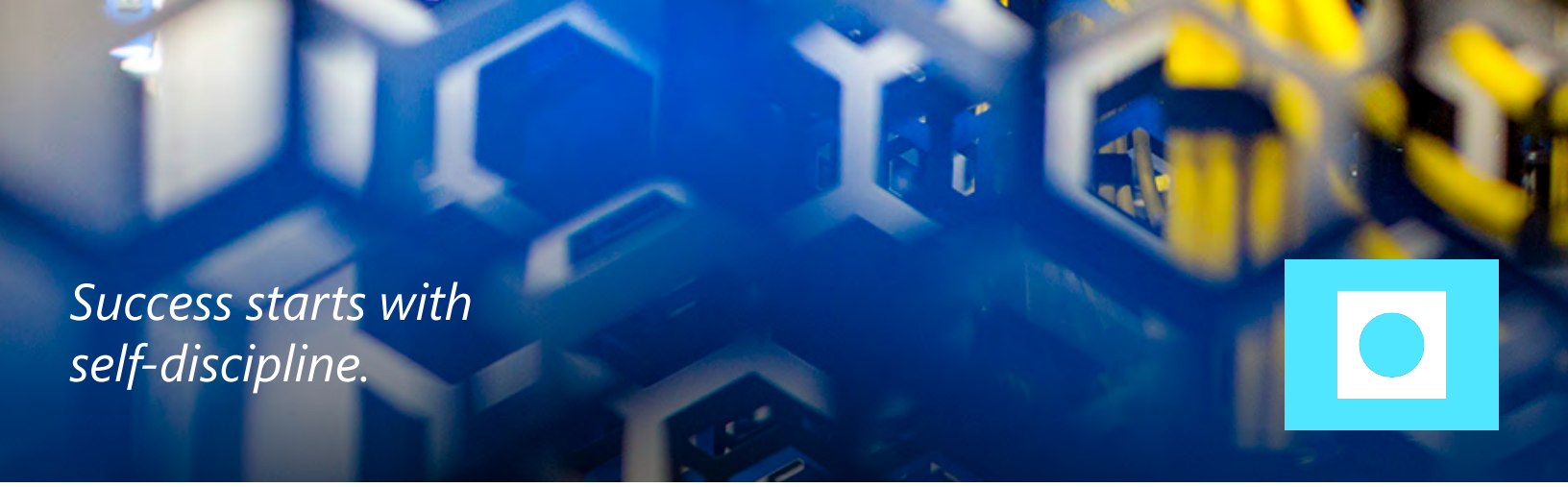
As part of the project, they also moved their historical databases of different applications to Azure Data Lake and connected with Power BI to provide analytical reporting to the end users.

The company now has a unified view of their customers, which helps them provide better customer service and allows marketing efforts to be more targeted.

11

Guide

Application lifecyle management



Introduction

Generally, everything has a life and its own lifecycle. Your solution goes through the same process, starting with conception, moving through implementation, then continuous operation, and finally to transition.

A solid application lifecycle management (ALM) strategy brings a successful solution to customers with complete visibility, less manual interaction with automation, improved delivery, and future planning.

In this chapter, we talk about ALM in the context of the overall solution lifecycle. Your solution may use one or more of the Dynamics 365 Business Applications such as Finance, Supply Chain Management, Sales, Field Service, or Commerce.

With ALM, you have defined processes and practices, a structured team, and tools at your disposal.

ALM is the management of your lifecycle (from conception to operation) of your solution.

ALM includes disciplines that span throughout the entire lifecycle of a solution, such as governance (decision-making), project management, requirement management, architecture, development, test management, maintenance, support, change management, and release management.

Your entire solution lifecycle may go through several iterations. Your application lifecycle may also be part of a larger solution lifecycle. ALM provides defined guidance for people, processes, and tools. It sets the stage for a successful solution outcome. Additionally, innovation and automation are essential for improving the lifecycle of your solution.

What is ALM?

ALM is managing the end-to-end lifecycle of your solution, starting from procuring the Dynamics 365 license, to mapping your business

requirements in your application, designing your solution, extending your custom requirements, validating and testing the solution considering business requirements, deploying the solution to business, and maintaining it through the lifetime of your solution (**Figure 11-1**).

The high-level lifecycle of your solution could be one or many implementations of your solution.

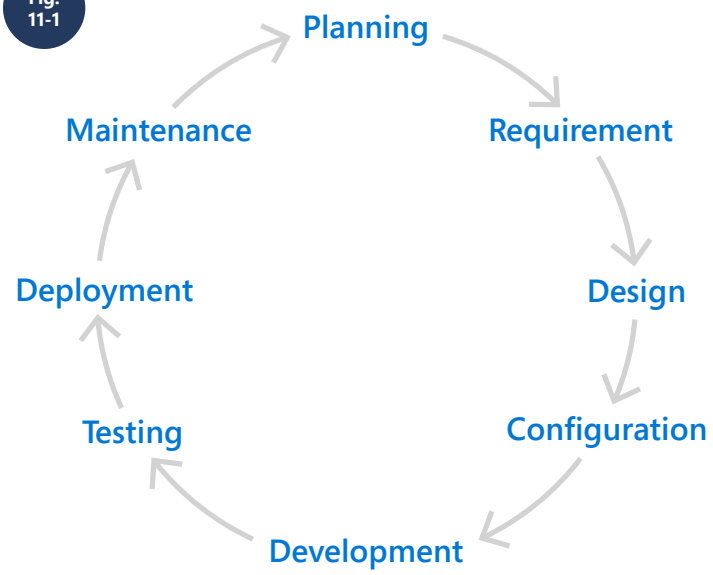
ALM focuses on a few key concepts:

- Defining best practices, processes, and templates for implementation
- Collaboration between customer, partner, and Microsoft team members
- Multiple phase rollouts of your entire solution
- Innovation and automation using tools

Having well-documented processes connected with teams using various tools will result in team confidence, a complete picture of the implementation, and ultimately the success of your solution.

It's very important that the implementation team has the appropriate ALM tooling. ALM tools (such as Microsoft Azure DevOps) are required to manage all aspects of the solution, including application governance, requirement management, configuration, application development, testing, deployment, and support. The ALM tool should be well connected with all team members as well as all processes. For example, when a developer checks in the code, they should mark the changes with a particular work item, which connects the development work item with the business requirement work item to the requirement validation work item.

Fig. 11-1



Application vs. development lifecycle management

Some people may define ALM as improving development processes such as version control, build automation, release deployments. Although this is

part of ALM, we refer to this part of the lifecycle as the software development lifecycle (SDLC), or Dev ALM.

ALM isn't just about the development lifecycle, in which we look at the solution's development activities including design, develop, test, and deploy. Beyond the development lifecycle, ALM covers the entire lifecycle of your implementation, including decision-making, project management, requirement management, the application development lifecycle, test management, and solution maintenance.

Why have an ALM strategy?

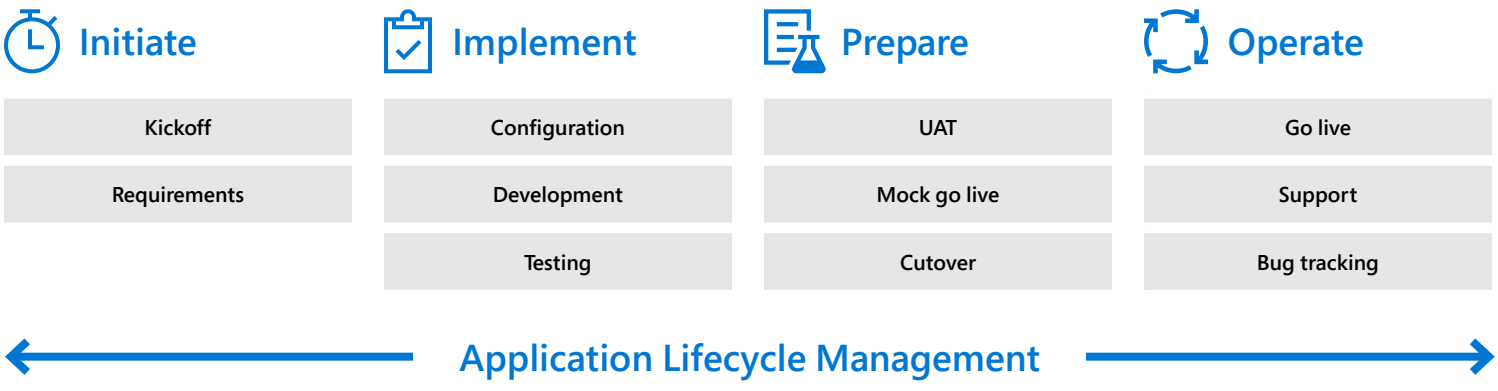
From the time you conceptualize your Dynamics 365 solution, you start the application lifecycle: from the project Initiate phase, to the Implement phase, Prepare phase, and finally the Operate phase (**Figure 11-2**).

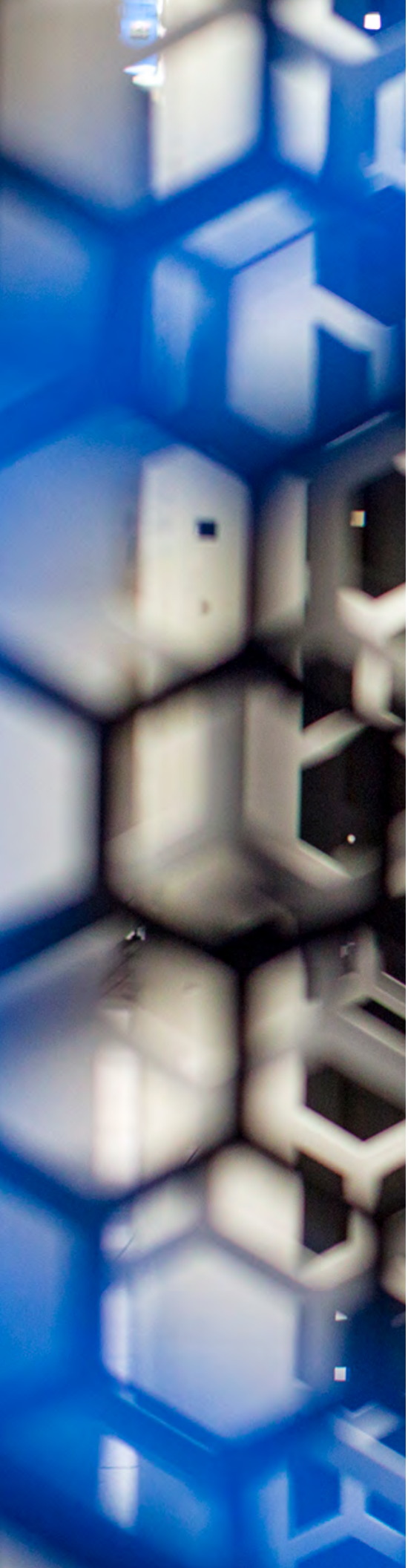
During the lifecycle of the Dynamics 365 solution, you identify partner teams, require project management, gather and map business processes, develop new processes, perform testing, deploy code, and finally maintain the solution in production.

Think about taking a trip. You set dates, book flights and hotels, and plan places to visit. All that planning will likely result in a great vacation.

In the same way, a well-planned ALM will lead to a solution that grows your business. With ALM recommended practices, you're set for success

Fig. 11-2





in your solution implementation. You gain visibility into several areas:

- The current work items (requirement, development, testing)
- The work items completed, in progress, or planned
- The teams that worked, are working, or will work on specific tasks
- Issues and risks associated with solution implementation
- Development best practices
- Code history or code version control
- Build and release automation
- A testing plan, test cases, and test results against requirements

Your ALM may not be perfect right from the start. But it's a foundation—you can refine your ALM practices over time.

Implementation without defined ALM

ALM is the lifeline of your implementation. Let's return to the planned trip. You planned your dates and booked your flights, but you didn't book lodging or decide on any sightseeing excursions or activities. Your trip could be a disappointment, and you might end up spending more time planning when you should be out having fun.

Like a poorly planned trip, if you don't have effective ALM, you can expect a negative impact on your implementation, solution quality, and business satisfaction.

Without effective ALM, you may have poor decision-making, recommended practices may not be followed, and teams are disjointed in the implementation, which can cause delays.

Is ALM only for complex implementations?

You might think that implementing well-defined processes and tracking work items, individuals, and using automation tools is too much work for a small or straightforward implementation.

However, every implementation no matter its size should follow application lifecycle components, such as having a project plan, defined scope, business requirements, and documentation. Your

implementation might not require development or integration, but it will still have configurations and application version updates.

With effective ALM and well-defined practices, you can keep your solution healthy and up to date with the latest application releases.

Steps to successful ALM

You should use ALM from solution conception through operation. In this section, we dive into some of the areas of the ALM process, including before and after implementation, project management, process management, other configurations, development, testing, and finally maintenance and support.

During implementation

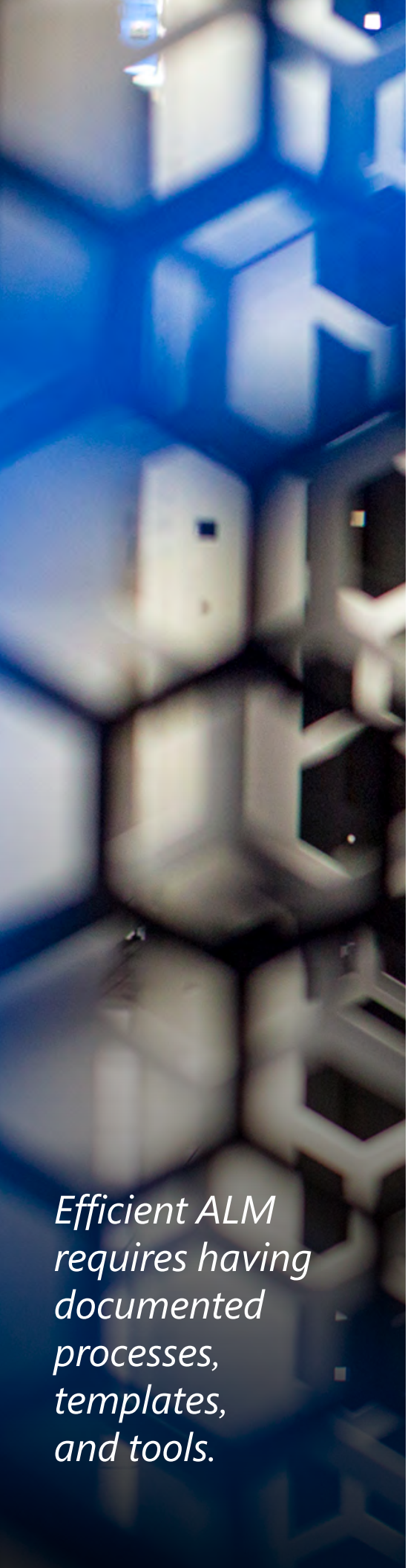
While you're implementing the solution, you go through multiple phases: Initiate, Implement, and Prepare. ALM applies to all these aspects in your implementation:

- Project management
- Business process management
- Application configuration
- Development
- Testing
- Bug tracking
- Ideas, issues, risks, and documents
- Release management

After implementation

When you're live in production, you're in the Operate phase. ALM continues with the following aspects:

- Continuous updates
- Independent software vendor (ISV) updates
- Maintenance and support
- New features
- Next phase



Efficient ALM requires having documented processes, templates, and tools.

Project management

Efficient ALM requires having documented processes, templates, and tools for all project management activities, such as project planning, cost management, team management, ideas for improvements or solutions, issues, and risk management. A project manager performs their responsibilities efficiently when these areas are defined and documented. For example, you should have a template for the project plan that the project manager can apply in the Implement phase.

Chapter 8, "Project governance," discusses project management in more detail.

Business process management

ALM processes and tools should include managing business requirements and processes efficiently and effectively. After you define your templates and tools, provide a framework for functional team members to work together to define requirements and business processes efficiently. Teams can track business requirements for business processes and connect further with configuration and development.

For example, a functional team member can gather and define business requirements in a given template and track them in Azure DevOps. They can have it reviewed by assigning a work item to business users and then store the business requirement document in a repository.

Chapter 7, "Process-focused solution," covers requirement management in detail.

Application configuration

ALM processes and tools should also include managing application configurations. Before configuring an application, the business should define and review business processes and requirements. Functional team members should have defined processes with tools to perform fit gap analysis, and a defined configuration required in the application. It's important to track your configuration setup and changes using

tools such as Azure DevOps. Typically, configuration is a one-time activity without documentation and tracking. This leads to confusion, no visibility, and questions about why particular decisions were made. You should always maintain your documentation for application configuration during testing, user acceptance testing (UAT), cutover, and even after go live during maintenance and support.

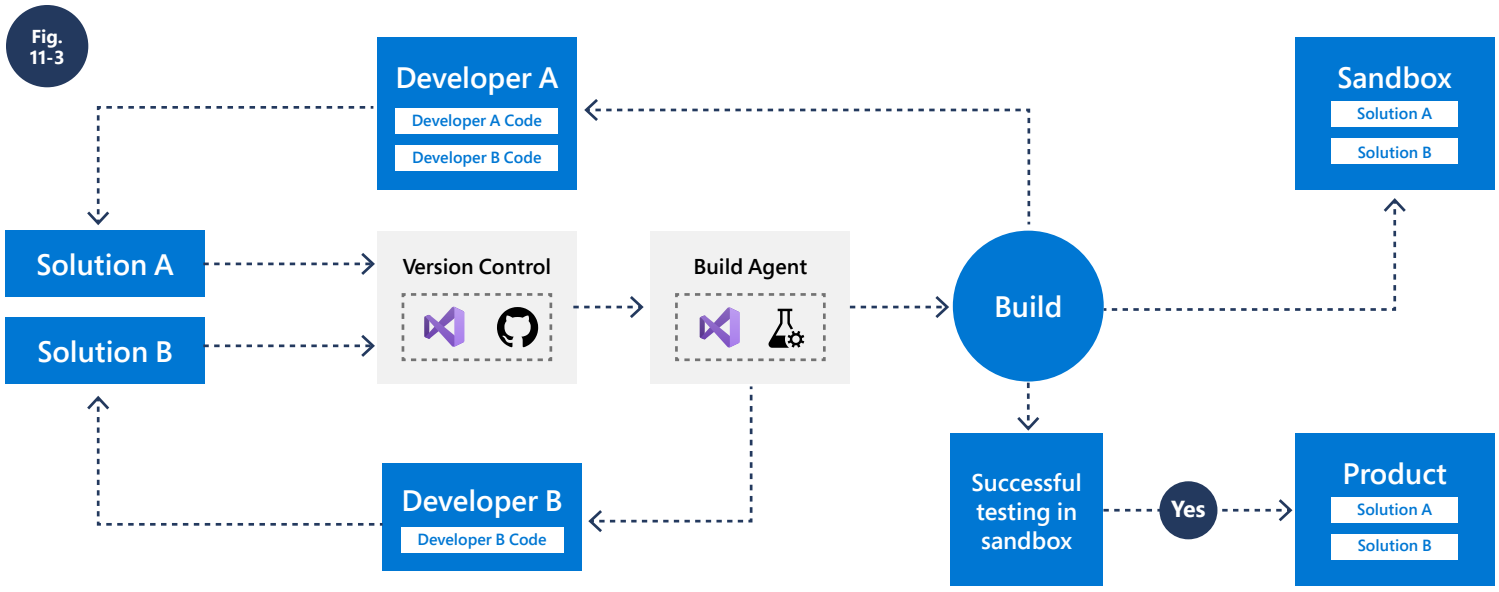
Development

Having an efficient development lifecycle is one of the mandatory aspects of ALM. In general, the development lifecycle consists of the following: Design, Develop, Build, Test, and Deploy. Continuous integration and continuous deployment (CI/CD) is one of the best and latest practices to enable delivering code changes more frequently and reliably.

After the business requirements are defined and identified, the development team should get involved to work on any gaps in the solution. The development team analyzes the requirements, reviews the functional design, prepares the technical design, and gets the technical design reviewed. The development team should use version control, create development tasks, link check-ins with work items, and prepare a unit-testing document.

Finally, use build definitions and pipelines for automation and to identify and fix any build issues. Build automation is vital for continuous integration or gated check-ins. Manual builds are error-prone and time-consuming. The build process, either manual or automated, is mandatory for development.

The following example of a CI/CD approach (**Figure 11-3**) has a centralized version control in place, a build agent using build definitions, and release pipelines to push code to a sandbox or other environments. In this scenario, Dev A develops Solution A and Dev B develops Solution B. Dev A gets Solution B (Dev B code) using version control and after a successful build automation. When the build is successful, the code is released to the sandbox environment using the release pipelines. After successful testing, the code is released to the production environment.



Testing

Testing is an integral part of ALM. Under ALM, test management processes and tools should be defined with templates to help manage test preparation, implementation, and reporting. It should also include what tools to use for these steps. Chapter 14, "Testing strategy," provides more information about test management.

Maintenance and support

In addition to the steps taken for planning, developing, and deploying your solution, as part of effective ALM, you should have a plan to make sure your solution continues to be optimal and healthy.

Chapter 20, "Service the solution," and Chapter 21, "Transition to support," cover maintaining the solution and the support process in detail.

ALM strategy for Dynamics 365

ALM can improve every aspect of your Dynamics 365 solution. Dynamics 365 offers various tools to achieve a successful ALM in your implementation.

Team responsibility

Various team members are involved in an implementation, with separate roles and responsibilities. Multiple teams such as the customer, partner, and ISV work are involved and must work together.

Every team member, either directly or indirectly involved, should own the ALM maintenance of the processes and tools they're using. For example, the project manager owns and maintains the project plan, and the tester owns and maintains the test plan. You should also have cross-team collaboration for maintaining ALM practices.

Azure DevOps

Microsoft recommends using Azure DevOps as the tool for managing or maintaining your ALM practices and processes. For some areas of Dynamics 365 (such as Finance and Supply Chain Management), Azure DevOps is the only version control tool.

In addition to the standard templates in Azure DevOps, you can build your own custom templates according to your business needs. Azure DevOps lets you store all your work items, including features, user stories, requirements, tasks, bugs, and more.

A standard lifecycle is available for each work item, and you can build your own custom states and rules for each type of work item.

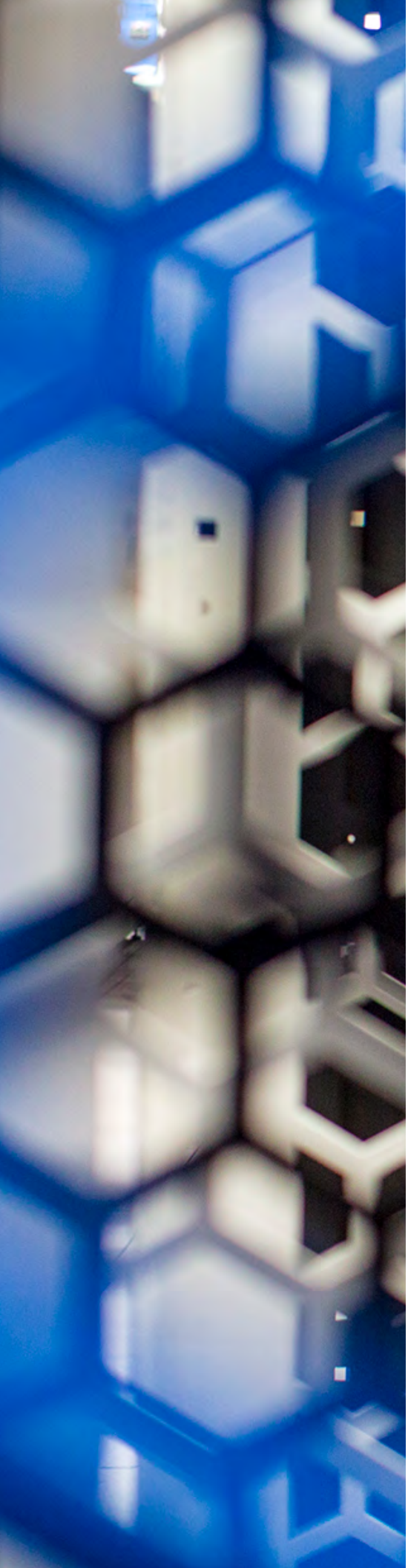
Dynamics 365 also recommends Azure DevOps as the version control management tool for your code. You can achieve build automation and release pipelines using build definitions and release definitions.

Take some time to learn more about DevOps tools on Azure.

You can also use Azure DevOps for project management, test management, bug tracking, release management, and many more aspects of your implementation.

Operations

In this section, we provide some Finance and Supply Chain Management recommendations to achieve an efficient ALM in your



solution. We'll cover things such as Microsoft Dynamics Lifecycle Services, business process modeling, development, and version control.

Microsoft Dynamics Lifecycle Services

Microsoft Dynamics Lifecycle Services (LCS) is a cloud-based solution in which all participants (customer, partner, and Microsoft) work collaboratively, and you manage your implementation from pre-sales, through implementation, and finally the Operate phase. LCS provides various checklists and tools to help you manage your project efficiently:

- A single location for all your projects
- Project onboarding
- Project methodologies
- Azure DevOps configuration
- Business process modeler (BPM)
- Cloud-hosted environments (CHE)
- Data application lifecycle management (DataALM)
- Management of Microsoft support
- Customization analysis
- Subscription estimator
- Issue search
- Continuous updates
- Service requests to Dynamics Service Engineering (DSE)
- Environment monitoring and diagnostics
- Asset library

The goal of LCS is to deliver the right information, at the right time, to the right people, and to help ensure repeatable, predictable success with each rollout of an implementation, update, or upgrade.

Business process modeler

Business process modeler (BPM) provides abstract explanations of your solution's business processes. Business processes are the underlying foundation for application configuration and application development.

Effective ALM provides defined, documented, and templated processes to follow in your implementation. You can model your business processes in Dynamics 365 Finance and Supply Chain Management using BPM under LCS.

You can use the BPM tool to define, view, and edit the Finance and Supply Chain Management out-of-box business processes (in the form of libraries), which you can use in future implementations. The tool helps you review your processes, track the progress of your project, and sync your business processes and requirements to Azure DevOps.

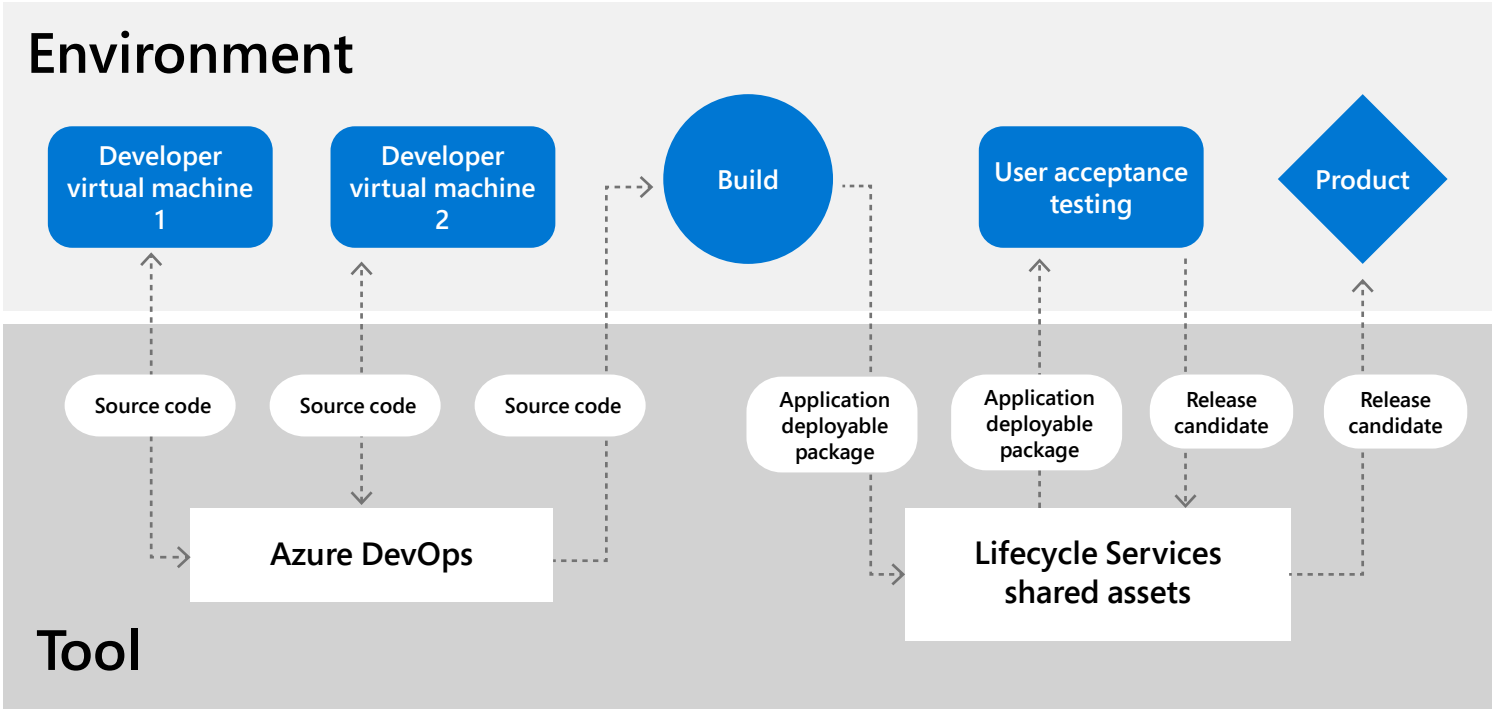
Every business is different in some ways; this tool helps you align your business processes with your industry-specific business processes and best practices. BPM libraries provide a foundation for your business process, and you can add, remove, and edit your processes according to your solution requirements.

Development

In Finance and Supply Chain Management, Microsoft Visual Studio is used as the development environment. The development lifecycle includes the following steps (**Figure 11-4**):

- Each developer uses their own development environment
- Developers write source code and check in their code to Azure DevOps
- Developers also sync code from Azure DevOps to get the source code from other developers
- The build takes the source code from Azure DevOps, uses the build definition, and creates a deployable package

Fig. 11-4



- The build pipeline also pushes the deployable package to the LCS asset library
- Azure release pipelines work with Visual Studio to simplify deploying packages to UAT
- When UAT is complete, the deployable package is marked as a release candidate to deploy to production
- The Dynamics Service Engineering (DSE) team deploys it to production using a service request from LCS

Version control

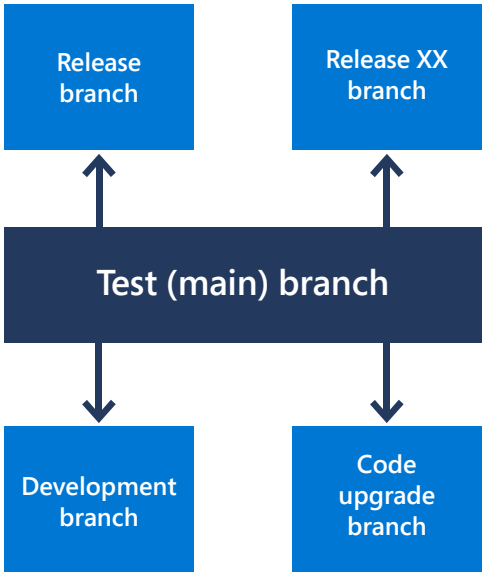
The primary purpose of version control is storing and maintaining source code for customizations, as well as ISV solutions. You develop against local, XML-based files (not the online database), which are stored in version control tools such as Azure DevOps. The following are recommendations for version control branching:

- Consider a using minimum branching option
- Consider the following recommended branching strategy:
  - **Development** Developer check-in and testing with development data (Trunk/Dev)
  - **Test** Deploying to the tier 2+ and testing with current production data (Trunk/Main)
  - **Release or Release XX** Retesting in the tier 2+ and deploying to production (Trunk/Release) or v-next (Trunk/Release XX)
- Use the shelve command or suspend changes to keep work safe
- Request a code review to ensure code quality
- Check in code when a feature is complete, and include changes from one feature in each changeset
- Merge each feature in a separate changeset
- Don't check in code directly to the test or release branches
- Don't check in changes for more than one feature in a single changeset
- Don't mark deployable packages from the development and test branches as release candidates
- Don't merge untested features into the release branch

Figure 11-5 illustrates a recommended branching strategy.

For more information, take a look at our document on how to develop and customize your home page.

Fig. 11-5



Arrows show the direction when creating the branch

To learn more about build automation, check out our guides on how to deploy and use a continuous build and test automation environment, create a build automation that uses Microsoft-hosted agents and Azure Pipelines, and update model versions in an automated build.

Build automation

Speed is essential for rapid implementation, and build automation is the key to achieving this.

The build process is mandatory for any code to run. This process involves compiling the source code and producing binary files (assemblies). A database sync also requires a build first because the schema is retrieved from the assemblies (and not the XML files).

The Azure DevOps build system provides the following triggers for builds:

- Scheduled builds, such as nightly at 6 PM
- Continuous integration, such as:
  - Starting a build as soon as code is checked in
  - Gated check-in
- Manual builds (on demand)

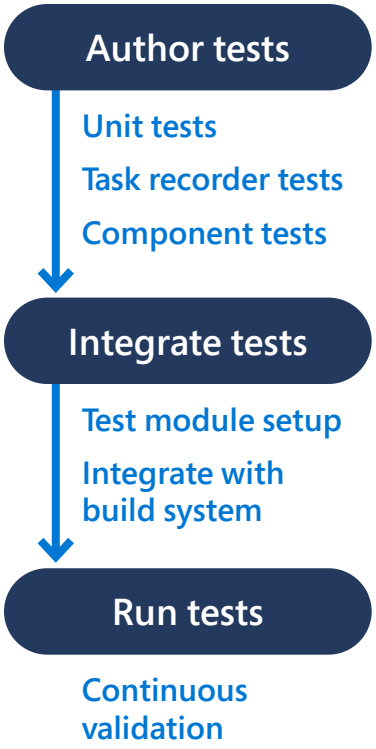
Here's a checklist to ensure your build goes smoothly:

- Make sure the code compiles without errors
- Don't use build environments for development activities
- Create a build definition for each branch and for each build trigger
- Consider using continuous integration or gated check-in build triggers
- Perform automated testing after successful builds
- Create a single deployable package that contains all packages
- Don't use Visual Studio to create a deployable package, for the following reasons:
  - Deployable packages contain all the code on the development machine, including experimental code and incomplete features
  - Deployable packages may not include all the modified packages, which means they're not independent and testing results could vary
- Create a deployable package with a build machine so it's self-contained and contains all applicable modules
- Make sure to keep service updates on the build machine less than or equal to production
- Follow a consistent naming convention for your deployable package

Automated testing

As part of development ALM, testing automation should be in place

Fig. 11-6



to achieve fast-moving code from development to deployment. In Finance and Supply Chain Management, you can integrate testing and validation two different ways:

- Unit and component level testing using SysTest framework
- Automated testing using Task recorder and the Regression suite automation tool (RSAT)

To keep up with innovation and constant changes in your solution, it's critical to invest and build in continuous validation. Achieving this requires different components (**Figure 11-6**).

The RSAT significantly reduces the time and cost of UAT for Finance and Supply Chain Management applications. RSAT lets functional super users record business tasks by using Task recorder and converting the recordings into a suite of automated tests, without having to write source code.

Deployment

Let's review the key concepts for Finance and Supply Chain Management deployment:

- **Deployable package** A unit of deployment that can be applied in an environment
- **Deployment runbook** A series of steps that are generated to apply the deployable package to the target environment
- **AX Installer** Creates a runbook that enables installing a package

You can deploy a package manually or through automated deployment (which isn't applicable to production). We recommend the following when deploying Finance and Supply Chain Management applications:

- Consider using automated deployment to achieve continuous deployment.
- Make sure that the deployable package that should be applied is valid.
- Make sure that the package is applied in a sandbox environment before it's applied in the production environment.
- If you want to apply multiple packages, create a merged package that can be applied first in a sandbox environment and then in the production environment.

For more information about automated testing, we offer tutorials on testing and validations, using the regression suite automation tool, and acceptance test library resources.

For more information about deployment, refer to our guides on how to create deployable packages in Azure Pipelines, apply updates to cloud environments, troubleshoot package application issues, and uninstall a package.

- Production deployment is manual. After successful UAT, mark the deployable package as a release candidate and schedule a service request with the DSE team.
- Clean up the old deployable package periodically from the LCS asset library.

Customer engagement

When organizations implement their software as a service (SaaS) cloud solution, some level of customization and extensibility typically exists, with the goal to provide additional value and adjust the functionalities to specific business or industry needs.

The following concepts are important to understanding ALM when using the Power Platform.

Solutions

Solutions are the mechanism for implementing Dev ALM in Power Platform apps. They're the vehicle that distributes components across the different environments.

A component represents what can be customized within the solution, such as site maps, applications, the components of the data model, forms, charts, or plug-ins.

Solutions can either be managed or unmanaged. You should use unmanaged solutions in development environments while changes are still underway. Managed solutions are used to deploy to any environment that isn't a development environment for that solution.

You should have a good understanding of the solution concepts in the Power Platform—the solution is the baseline for the components across their lifecycle. For more information, refer to our overview of solution concepts.

Tools

Several tools are available to help automate the process of managing and shipping solutions, which can help increase efficiency, reduce

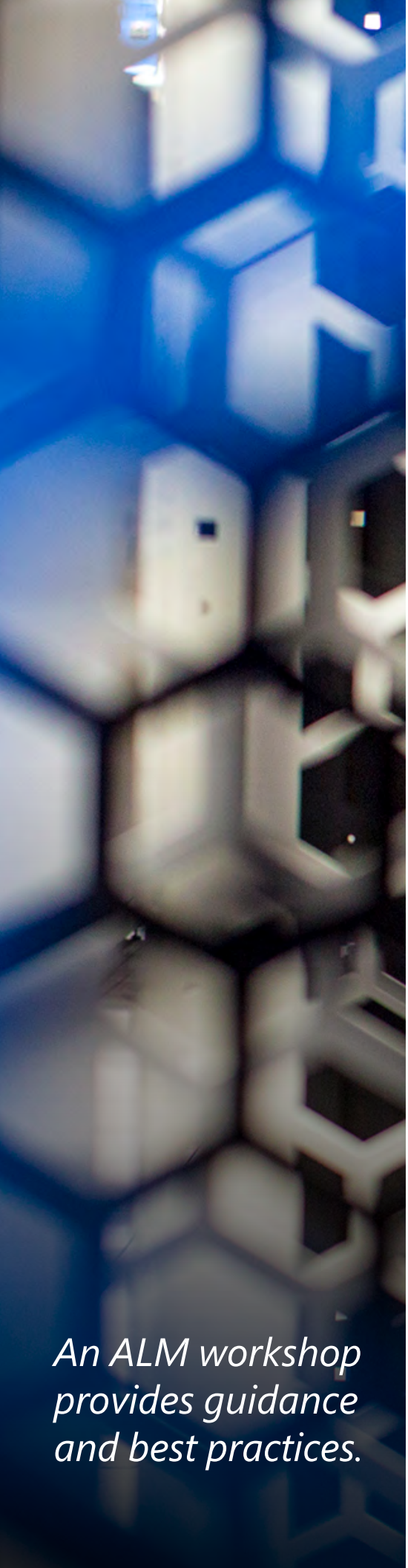
manual labor, and reduce human error when working with solutions throughout their lifecycle.

Check out our additional resources for Microsoft Power Platform Build Tools for Azure DevOps, Power Apps build tools for Azure DevOps, and key concepts for new Azure Pipelines users.

Applications like Azure DevOps provide developer services that support the activities required to plan, store, collaborate, build, and deploy. This includes synchronization of solution metadata between development environments and your version control system, generating build artifacts, deploying to downstream environments, provisioning or de-provisioning environments, and performing static analysis checks against your solution by using the Power Apps checker service.

Consider using the following tools:

- **Version control system** This category of software tools helps record changes to files by keeping track of changes committed to software code. A version control system is a database of changes, which contains all the edits and historical versions of a software project. Version control systems allow you to maintain a single source of truth and recall specific versions when needed. Git is a popular example of a version control system.
- **The Configuration Migration tool** This tool enables you to move configuration and reference data across environments. Configuration and reference data is different from user and transactional data, and is used to define custom functionality in apps based on Dataverse.
- **Package deployer** The package deployer lets administrators or developers deploy comprehensive packages of relevant assets to Dataverse environments. Packages can consist of not only solution files, but also flat files, custom code, and HTML files.
- **Solution packager** This tool can unpack a compressed solution file into multiple XML files and other files, so they can be easily managed by a source control system.
- **The Power Apps CLI** The Power Apps CLI is a simple, single-stop developer command-line interface that empowers developers and app makers to create code components.
- **PowerShell cmdlets** The PowerShell cmdlets for administrators, app makers, and developers allow automation of monitoring, management, and quality assurance tasks that are possible through the Power Apps admin center user interface.



An ALM workshop provides guidance and best practices.

ALM workshop strategy

The implementation team should plan for an ALM workshop before project initiation or before project kickoff. The customer and partner should gather the required processes, tools, and teams before project kickoff, such as:

- Project plan template
- Business requirement document template
- Functional and technical spec templates
- Test plan and test suite template
- Defined tools such as DevOps template
- Defined roles required for partner and customer teams

In addition to this workshop, the implementation team should plan for a Dev ALM workshop before starting development activities. Our FastTrack ALM workshop focuses on your development lifecycle and provides guidance and best practices such as development environments, version control, build automation, and release pipelines. It underlines the importance of having fast builds, quick testing, and fast deployments using automation, which helps ensure a rapid development lifecycle, especially for agile development methodology.

Workshop scope

FastTrack ALM workshops are designed for implementers who want to make sure that their development approach meets the requirements of the implementation and is aligned with typical best practices. The workshop could cover several topics:

- **Development work management** Focuses on high-level, day-to-day developer activities. It reviews that the team has development guidelines, development best practices, and work items such as requirement, tasks, and bugs management.
- **Code management** Reviews your version control, branching, code merging, and code reviews strategy. Branching can be simple or complex depending on the project and phases. It also looks at how customization is checked in, such as gated check-in.
- **Build management** Looks at your build strategies, such as

manual build or automated build. It mainly reviews your build plan for build definitions, different variables, and build triggers for your implementation. It also reviews whether you're using a build environment or Azure-hosted builds.

- **Release management** Assesses your deployment plan for customizations, ISVs, and continuous (one version) updates. It also reviews how hotfixes and your code release, or one version updates, are applied to production and nonproduction environments.

Timing

It's important to ensure ALM workshops are conducted at the right time. Ideally, the ALM workshop should be before project kickoff, and the Dev ALM workshop should be before starting development or sooner.

You should complete both workshops either in the Initiate phase or when starting the Implement phase. If you plan ALM workshops too far into implementation, any findings and recommendations could cause significant rework.

Conclusion

ALM is the management of your solution lifecycle from conception to operation. It includes governance (decision-making), project management, managing requirements, solution architecture, development, CI/CD, test management, maintenance and support, change management, release management, and many more areas.

Your solution lifecycle may go through several evolutions. Each evolution may use the same SDLC methodology. Basically, SDLC is part of ALM, ALM is far bigger than Dev ALM, and Dev ALM is part of SDLC.

References

DevOps and Azure Pipelines

DevOps solutions on Azure

Release pipelines

Sign up for Azure Pipelines

Operations

Implement application lifecycle management in Finance and Supply Chain Management apps

Business process modeler (BPM) in Lifecycle Services (LCS)

Dynamics 365 Finance and Supply Chain Management Tools

Customer Engagement

Application lifecycle management (ALM) with Microsoft Power Platform

Create your first pipeline

Microsoft Dynamics CRM SDK Templates

Solution Lifecycle Management: Dynamics 365 for Customer Engagement apps

Checklist

- [ ] Establish the ALM at the onset and align it to the environments, source control, implementation methodology, and automation strategy.
- [ ] Ensure the ALM provides an efficient, reliable, robust, and automated mechanism to smoothly deploy the configurations and customizations across different environments, including peripheral solution components.
- [ ] Ensure ALM strategy enforces a source control-centric approach that enables creation of new environments with the latest configurations without the need to replicate existing environments.
- [ ] Use a build automation and CI/CD pipeline for the project.
- [ ] Consider the structure of the implementation team to enable effective collaboration while providing exclusive control to developers where needed.
- [ ] Follow documented best practices in general and specifically to each app.
- [ ] For Customer Engagement projects, download the whitepaper Solution Lifecycle Management: Dynamics 365 for Customer Engagement and complete the self-assessment.

Case study

Global transport systems company finds ALM strategy to be a cornerstone of implementation

A global transport systems company embarked on a multiple-phase, multiple-region implementation of Dynamics 365 Sales, Customer Service, and Field Service with an aggressive schedule for achieving a minimum viable product (MVP) go live.

To give business stakeholders a taste of the solution before the official project kickoff, the customer asked their Dynamics 365 implementation partner to lead a rapid prototyping phase, which the customer referred to as "Phase 0." The Phase 0 team included technical salespeople, as well as consultants who would leave when the project kicked off.

Considering Phase 0 to be a rapid prototype, the implementation team didn't see a need to focus on governance and quality. To meet the deadlines, the team made several quick decisions without assessing the impact on the overall application management:

- Staffing Phase 0 with resources who wouldn't remain part of the project team.
- Developing the prototype using separate, unmanaged solutions with different publishers for Sales, Customer Service, and Field Service.
- Pivoting to managed solutions for test and production environments too late in the design phase, after unmanaged solutions were already built and deployed to these environments during Phase 0.

- Merging the Sales, Customer Service, and Field Service solutions into a single "workstream" solution during the build phase of the implementation lifecycle.
- Resolving errors too quickly during solution deployments because of pressure from project leadership, instead of taking the time to understand each problem's root cause. (Such pressure led to developers customizing directly within the test environment.)

As the project team prepared for the go live, the seemingly independent decisions made during the initial phases resulted in deployment issues that eventually stalled the go live. Investigations by Microsoft Support confirmed that there was no ALM strategy in place, and identified the key issues:

- Unmanaged solutions caused solution-layering issues for testing and production, affecting the sanctity and integrity of the environments
- A prototype solution employed for production purposes introduced quality issues
- Failure to use ALM practices such as DevOps caused traceability issues and prompted developers to build functionality that wasn't aligned with customer requirements
- Suboptimal code was implemented because tools such as solution checker weren't used to enforce code quality
- Testing without traceability was insufficient, and buggy code was deployed to other environments

As the old saying goes, "By failing to prepare, you are preparing to fail." The MVP go-live date was delayed by 12 weeks and Microsoft worked alongside the project team to determine the root cause of each issue. The team eventually acknowledged that a series of seemingly unrelated decisions affected the MVP go live, and they sent every team member working in a technical role to an ALM refresher training. The project team also confirmed plans that should have been solidified at the beginning of the project, including an environment strategy and a solution management and ALM approach.

After the refresher training, the project team started with a "crawl to walk" approach. During the "crawl" stage, they implemented mandatory ALM practices with these governance elements:

- Cleaning up their production and test environments, and moving from unmanaged to managed solutions
- Implementing a build process to deploy builds from development to test to production
- Establishing a governance process to restrict developer access to production and test environments
- Adding a bug-triaging process that allowed the development team to troubleshoot and fix issues in development environments and use the build process to deploy fixes in higher-tier environments
- Mandating the generation of solution checker reports

Once these practices were in place, the implementation team moved toward a partially automated ALM process that included setting up DevOps pipelines to automate build deployments and auto-generate solution checker reports. As this case study illustrates, an ALM strategy is a cornerstone of the entire implementation—whether it's a fresh implementation or a rapid prototype elevated to a production-use solution.

12 Guide Security

Introduction

In this chapter, we look at the fundamental security principles applicable to Microsoft Dynamics 365 implementations.

Next, we discuss in more detail how some of these principles apply differently to Dynamics 365 Customer Engagement, Dynamics 365 Finance, and Dynamics 365 Supply Chain Management applications. We then address the importance of making security a priority from day one, with specific examples from each product that build upon the concepts we've discussed. Finally, we look at how to avoid common mistakes by examining some key anti-patterns.

Security overview

Security is the protection of IT systems and networks from theft or damage to their hardware, software, or data and disruption of the service.

Dynamics 365 is a software as a service (SaaS) offering from Microsoft. In a SaaS service, data and applications are hosted with a provider (in this case, Microsoft) and accessed over the internet. In this deployment model, the customer maintains ownership of the data, but shares application control with the provider. Therefore, security, compliance, privacy, and data protection are shared responsibilities between provider and customer.

Fig. 12-1

Figure 12-1 illustrates the areas of responsibility between customer and Microsoft based on deployment type.

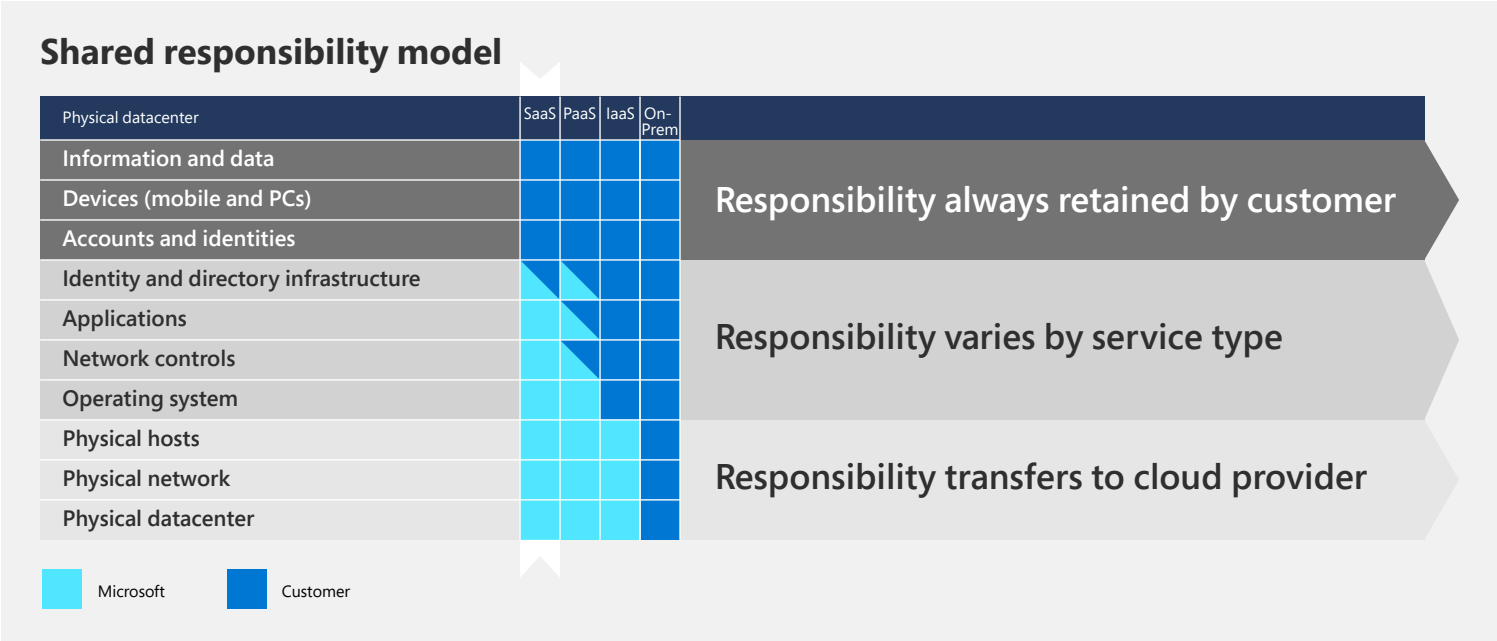


Fig. 12-2

Microsoft takes its commitment seriously to safeguard customers' data, to protect their right to make decisions about that data, and to be transparent about what happens to that data. On our mission to empower everyone to achieve more, we partner with organizations, empowering them to achieve their vision on a trusted platform. The Microsoft Trusted Cloud was built on the foundational principles of security, privacy, compliance, and transparency, and these four key principles guide the way we do business in the cloud (**Figure 12-2**). We apply these principals to your data as follows:

- **Security** Implement strong security measures to safeguard your data
- **Privacy** Protect your right to control and make decisions about your data to help keep it private
- **Compliance** Manage your data in compliance with the law and help you meet your compliance needs
- **Transparency** Be transparent about our enterprise cloud services and explain what we do with your data in clear, plain language

Microsoft has been defending against threats and providing security protections for our online services since 1994, and we invest over $1 billion dollars per year to continue our commitment to protecting our customers. For these reasons, we say that our products and services

run on trust. You can place your trust in our expertise. This confidence allows you to focus on running your business. Ninety-five percent of Fortune 500 businesses run on the trusted Microsoft Cloud.

Compliance

Every organization must comply with the legal and regulatory standards of the industry and region they operate in, and many are also subject to additional contractual requirements and corporate policies. **Figure 12-3** lists some standard compliance goals and their implementation in Dynamics 365.

Fig. 12-3

Compliance goals

Define and document standard operating procedures that meet multiple certification requirements

Run the service in a compliant fashion and collect evidence

Control and audit access to environments as well as actions to data

Implementation details

Adhere to strict privacy and security practices when building features and when operating the service

Pass internal and external audits

Prioritize cross-industry certifications

Authentication and authorization

Microsoft is responsible for the platform, including the services it offers, and provides a cloud service that can meet or exceed the security, privacy, and compliance needs of your organization. Microsoft complies with data protection and privacy laws applicable to cloud services, and our compliance with world-class industry standards is verified. Detailed information about compliance for our cloud services and solutions that help organizations meet regulatory requirements for data security is available at our online Microsoft Trust Center.

As noted earlier, compliance is a shared responsibility. To comply with laws and regulations, cloud service providers and their customers enter a shared responsibility to ensure that each does their part. Tools available at the Trust Center include compliance offerings that help you comply with national, regional, and industry-specific requirements governing the collection and use of data, and audit reports that help you verify technical compliance and control requirements. Specific tools you can access at the Trust Center include:

- Compliance Manager, a cross-Microsoft cloud services solution designed to help organizations meet complex compliance obligations like the General Data Protection Regulation (GDPR). It performs real-time risk assessment that reflects compliance posture against data protection regulations when you use Microsoft cloud services. It also provides recommended actions and step-by-step guidance.
- Service Trust Portal, which contains details and documents such as

whitepapers, ISO reports, and other resources detailing Microsoft's implementation of controls and processes that protect cloud services and customer data.

Customer responsibility

As a customer, you're responsible for the environment after the service has been provisioned. You must identify which controls apply to your business and understand how to implement and configure them to manage security and compliance within the applicable regulatory requirements of your nation, region, and industry.

Refer to Microsoft compliance offerings for more information about regulatory compliance standards and Microsoft products.

Privacy

You are the owner of your data; we don't mine your data for advertising. If you ever choose to end the service, you can take your data with you.

Fig. 12-4

Figure 12-4 lists some standard privacy goals and their implementation in Dynamics 365.

Privacy goals

You own your data

You know where your data is located

You control your customer data

Implementation details

All data is classified

Role-based security puts the customer in charge

Your own logically isolated data repository helps maximize the security and integrity of your data

Documented data-handling practices in the Microsoft Online Services in the Trust Center

How we use your data

Your data is your business, and you can access, modify, or delete it at any time. Microsoft will not use your data without your agreement, and when we have your agreement, we use your data to provide only the services you have chosen. We only process your data based on your agreement and in accordance with the strict policies and procedures that we have contractually agreed to. We don't share your data with advertiser-supported services, nor do we mine it for any purposes like marketing research or advertising. Learn more about how Microsoft categorizes data in the delivery of online services.

We believe you should have control over your data. The Trust Center can tell you more about how we handle data requests from government and law enforcement agencies.

Customer responsibility

As a customer, you're responsible for data classification, identity management, and assigning appropriate security roles to secure the data.

Security

Security is a shared responsibility in a SaaS deployment. This means that some aspects of security are shared by both the customer and the provider, other aspects are the responsibility of the customer, and others are the responsibility of the provider. For Dynamics 365 deployments, Microsoft as the cloud provider is responsible for security aspects including physical datacenter security, the operating system, network controls, and providing a secure application framework (**Figure 12-5**).

Figure 12-6 lists some standard compliance goals and their implementation in Dynamics 365.

Fig. 12-5

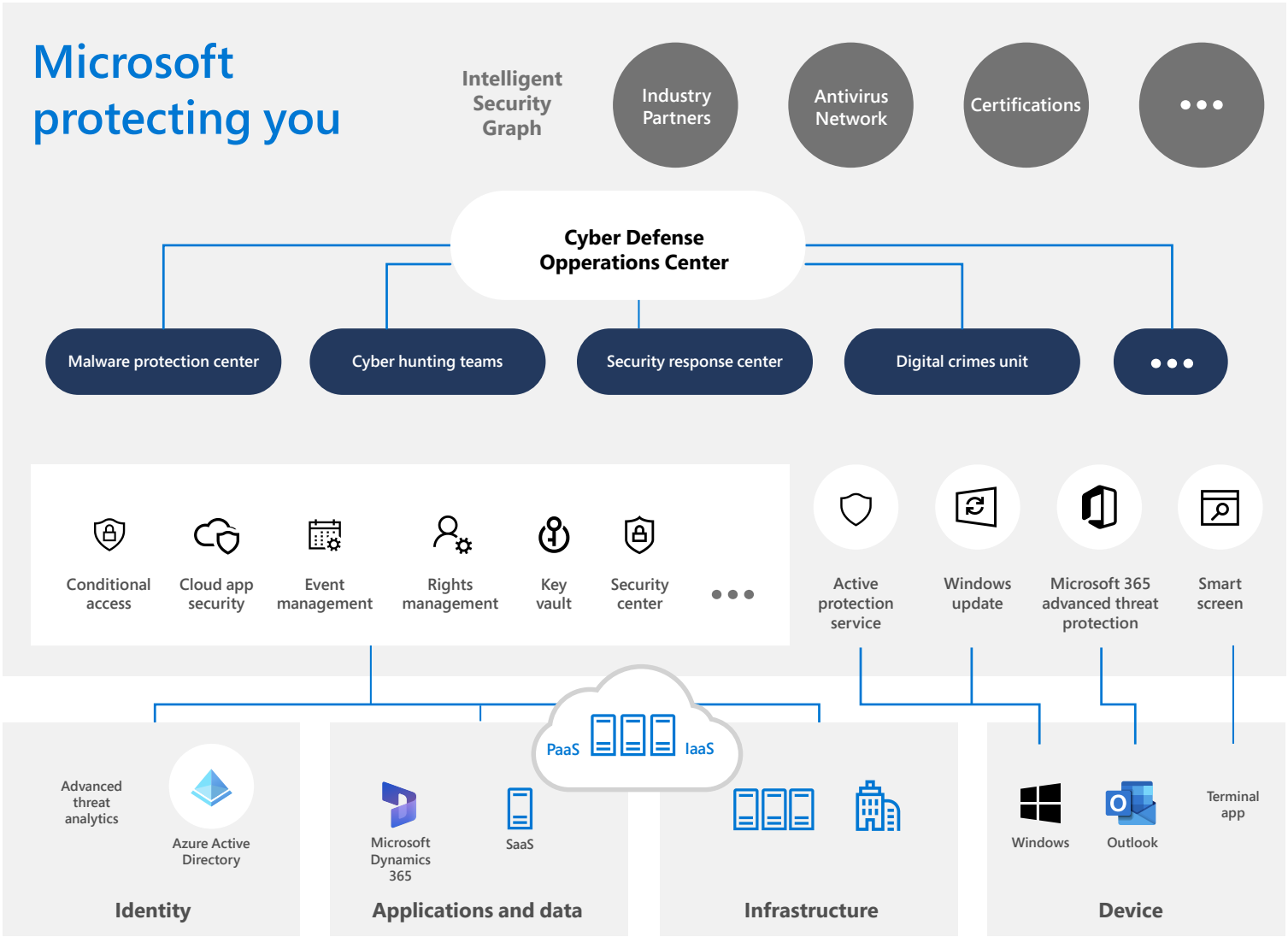


Fig. 12-6

Security goals

Safeguard data using state-of-the-art security technology and processes of the industry

Use the same identity platform as Microsoft 365, so users have the same username and password for all

Control who can access what data

Customer data is isolated from other tenants

Implementation details

State-of-the-art physical security measures

24/7 incident response teams to mitigate threats and attacks

Encryption of data transferred between our datacenters and at rest

Single sign-on federated through Azure Active Directory

Role-based security

The following is a list of core security controls available in Dynamics 365:

- Security Development Lifecycle
- Datacenter security
- Data segregation
- Encryption
- Secure Identity
- Authorization
- Auditing and monitoring

Security Development Lifecycle

The Security Development Lifecycle (SDL) is a process that helps developers build more secure software and address security compliance requirements by introducing security and privacy considerations throughout all phases of the development process. It consists of the following phases: training, requirements, design, implementation, verification, release, and finally response (**Figure 12-7**).

Datacenter security

Microsoft designs, builds, and operates datacenters in a way that strictly controls physical access to the areas where your data is stored. Microsoft understands the importance of protecting your data and is committed to helping secure the datacenters that contain your data. We have an entire division at Microsoft devoted to designing, building, and operating the physical facilities supporting Azure, which delivers Dynamics 365 (**Figure 12-8**). This team is invested in maintaining state-of-the-art physical security.

Fig. 12-7

Training	Requirements	Design	Implementation	Verification	Release	Response
Core security training	Establish security requirements	Establish design requirements	Use approved tools	Perform dynamic analysis	Create an incident response plan	Implement incident response plan
	Create quality gates and bug bars	Perform attack surface analysis and reduction	Desperate unsafe functions	Perform fuzz testing	Conduct final security review	
	Perform security and privacy rich assessments	Use threat modeling	Perform static analysis	Conduct attack surface review	Certify release and archive	

DDoS defense system

A distributed denial of service (DDoS) is an attack in which multiple compromised computer systems attack a target, such as a server, website, or other network resource, and cause a denial of service for users of the targeted resource.

Azure has a defense system against DDoS attacks on its platform services. It uses standard detection and mitigation techniques, and is designed to withstand attacks generated from outside and inside the platform.

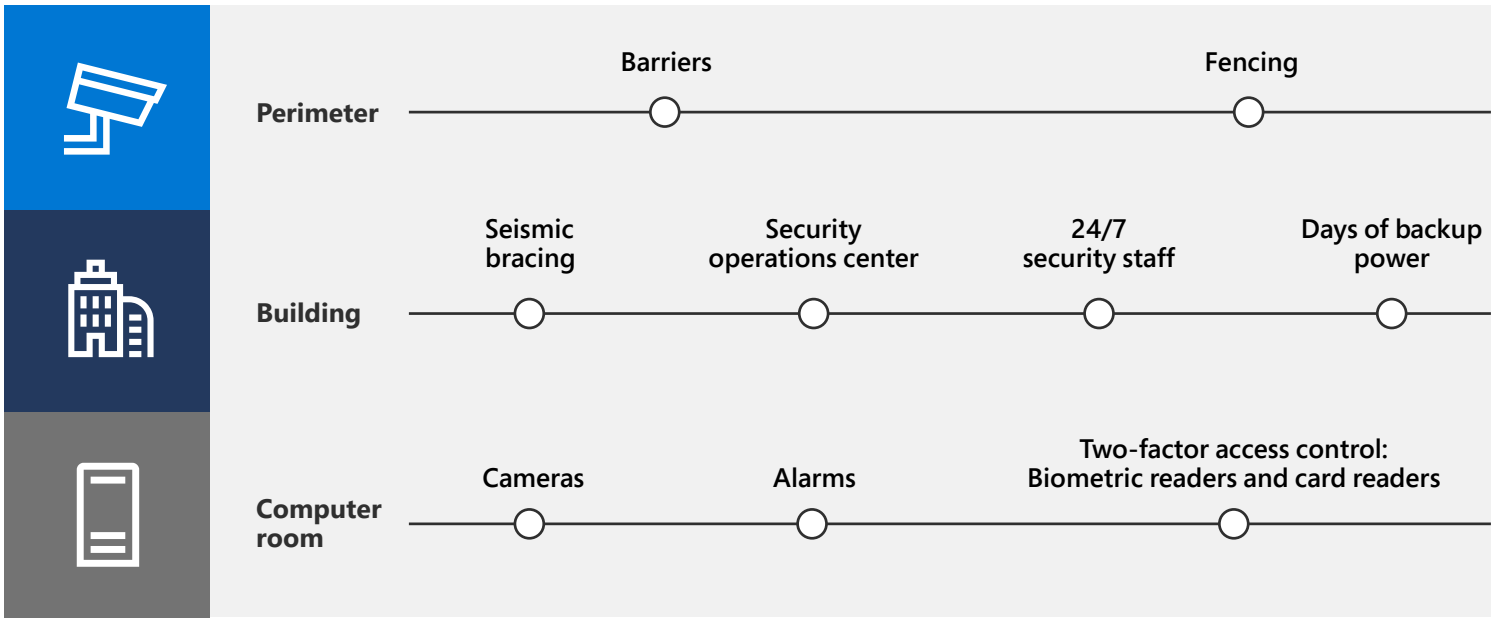
Data segregation

Dynamics 365 runs on Azure, so it's inherently a multi-tenant service, meaning that multiple customers' deployments and virtual machines are stored on the same physical hardware. Azure uses logical isolation to segregate each customer's data from others. This provides the scale and economic benefits of multi-tenant services while rigorously preventing customers from accessing one another's data.

Encryption

Data is an organization's most valuable and irreplaceable assets, and encryption serves as the last and strongest line of defense in a multi-layered data security strategy. Microsoft business cloud services

Fig. 12-8



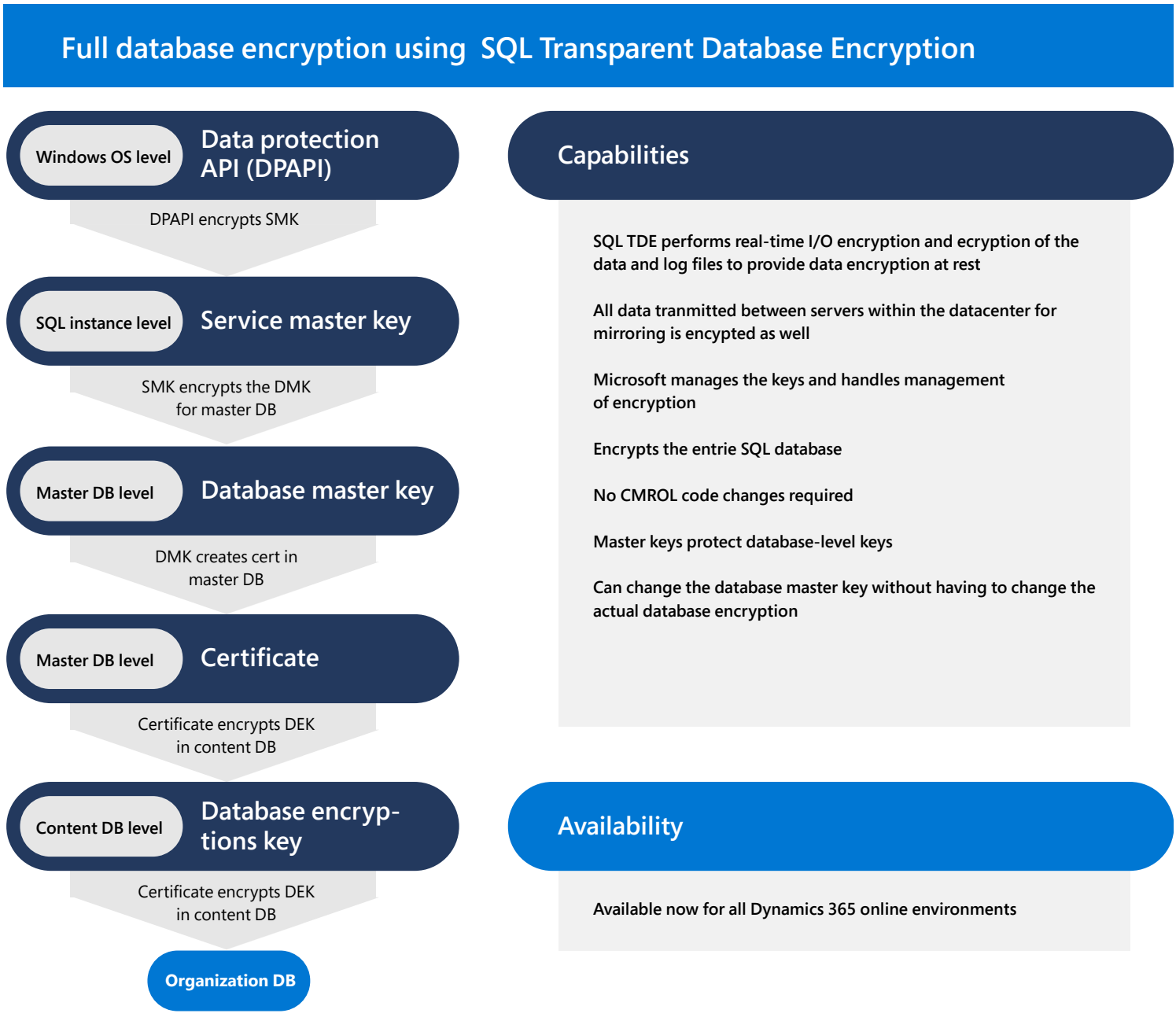
and products use encryption in transit and at rest to safeguard customer data and help maintain control over it.

At-rest data protection

At minimum, Dynamics 365 environment databases use SQL TDE (Transparent Data Encryption, compliant with FIPS 140-2) to provide real-time I/O encryption and decryption of the data and log files for data encryption at rest (**Figure 12-9**).

By default, Microsoft stores and manages the database encryption keys for your Dynamics 365 deployments. Finance and Supply Chain Management apps use server-side encryption using service-managed

Fig. 12-9



keys. All key management aspects such as key issuance, rotation, and backup are handled by Microsoft. For Customer Engagement apps, the optional customer managed encryption key capability provides administrators the ability to self-manage the database encryption key for all the instances in their tenant.

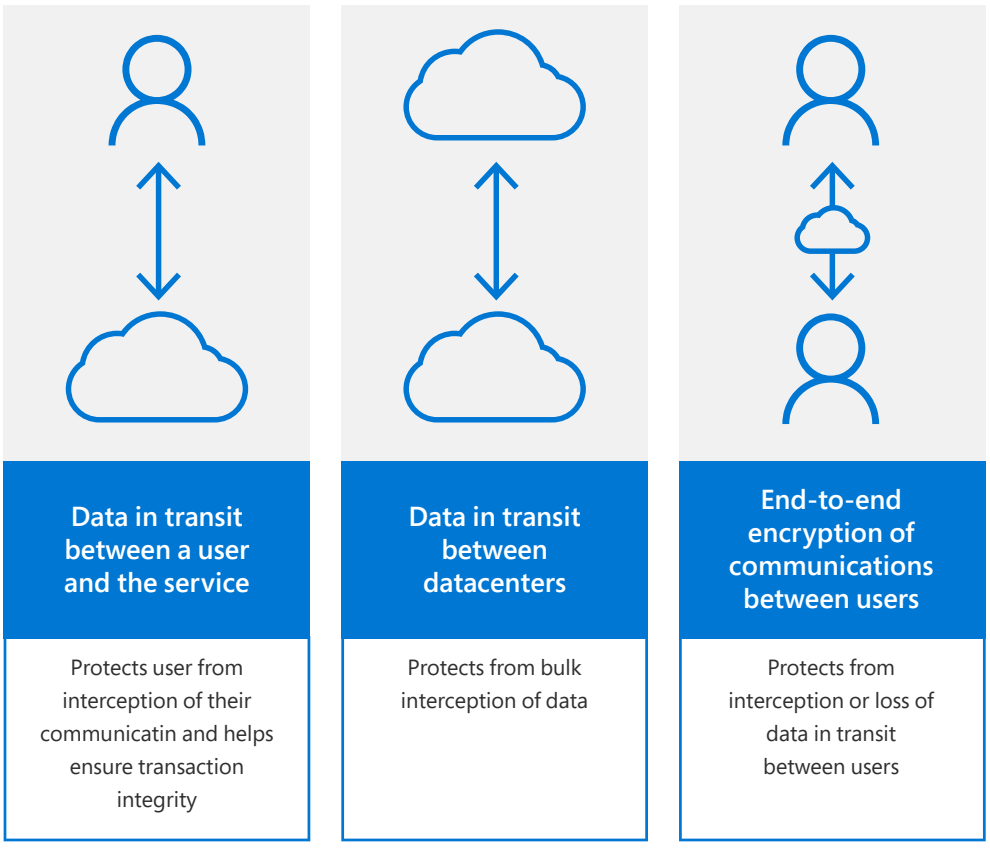
In-transit data protection

Your data in transit is protected in several ways (**Figure 12-10**). Dynamics 365 uses HTTPS encryption so that you can enable encryption for traffic between Dynamics 365 and end users. Azure protects data in transit to or from outside components, as well as data in transit internally, such as between two virtual networks. Azure uses industry standard transport protocols such as TLS between user devices and Microsoft datacenters, and within datacenters themselves.

Secure identity

Identity and access management is critical to every organization. Azure Active Directory (Azure AD) is a complete identity and access management solution with integrated security that connects 425 million people to their apps, devices, and data each month. Dynamics 365 applications are safeguarded using Azure AD as a seamless identity solution.

Fig. 12-10



Authentication: Users

Authentication is the process of proving an identity. The Microsoft identity platform uses the OpenID Connect protocol for handling authentication. By default, only authenticated users can access Dynamics 365.

Azure AD is used as a centralized identity provider in the cloud. To access the system, users must

be provisioned into a Dynamics 365 environment and should have a valid Azure AD account in an authorized tenant. Azure AD is built to work for apps in the cloud, on mobile, or on-premises, and delegating authentication and authorization to enable scenarios such as the following (**Figure 12-11**):

- Conditional access policies that require a user to be in a specific location
- Multifactor authentication (MFA)
- Single sign-on (SSO), in which a user can sign in once and then be automatically signed into all the web apps that share the same centralized directory

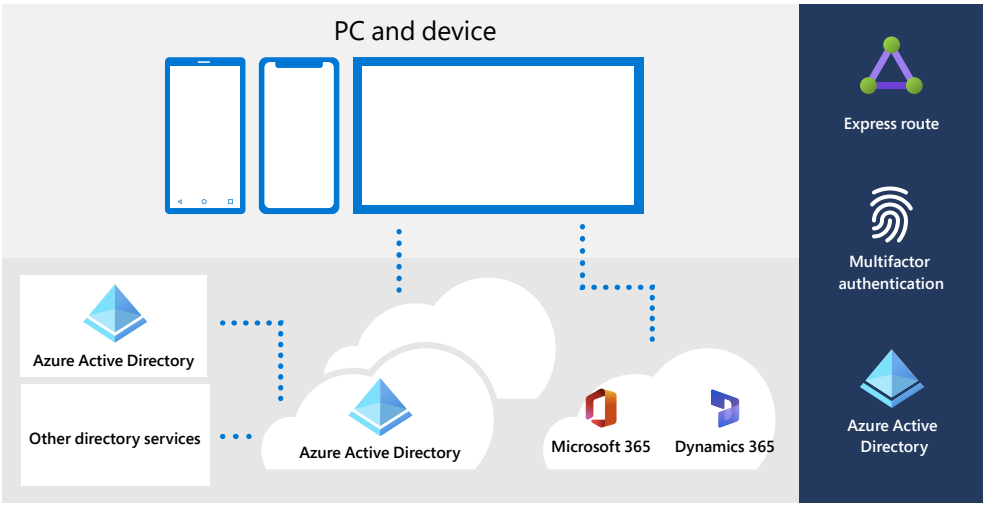
Azure AD provides fully secured identity federation with Active Directory on-premises. Federation with Azure AD or Microsoft 365 enables users to authenticate using on-premises credentials and access all resources in the cloud.

Authentication: Conditional access

Users can access cloud applications from anywhere and from any device such as mobile phones, tablets, or laptops from the office or home. As a result, there are many scenarios in which access control decisions shouldn't be made based solely on who can access a resource—you also need to consider how a resource is accessed. With Azure AD conditional access, you can address this requirement.

Fig. 12-11

Access control: Authentication for online services



With conditional access, you can implement automated access control decisions for accessing your cloud apps that are based on conditions (**Figure 12-12**).

Authorization

Authorization is the control of access to the Dynamics 365 applications. It allows you to control the functionality and data accessible to the users. Dynamics 365 uses security roles for

authorization. Once a user is authenticated, the security roles assigned to a user (or groups of users) authorizes the users for access to data, services, menus, or other Dynamics 365 features and capabilities. Note that the concept of security roles is different for Customer Engagement apps (based on Microsoft Dataverse and Power Apps) and Finance and Operations apps. We examine this difference later in this chapter.

Auditing and monitoring

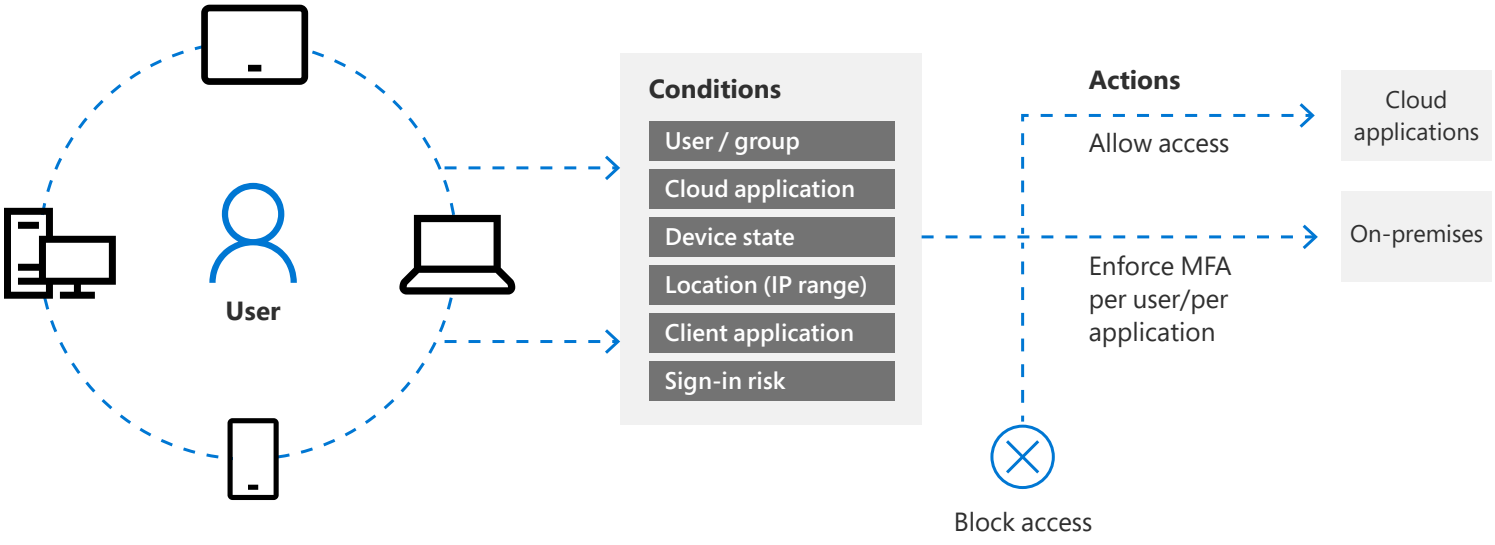
All Dynamic 365 applications provide audit functionality. Auditing provides a log of events related to activity for secure datastores, logins, or other important events. As with authorization, there are differences related to auditing in Customer Engagement apps and Finance and Supply Chain Management apps, which we examine later in the chapter.

Customer responsibility

As a customer, you're responsible for:

- Account and identity management
- Creating and configuring conditional access policies
- Creating and assigning security roles
- Enabling and configuring auditing and monitoring
- Authentication and security of components of the solutions other than Dynamics 365

Fig. 12-12



Security for Dynamics 365 integrations

Most Dynamic 365s implementations have integrations with a range of Azure services, on-premises systems, and cloud systems (**Figure 12-13**).

Consider the out-of-the-box integrations before building custom solutions. The native integration with the other Microsoft services is a tried, tested, and recommended solution for these scenarios. The following are examples of native integration options available in Dynamics 365:

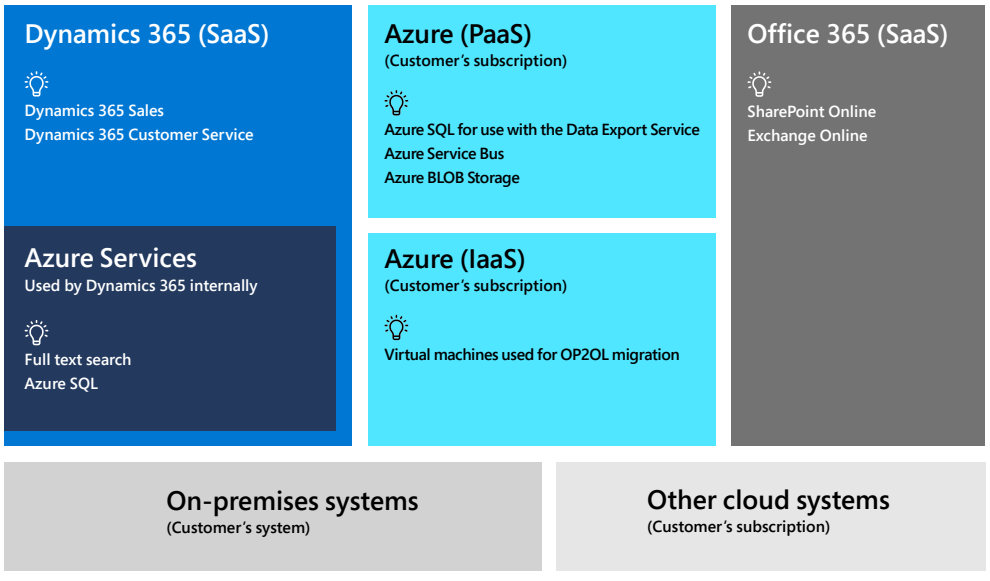
- Dynamics 365 and Microsoft SharePoint integration
- Dynamics 365 and Microsoft Exchange Online integration
- Dual-write for integration between Customer Engagement apps and Finance and Supply Chain Management apps.

Use server-to-server (S2S) authentication to securely and seamlessly integrate Dynamics 365 with your custom websites and services. Some of the native integrations we mentioned such as Dynamics 365 and SharePoint also use S2S authentication.

Regarding your on-premises systems, Microsoft recommends you deploy an on-premises data gateway connected to an Azure hybrid connection where possible.

Azure Service Bus is a fully managed enterprise message broker with message queues and publish-subscribe topics. Service Bus is used to decouple applications and services from each other. In many cases, we recommend integrating line-of business applications with cloud services by using Service Bus, which provides a secure channel for communicating between different on-premises or cloud-based line of business applications.

Fig. 12-13



Many customers implement an allowlist for IP ranges that are permitted for outbound connections. Modern cloud services such

as Dynamics 365 don't have static and specific IP ranges for Dynamics 365 solutions. We recommend you include the whole Azure region on the IP allowlist for the system to work properly. Note that the list of IPs to allow is considerably shorter for Azure Logic Apps. In addition, the use of the on-premises data gateway previously mentioned can reduce or eliminate this need.

Transparency

Microsoft is transparent about where your data is located. You know where your data is stored, who can access it, and under what conditions. Dynamics 365 customers can specify the Azure datacenter region where their customer data will be stored. Microsoft may replicate customer data to other regions available within the same geography for data durability, except in specific scenarios, such as the following:

- Azure AD, which may store AD data globally
- Azure multifactor authentication, which may store MFA data globally
- Customer data collected during the onboarding process by the Microsoft 365 admin center

Fig. 12-14

Transparency goals	Implementation details
Choose where our data is stored	You select the region where your Dynamics 365 data is stored
Transparent about how we respond to government requests for your data	No third-party (including law enforcement, other government entity, or civil litigant) is given direct or unfettered access to customer data except as you direct

Microsoft imposes carefully defined requirements on government and law enforcement requests for customer data. As described at the Microsoft Privacy Principles, if Microsoft receives a demand for a customer's data, we direct the requesting party to seek the data directly from the customer. If compelled to disclose or give access to any customer's data, Microsoft promptly notifies the customer and provides a copy of the demand unless legally prohibited from doing so.

Figure 12-14 lists some standard transparency goals and their implementation in Dynamics 365.

Customer Engagement apps security

The Customer Engagement apps (Dynamics 365 Sales, Dynamics 365 Customer Service, Dynamics 365 Field Service, Dynamics 365

Marketing, and Dynamics 365 Project Service Automation) use the Microsoft Dataverse to provide a rich security model that can adapt to many business scenarios.

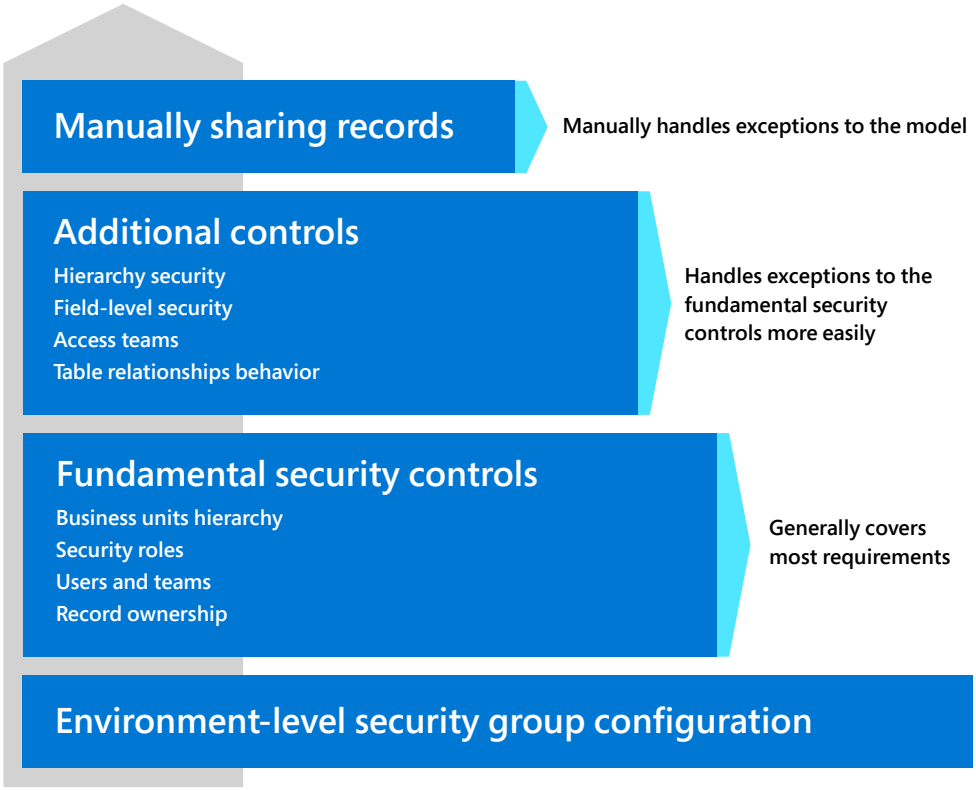
The goals of Dataverse security model are as follows:

- Provide users with access only to the appropriate levels of information that is required to do their jobs
- Categorize users by role and restrict access based on those roles
- Support data sharing so that users and teams can be granted access to records that they don't own for a specified collaborative effort
- Prevent a user's access to records the user doesn't own or share

Security features

We use three main categories of security features to provide appropriate end-user access (**Figure 12-15**): fundamental security controls, additional security controls, and manual sharing. Most of the security requirements should be addressed using fundamental security controls; other options should be used to manage the exceptions and edge scenarios.

Fig. 12-15



Record ownership

Dataverse supports two types of record ownership:

- **Organization owned**
  When a record is assigned to Organization, everyone in the environment can access the record
- **User or Team owned**
  If not assigned to the organization, a record is assigned to Business Unit, Child Business Unit, Team, or User

Some out of the box tables are exceptions to the above 2 types such as system user record is owned by a Business Unit.

Business units

Business units are a security modeling building block that helps in managing users and the data they can access. The name "business unit" can be misleading because the term doesn't necessarily have any direct relationship with an organization's operating business units. In Dataverse, business units provide a framework to define the organizational structure of users, teams, and records. Business units group users and teams by organizational hierarchy and can work in conjunction with security roles to grant or restrict access to data.

As a best practice, minimize the number of business units in your implementation based on access control requirements instead of mapping the organizational chart into business units.

The real power of business units comes from their hierarchical nature. Users can be given access to records just in their business unit, or their business unit and the business units under their unit. For example, the hierarchical nature of business units can allow you to limit access to records at the site, district, region, and corporate levels. Business units are useful to segment data into ownership containers for access control.

Security roles

A privilege is permission to perform an action in Dynamics 365. A security role is a set of privileges that defines a set of actions that can be performed by a user. Some privileges apply in general (such as the ability to use the export to a Microsoft Excel feature) and some to a specific table (such as the ability to read all accounts).

Customer Engagement apps come with several out-of-the-box security roles. If you need additional or different security roles, start by copying existing ones.

To access Dynamics 365, users must be assigned one or more predefined or custom security role, either assigned directly to their user or inherited from a team they're a member of.

For example, the salesperson role is assigned a set of privileges that are relevant to the performance of the tasks defined for that role.

Teams

Teams provide a straightforward way to share records, collaborate with other people across business units, and assign security roles. While a team belongs to one business unit, it can include users from other business units. You can associate a user with more than one team, which is a convenient way to grant rights to a user that crosses business units.

Multiple types of teams available are in Dataverse:

- **Owner teams** You can make an owner team the owner of a record, which is a useful way to link a record to a specific business unit. Owner teams can also share records and be assigned to security roles.
- **Azure AD security group teams and Azure AD office group teams** These are special types of owner teams. Their membership can't be managed within Dynamics 365, instead they're linked to an Azure AD security group or Microsoft 365 groups. Provided that the users are appropriately licensed and part of the environment security group (optional), users who are added to the Azure AD group are automatically enabled in the system and added to the team when they connect to the environment. Using Azure AD is particularly useful because the groups and permissions also extend to Azure AD-enabled apps outside of Dynamics 365.
- **Access teams** These teams are unique because they can't own records and can't have a security role association. However, just like owner teams, access teams can have records shared with them. When enabled at the table level, access teams can grant specific record-level permissions to the members of a record's access team. This is an alternative to manually sharing the record with a user or a team.

Use owner teams when:
- Team-level record ownership is required, rather than user
- You need to assign security roles to users through teams
- Reporting on team progress is required

Use access teams when:
- The team members need different permissions on individual records than the same permission on the record type

Field-level security

You can use field-level security to restrict access to high business impact fields to specific users or teams. For example, you can enable only certain users to read or update the credit score of a business customer.

Field-level security is available for the default fields on most out-of-the-box tables, custom fields, and custom fields on custom tables. Field-level security is managed by the security profiles. The scope of field-level security is organization-wide and applies to all data access requests.

Sharing

Record sharing lets a user give access to a table record to other users or team. The user must have a share privilege on the record to be able to share it. Sharing should be seen as a way for users to manually manage exceptions to the default security model.

Keep in mind the following recommended practices:

- Share only the necessary records with the smallest set of users or teams possible
- Grant the minimum access required for users to do their jobs
- Disable share rights in security roles where possible
- Extensive sharing can cause performance issues at scale
- Microsoft recommends that you don't automate sharing

Hierarchy security

You can use a hierarchy security model for accessing data from a user or position hierarchy perspective. With this additional security, you gain more granular access to records, for example by allowing managers to access the records owned by their reports for approval or to perform tasks on reports' behalf.

Summary of different access control mechanisms

Figure 12-16 summarizes the different access control mechanisms discussed in this section. It highlights the ideal scenarios, benefits, and limitations of each option.

Environment security group

You can associate an Azure AD security group with a Dataverse environment to control user access. Unless users have specific, highly privileged Microsoft 365 admin roles, a user in Azure AD can't access any information in Dynamics 365 even with a valid authentication and security role assigned unless they are also a member of the correct environment security group in Dynamics 365.

It's a best practice to associate a security group with an environment. This prevents all the Azure AD-eligible (appropriately licensed) users from appearing as active users in the Dynamics 365 environment.

Audit

Auditing helps you comply with internal policies, government regulations, and consumer demands for better control over confidential data. Organizations audit various aspects of their business systems to verify

that system and data access controls operate effectively and identify suspicious or non-compliant activity.

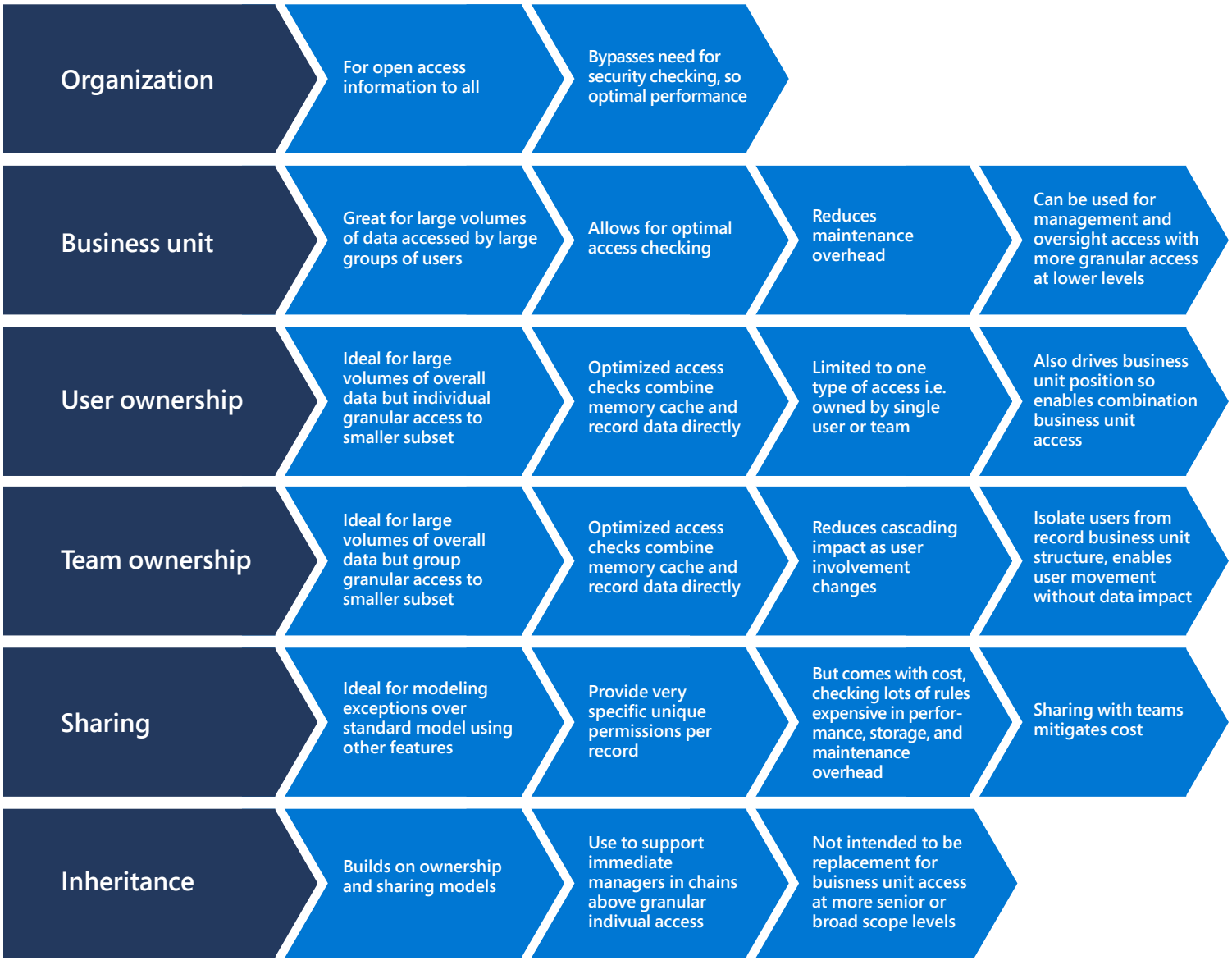
Dataverse audit

Audit logs are provided to ensure the data integrity of the system and to meet certain security and compliance requirements. The auditing feature logs the changes that are made to records and user access so the activity can be reviewed later.

Auditing can be enabled on:

- Table level (you can select specific fields in a table for auditing)
- User access

Fig. 12-16



Data auditing can be viewed:

- Contextually, at the record level, on the form
- Globally, from the audit summary view
- From the APIs

Don't enable auditing for all tables and columns. Do your due diligence to determine which tables and fields are required for auditing. Excessive auditing can affect performance and consume large volumes of log storage.

Microsoft 365 audit log

Activity logging records user and admin activities across Microsoft 365 and Dynamics 365 apps. Data is stored in a Microsoft 365 unified audit log. It has a smaller footprint on system resources compared to the previous Dataverse audit functionality. Microsoft 365 audit logs have several additional benefits:

- **Management in a central location on the Microsoft 365 security and compliance portals** Microsoft 365 admins can manage settings and access activity reporting for all environments within the Microsoft 365 security and compliance portals. Dataverse auditing is set up and stored separately within each Dynamics 365 environment.
- **All data in the system is logged** This functionality logs all data transactions, including read operations, plug-in operations, table operations, bulk operations, user sign in and sign out sessions, and even Microsoft Support operations.
- **Configurable alert policies** You can set up the system to notify administrators or compliance officers of certain events according to configurable settings.
- **Audit log search capability** Administrators can easily query audit logs via predefined or custom filters.
- **Analyze suspicious behavior with Security Information and Event Management (SIEM)** Use SIEM functions in near-real time to analyze and alert administrators of possible suspicious behavior within the system and provide actions to address these events.
- **SIEM vendor integration** Dynamics 365 now provides out-of-the-box integration with multiple SIEM vendors such as ArcLight, Microsoft OMS, Okta, SumoLogic, BetterCloud, and standard CEF

format integration for many others. Microsoft has also released a connector to integrate activity logs into Azure Sentinel.

By default, activity logging data is retained for 90 days or 1 year based on the Microsoft 365 license type. We recommend changing the retention policy or moving these logs to external stores if you need to retain the data for longer periods.

Finance and Supply Chain Management apps security

Finance and Supply Chain Management applications use Azure AD as a primary identity provider and utilize the security capabilities of Azure. To access these applications, users must be provisioned into a Finance and Supply Chain Management instance and should have a valid Azure AD account in an authorized tenant. Once authenticated, these applications use role-based security to authorize user access to individual elements of the applications. Additionally, you can use extensible data security policies to restrict access to a subset of data.

The security architecture of Finance and Supply Chain Management consists of the following components (**Figure 12-17**):

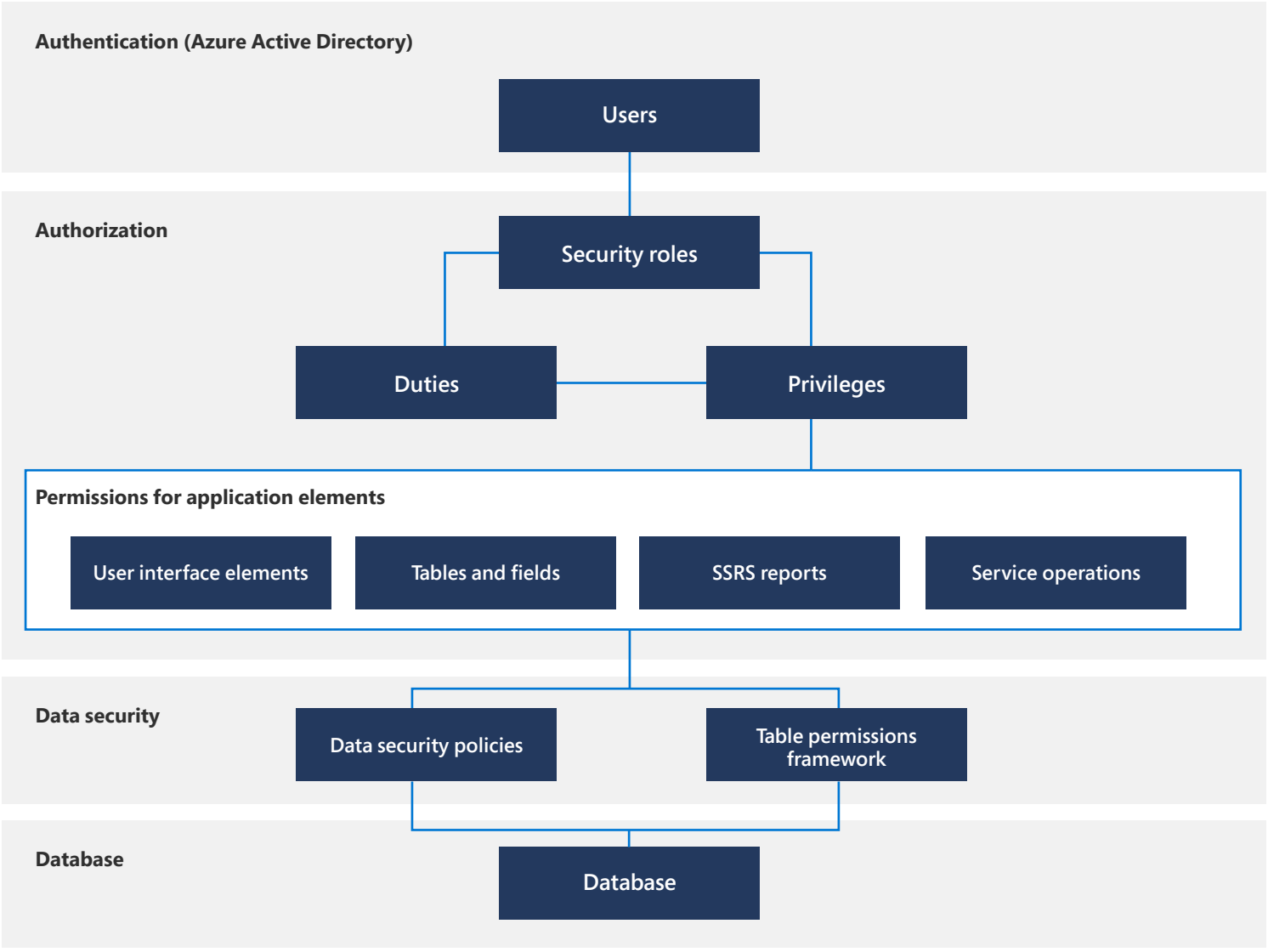
- **Authentication** Only authenticated users with user rights in Finance and Supply Chain Management apps can connect
- **Authorization** Security roles are comprised of duties, duties are comprised of privileges, and privileges are comprised of permissions
- **Security roles** Roles grant access to Finance and Supply Chain Management apps
- **Duties** Duties allow access to parts of business processes
- **Privileges** Privileges allow access to individual tasks
- **Permissions** Permissions grant access to individual securable objects
- **Data security** You use data security to deny access to tables, fields, and rows in the database (extensible data security framework)

- **Auditing** The system logs when a user signs in or out of the application

Role-based security

In Finance and Supply Chain Management apps, role-based security is aligned with the structure of the business. Users are assigned to security roles based on their responsibilities in the organization and their participation in business processes. Because you can set up rules for automatic role assignment, the administrator doesn't have to be involved every time a user's responsibilities change. After business managers set up security roles and rules, they can control day-to-day user access based on business data. A user who is assigned to a security role has access to the set of duties that are associated with that role, which is comprised of various granular privileges. A user who isn't

Fig. 12-17



assigned to any role has no privileges. Privileges are composed of permissions and represent access to tasks, such as cancelling payments and processing deposits. Duties are composed of privileges and represent parts of a business process, such as maintaining bank transactions. Both duties and privileges can be assigned to roles to grant access to Finance and Supply Chain Management.

We recommend duplicating the sample security roles and using these for role assignments for all users before you start creating custom roles.

By default, approximately 100 standard security roles are provided within the solution. All functionality is associated with at least one of the standard security roles. The administrator can duplicate and modify the sample security roles to fit the needs of the business or create entirely new security roles.

Security diagnostics for task recordings

The Task recorder for Finance and Supply Chain Management apps is a utility that lets users record business processes for several different use cases. This is an invaluable tool for many purposes, including process definition, training, testing, and security. The security diagnostics for the Task recorder allows implementers to analyze and manage security permission requirements in any task recording. When you use the Task recorder to define and document business processes for testing or training purposes, the security requirements for implementing those tasks are also captured. Implementers can then use this tool to tailor duplicated copies of the sample security roles to better meet their needs.

Segregation of duties

You can set up rules to separate tasks that must be performed by different users. For example, you might not want the same person to acknowledge the receipt of goods and to process payment to the vendor. This concept is named segregation of duties. Segregation of duties helps you reduce the risk of fraud, enforce internal control policies, and detect errors or irregularities.

The standard security roles included in Finance and Supply Chain Management apps incorporate segregation of duties. If a user needs full access, they often need a combination of roles (such as an HR assistant or HR manager for full access to HR features).

When an organization defines rules for segregation of duties, you should evaluate existing role definitions and role assignments for compliance, thereby preventing role definitions or role assignments that don't

comply. Future role assignments that result in conflicts with existing rules are logged and must be allowed or denied by an administrator.

Security reports

Finance and Supply Chain Management applications provide a set of rich security reports to help you understand the set of security roles running in your environment and the set of users assigned to each role. In addition to the security reports, developers can generate a workbook containing all user security privileges for all roles.

- **User role assignments report** Generates a view of the current user role assignments in your system. By default, the report includes all users with roles assigned. You can optionally limit the report to a specific set of users by entering a list of users when generating the report. In the User role assignments parameters pane, under Records, choose Filter. From here you can add or remove filters to the list of users the report will be generated for.
- **Role to user assignment report** Provides an aggregation of role assignments. Expanding a role in the report shows the list of users assigned to the role, and expanding the username shows any restrictions the role has applied. You can apply the same method for filtering the set of users to this report as described for the User role assignments report.
- **Security role access report** Provides a view of the effective permissions for each security role. This report provides a flattened list of permissions grouped by type across all sub-roles, duties, and privileges contained in the role.
- **Security duty assignments report** Provides a view of all the duties contained within a role. You can configure this report to run on any collection of roles to ensure that segregation of duties is maintained between roles. By default, the report includes all the roles. To limit the roles included, use the filtering provided in the Records to include section.

Familiarize yourself with these reports during implementation and engage your compliance team early. There is no benefit to waiting for an audit to use these reports for the first time.

To aid compliance with GDPR requirements, you can use the person search report when a data subject (the individual EU citizens or residents that share personal data) approaches the controller (the

organization collecting and managing personal data) to request a copy of their personal data.

System log

System administrators can use the User log page to keep an audit log of users who have logged on to the system. Knowing who has logged in can help protect your organization's data. The user logging capability allows administrators to identify roles that provide access to sensitive data. The sensitive data identifier enhances the user logging experience by letting your organization produce audit logs that show who in your system has access to sensitive data. This capability is helpful for organizations that might have multiple roles that have varying degrees of access to certain data. It can also be helpful for organizations that want a detailed level of auditing to track users who have had access to data that has been identified as sensitive data.

Extensible data security policies

Extensible Data Security (XDS) policies in Finance and Supply Chain Management apps allow developers to supplement role-based security by restricting access to table records based on security policies. Policies can be created that contain a query to select information and a table or tables that will be constrained by the policy. The query in the policy applies a filter and only records that satisfy the conditions of the filter are accessible from the restricted tables. XDS provides a powerful data security tool for advanced requirements, but you should use it judiciously to avoid unanticipated impacts on performance or usability.

Database logging

Database logging provides a way to track specific types of changes to the tables and fields in Finance and Supply Chain Management apps, including insert, update, delete, and rename key operations. When you configure logging for a table or field, a record of every change to that table or field is stored in the database log table (sysdatabaselog) in the environment database.

You can use database logging to create an auditable record of changes to specific tables that contain sensitive information, or to monitor the use of electronic signatures. Note that this feature is in addition to

audit trail capabilities already available in many modules. Database logging is intended to track individual changes to specific, sensitive information or configurations. It's not intended to track automated transactions that are run in batch jobs, and is not recommended for transactions. Although database logging can be valuable from a business perspective, it can be expensive with regard to resource use and management, so it too should be used judiciously.

Microsoft recommends the following when implementing database logging:

- Create a plan for reviewing and using your database logs
- Limit log entries and help improve performance by selecting specific fields to log instead of whole tables
- Create a plan for how long you will retain logged data and use the included cleanup job

Make security a day one priority

Security should be at the forefront when you start a Dynamics 365 project, not an afterthought. Negligence of security requirements can lead to significant legal, financial, and business risks. It can also impact the overall scalability and performance of the solution. In this section, we discuss some of the security impacts on Dynamics 365.

Impact on scalability

Security design can have a significant impact on scalability. Security design should consider scalability especially for multi-phased and global deployments.

Customer Engagement examples

In multi-phased global deployments, short-sighted security design can affect the rollout to other regions. The different regions can have their own compliance, business processes, and security requirements. The same model may not work for all the regions. Therefore, you should think about the security model holistically at the beginning of the project.

One such example can be a global deployment where in the first region (Phase 1) you create all users in the root business unit with business unit-level access in security roles. Because Phase 1 users were placed in the root business unit, when you roll out the solution to the next region (Phase 2), Phase 2 users have access to the users in the first region. This may cause some regulatory or compliance issues.

Several businesses use Organization owned entities for reference data tables, which means security roles don't apply to these entities. In multi-phased deployments, some departments may not want their reference data to be seen by other departments, or simply only want to see their reference data appearing when creating records. This segmentation is hard to achieve using Organization owned entities and much simpler to achieve when User or Team owned entities have Business Unit access. Entity ownership is a choice that happens at the time the entity is created and can't be changed. This assignment controls who can access the entity. If you want to change who has access to these entities, the only option you have is to delete these entities and recreate them in an owner (Organization, Team, Business Unit, and so on) that that is properly scoped.

Finance and Supply Chain Management examples

Similarly, for Finance and Supply Chain Management implementations, different legal entities may have distinct functions and purposes, be subject to different security constraints, and have completely different security requirements. Even if legal entities, subsidiaries, or regions have a high degree of standardization and centralization and are good candidates for streamlined security implementation, failing to source requirements from the right stakeholders, design security as modular building blocks, or otherwise fail to take a long-term view of security needs can result in subsequent rollouts struggling with ill-fitting global templates or a need to start over from scratch.

Design of role-based access control should follow best practices, or subsequent rollouts will be unable to use and further build upon established security roles. Security roles should be designed based on business processes and incorporate segregation of duties, not individual personas. We recommend that you avoid creating a single role for

each user, and avoid creating nested security roles. The resulting modular security roles can be assigned to users as they cumulatively give the access needed for a users' duties, and no more. Native tools such as the Task recorder and Regression suite automation tool support the testing of individual user permissions, which facilitate this approach. We also recommend following recommended practices for the initial rollout (such as copying standard roles rather than directly assigning them). This gives subsequent rollouts the flexibility of building upon existing role structures, while retaining access to standard roles for comparison. This prevents the subsequent rollouts from being forced to choose between modifying roles already in use elsewhere or starting from scratch because existing roles are too specialized.

Impact on performance

Security design can impact system performance. Dynamics 365 has multiple access control mechanisms, all with their own pros and cons. The ideal security design is a combination of one or more of these controls based on your requirements and the granularity, simplicity, and scalability they provide. The wrong choice can lead to poor performance.

Customer Engagement examples

Sharing provides particular and unique permissions on a specific record, but sharing for large volumes of records comes at a cost. To check access for a user, the system checks user security roles, team memberships, and records shared with a user and their teams. Checking lots of rules is expensive in terms of performance, storage, and maintenance overhead.

Sharing can also trigger cascading rules that can be expensive if not set appropriately. Sharing should be for edge cases and exception scenarios.

You can get more information on this topic from the Modelling Scalable Security Modelling with Microsoft Dynamics CRM whitepaper. It's an older document, but still relevant for Dynamics 365 security.

The volume of data will impact performance in the long run. You should have a well-defined archival strategy. The archived data can be made available through some other channels.

Finance and Supply Chain Management examples

Database logging in Finance and Operations can be valuable from a business perspective, but it can be expensive regarding resource use and management. Here are some of the performance implications of database logging:

- The database log table can grow quickly and increase the size of the database. The amount of growth depends on the amount of logged data that you retain.
- When logging is turned on for a transaction type, each instance of that transaction type causes multiple records to be written for the Microsoft SQL Server transaction log file. Specifically, one record is written for the initial transaction, and one record logs the transaction in the database log table. Therefore, the transaction log file grows more quickly and might require additional maintenance.
- Database logging can adversely affect long-running automated processes, such as inventory close, calculations for bills of materials, master planning, and long-running data imports.
- When logging is turned on for a table, all set-based database operations are downgraded to row-based operations. For example, if you're logging inserts for a table, each insert a row-based insert.

When you're implementing database logging, be sure to have a plan for how long you'll retain logged data, and how you'll limit log entries and help improve performance by selecting specific fields to log instead of whole tables.

Extensible Data Security policies allow developers to supplement role-based security by restricting access to table records based on security policies. The query in the policy (the policy query) applies a filter so only records that satisfy the conditions of the filter are accessible from the restricted tables. The policy query is added to the WHERE clause, or ON clause, on SELECT, UPDATE, DELETE and INSERT operations involving the specified constrained tables.

Unless carefully designed and tested, policy queries can have a significant performance impact. Therefore, make sure to follow the simple but important guidelines when developing an XDS policy. Performance can be significantly impacted by poorly performing queries. As a best practice,

avoid numerous joins, consider adding lookup tables to improve performance, carefully tune queries for optimal performance, and carefully and thoroughly test any XDS policies when using this feature.

Impact on regulation and compliance

Several different security controls are required to meet regulations and compliance based on industry and region. These controls are a combination of duty separation, data categorization, identity management, access control, log management, and auditing. Missing any of these controls can make your solution noncompliant.

Governance takes time to understand, roll out, and adopt. The earlier you begin the process, the easier compliance becomes.

Customer Engagement examples

The activity logging data is retained between 90 days to 1 year based on the Microsoft 365 license type. For compliance reasons, if you need to keep these records for longer, you should move this data to an external store.

Impact on rollout

In many projects, security design is created late in the implementation process. It can lead to a lot of issues, ranging from system design to inadequate testing. Keep the following recommended practices in mind:

- Test the system using the right security roles from the very beginning.
- Use the correct security profile during test cycles. Don't provide an administrator role to everyone.
- When training people, make sure they have access to make all the required actions. They shouldn't have access to more information than required to complete those actions.
- Validate security before and after data migration occurs.

Finance and Supply Chain Management examples

Design your security roles with both compliance and scalability in mind. By engaging your security and compliance team early in your implementation, you can use the segregation of duties framework to

identify conflicts and create rules that enforce separation of duties that should be separated.

For example, a global company has made efforts to standardize business processes and is planning to roll out a new functionality in multiple phases to globally. Rules that segregate duties in your large legal entities prevents assignments that may result in conflicts.

However, in smaller legal entities, users may need to accomplish the same tasks with a smaller staff, which makes it challenging to construct declarative rules that enforce segregation of duties. In this case, modular security roles allow you to enforce the same segregation of duties rules but when assigning multiple roles to some users, and apply documented overrides to the detected conflicts. In this manner, your global rollout can reuse modular security components, and you have a robust segregation of duties framework in place for entities of many sizes, allowing them to grow without introducing the need to redesign the security implementation.

Impact on reporting

Security design can have a significant impact on reporting. For example, your security design identifies sensitive data that requires restricted access to be enforced and monitored. You need to create auditing reports related to access and changes on sensitive data, and designated personnel responsible for oversight need to review them periodically.

In addition to built-in reporting capabilities, you have multiple options for exporting Dynamics 365 data into external stores for reporting and analytics. The two native options are exporting to a data lake or using the Data Export Service (not available in Finance and Supply Chain Management). When data is in external stores, it doesn't honor Dynamics 365 security permissions (nor would it be useful to do so in many cases). You need to apply your own security mechanism for reporting. We recommend having a data management strategy for all data that is stored outside of a core application. This strategy should include security requirements (encryption, access controls, row level, personally identifiable information) as well as data residency and retention requirements. The strategy should consider the future state

of Dynamics 365 data in the analytical platform as well as its legacy and other applications.

You can use Tabular Data Stream (TDS) endpoints with Customer Engagement for direct query scenarios, which honor the Dynamics 365 security context.

Operational aspect of security

Operation aspects can have a big impact on the security design. Operation aspects can decide which access mechanisms are the best fit for the solution, such as in cases of organizational changes or maintenance.

Organizational changes

What happens if a user moves to another team or business unit? Do you need to need to reassign the records owned by the user? What should be the process?

Do you reassign all records or just active record? What is the effect of changing the ownership on the child records, especially closed activities, closed cases, and won or lost opportunities? Based on these questions, you can come up with a process and cascading rules in Dynamics 365.

Reassigning a large volume of records can also take a long time and affect the system's performance. Learn more about assigning or sharing rows.

Maintenance overhead

Plan out how to assign security roles to new users. Are you going to automate a process? For example, you can use Azure AD group teams in Dynamics 365 to assign roles to new team members. This can make it very easy to assign license and security roles to new team members.

Security anti-patterns

An anti-pattern is a frequently implemented, ineffective response to a problem. Several security anti-patterns should be avoided for scaling, performance, and security reasons. We shared a few in the previous

section, such as using Organization owned entities for reference data tables in Customer Engagement, or indiscriminately logging transactional tables in Finance and Operations apps. In this section, we discuss a few more anti-patterns to avoid.

Security through obscurity

Project teams may try to implement security through obscurity. One example is hiding the fields from forms and views through custom code. This is not security, because anyone with a proper security role can access those fields in several ways such as with the use of Advance Find, Power Automate, and APIs.

Always consider all the ramifications of avoiding default security controls.

Credentials stored in Dynamics 365

Azure Key Vault allows you to securely store and manage application credentials such as secrets, keys, and certificates in a central and secure cloud repository. Key Vault eliminates the need to store credentials in your applications. Your applications can authenticate to Key Vault at run time to retrieve credentials.

This is also one of the more common anti-patterns. Projects often use calls to external systems through plugins and JavaScript. Calling external services through plugins is recommended, but the credentials to call these services are stored in the code of some configuration entities in the system. This practice creates a significant security risk because the credentials could be discovered by numerous roles not directly related to the use of the service. In addition, if the credentials change, the call to the external service fails. As a best practice, don't store any credentials in the system.

Conclusion

In this chapter, we introduced how the Trusted Cloud is built on the foundational principles of security, privacy, compliance, and transparency, and outlined basic concepts of information security as they apply to Dynamics 365. We then took a closer, product-specific look at how these security concepts are applied to Customer Engagement, Finance, and Supply Chain Management applications. With that as a foundation, we then examined why it's crucial to the success of your implementation to make security a priority from day one, citing some

Resources

General

Microsoft Dynamics CRM Online security and compliance planning guide

Trusting the Cloud

Compliance

An Introduction to Cloud Computing for Legal and Compliance Professionals

Achieving Trust and Compliance in the Cloud

Privacy

Privacy at Microsoft

Protecting Data and Privacy in the Cloud

Security

Dynamics 365 Security Assessment

Security Incident Management in Microsoft Dynamics 365

Encryption in the Microsoft Cloud

Security in Microsoft Dataverse

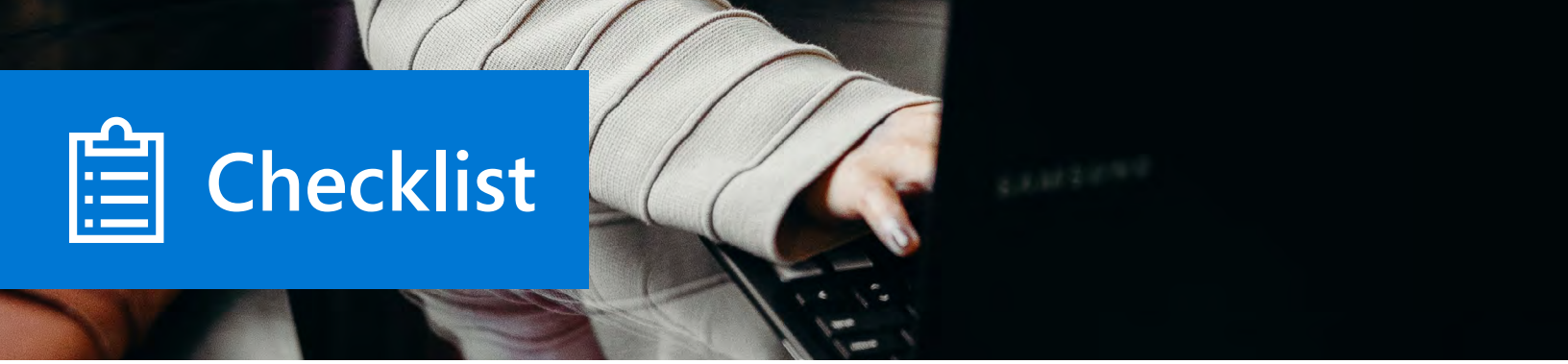
Security architecture

Plan and implement security in Finance and Operations apps

Review the security model for your Dynamics 365 solutions

specific examples and considering the impact of security on performance. Finally, we wrapped up with a handful of anti-patterns to avoid.

Equipped with this information, you can be confident in the security considerations and capabilities of Dynamics 365 products and better ensure the security of your implementation.



Checklist

Privacy and compliance

- [ ] Understand the responsibilities of the service provider as a data processer versus the customer responsibilities as the owner and data controller to ensure compliance on both sides.
- [ ] Refer to the Dynamics 365 cloud service agreements and compliance documentation to understand the policies, procedures for handling data, disaster recovery strategy, data residency, encryption, and more.

Identity and access

- [ ] Establish an identity management strategy covering user access, service accounts, application users, along with federation requirements for SSO and conditional access policies.
- [ ] Establish the administrative access policies targeting different admin roles on the platform, such as service admin and global admin.
- [ ] Enforce relevant DLP policies and procedures to make changes or request exceptions.
- [ ] Have necessary controls to manage access to specific environments.

Application security

- [ ] Understand the app-specific security constructs and model your security based on the native access control mechanisms rather than customizing the build.
- [ ] Understand that hiding information from the view doesn't necessarily remove access—there are alternate mechanisms to access and extract information.
- [ ] Understand the impact of losing the security context when the data is exported.
- [ ] Ensure the security model is optimized to perform and provides the foundation for further expansion and scale by following the security model best practices.
- [ ] Have a process to map changes in the organization structure to the security model in Dynamics 365. This needs to be done cautiously in a serial way due to the downstream cascading effect.

13

Guide

Business intelligence, reporting, and analytics

Introduction

In a world where so much comes down to reporting and analytics, business intelligence has become the differentiator—it's all about actions and outcomes, measuring what matters the most, and empowering users with the right insights at the right time.

Business intelligence solutions have evolved to keep up with an ever-increasing number of data sources, as organizations seek ways to expand their global reach to attract more customers, and give their current customer base access to more content, purchasing channels, and brand options. In this context, turning collected data into competitive business knowledge is more important than ever.

While AI and machine learning keep evolving—and are now used natively by apps—your organization's intelligence strategy should be based on a modern data estate, with a focus on how to strengthen the organization.

In this chapter, we discuss how data helps businesses make informed decisions. We also cover the evolution of business intelligence solutions, and how organizations can build a data estate based on modern components, on top of solutions that support augmented analytics and embedded intelligence.

Business intelligence solutions strengthen organizations, prepare them for challenges and market changes, and guide them toward the best insights and actions.

The importance of data

As James Phillips, president of Microsoft's Digital Transformation Platform

Group, wrote in the Microsoft Dynamics 365 blog, "There is a fundamental change occurring across industries and organizations: Data now comes from everywhere and everything." As an essential part of a business solution, data represents a sizable portion of each user's engagement. The solution processes and analyzes the data to produce information, which allows an organization to make informed decisions, and determines actions that can come from it.

As a part of the day-to-day usage of any application, users generate significant amounts of process-related and customer-related data—structured and unstructured. For example, customers generate data by visiting a website or using a specific product. In turn, businesses generate data about their products or services, as well as how customers interact with those offerings.

In addition, the pace of data growth is increasing dramatically. Not only does user activity create data, but new data sources such as the Internet of Things (IoT) need to be captured, rationalized, and secured.

The key role of data is to generate information that helps businesses and customers make informed decisions. For businesses seeking to stay competitive, the data also must trigger actions demonstrating that they understand and value their customers.

Resilience—the ability to adapt quickly to changing conditions—has become a hallmark for business success today. Competitive, resilient organizations invest in turning the data they collect into business knowledge, build the right data estate to generate a 360-degree view of their users, and use business intelligence to determine the best actions and outcomes.

Organizations that have already embarked on their transformation journey to improve the ways they use data are better prepared to respond and adapt to change.

Break the silos and use the digital feedback loop

Organizations are challenged by how to deal with ever-increasing amounts of data—and how to generate user value with it—especially when that data becomes siloed by the systems that generate or collect

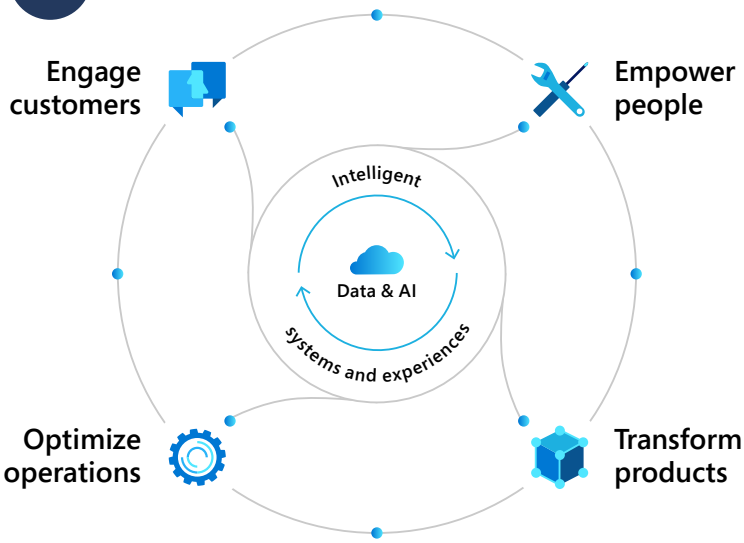
it. Data silos make the goal of having a 360-degree view of each user even more challenging. Successful organizations are able to digitally connect every facet of their business. Data from one system can be used to optimize the outcomes or processes within another. By establishing a digital feedback loop (**Figure 13-1**) that puts data, AI, and intelligent systems and experiences at the core, organizations can transform, become resilient, and unlock new values for users.

Evolution of business intelligence, reporting, and analytics

Practices for gathering, analyzing, and acting on data have evolved significantly over time. Traditional methods of standardizing and generating static reports no longer give businesses the agility to adapt to change. New technology and secure, highly reliable cloud services—which are poised to meet the needs of organizations that must quickly manage increasing amounts of data—have given rise to a new era of digital intelligence reporting.

Traditional reporting

Fig. 13-1



The early generations of business intelligence solutions were typically centralized in IT departments. Most organizations had multiple data repositories in different formats and locations that were later combined into a single repository using extract, transform, and load (ETL) tools, or they would generate reports within siloed sources and merge them to provide a holistic view of the business.

Once the data was unified, the next steps were deduplication and standardization, so the data could be structured and prepared for reporting. Business users who lacked the expertise to perform these tasks

would have to rely on the IT team or specialized vendors. Eventually, the business would receive static intelligence reports, but the entire process could take days or even weeks, depending on the complexity of the data and maturity of the processes in place. Data would then undergo further manipulation, when required, and be shared across different channels, which could result in the creation of multiple versions that would be difficult to track.

Such an approach, which also heavily depended on the business owner's analysis to generate business insights and actions, was often a time-consuming and error-prone process. And if an error was found, it had to be fixed in the central repository. The process to produce the report would need to be executed again—and repeated multiple times if the same fix wasn't applied at the source. This method of reporting was referred to as "closing activities," because it typically occurred at the end of a month, quarter, or year, which meant that organizations were slow to respond to opportunities because they waited until the end of a given period to analyze their data and make decisions.

Self-service reporting

The evolution to a more agile approach favored self-service capabilities to empower users. More user-friendly solutions reduced the IT dependency and focused on providing business users with access to data and visualization tools so they could do their own reporting and analytics. This accelerated the speed of data analysis and helped companies make data-driven decisions in competitive environments. However, in this model, reporting was unmanaged—increasing the number of versions of reports, as well as different representations of the data—which sometimes prevented organizations from using a standardized method to analyze data and inhibited effective decision-making.

This updated approach didn't entirely replace the IT-centric model, as many organizations started using a combination of traditional and self-service reporting, but it did provide quicker access to information, so organizations could react faster to business challenges and opportunities.

Reporting from anywhere

The growth of IT infrastructures, networks, and business usage of devices such as mobile phones and tablets launched a digital transformation from legacy systems into more modern solutions for most organizations. Business intelligence apps allowed reporting from anywhere at any time, giving users access to data while they were out of the office or on business trips. These apps improved how organizations could respond to customers, and gave a 360-degree view of each customer interaction. They also provided more succinct visualizations of data, with better features to interpret it.

The new era of business intelligence solutions

With data now coming from everywhere and everything, organizations must be prepared to convert that data into business knowledge so users can make informed decisions and trigger actions. Many organizations employ highly skilled workers, such as data scientists, who are responsible for analyzing the data and producing the necessary insights that can affect the business. This approach is expensive and adds dependency on specialized roles to perform tasks that usually can't be completed by typical business users.

An approach to reduce this dependency while simultaneously increasing ready access to data is to use augmented analytics, a feature of modern business intelligence solutions. According to Gartner, Inc., a research and advisory firm, "Augmented analytics is the use of enabling technologies such as machine learning and AI to assist with data preparation, insight generation, and insight explanation."

Embedded intelligence solutions that make use of augmented analytics allow processes, services, or products to deliver their own insights to increase quality, efficiency, and customer satisfaction. These are new types of reporting and analytic experiences where the intelligence is by-design, and the solution itself analyzes the data to provide insights.

While the data may not yet be unified with such an approach, customers and organizations can use insights provided by products and services from the earliest phases of usage or production, which increases their return on investment (ROI).

Reporting and analytics strategy

To be a market leader, an organization's products and services must evolve continuously and exceed customer expectations. The information used to improve a product or service is based on data that comes to the surface via reporting. Reporting requirements can be as simple as determining the status of an order or as complex as a cash-flow analysis for a global organization.

What is a reporting solution?

An effective reporting solution is one of the essential elements for business success. A properly implemented reporting solution provides a competitive advantage, as users have access to the exact information they need exactly when they need it, which accelerates and improves the decision-making process.

Define your analytics strategy

An analytics strategy determines how an organization will provide the ideal level of accurate information to users for timely decision-making. The strategy includes a list of tools and types of reporting (such as analytics, static reports, and ad-hoc reporting), as well as how the information will be secured and shared with users. Implementation of the strategy forms a foundation for defining standards, processes, and best practices.

Your analytics strategy can help transform data collected from different processes and systems into knowledge to help you stay competitive in the market.

A successful strategy caters to user requirements and how those requirements must be fulfilled. For example, some business users may require printed reports, which can be sent to customers and vendors. Others may need data to be available for summarized, detailed, or ad-hoc reporting. Business management may require financial or operational reports to understand the health of the organization and determine an overall strategy. Serving such varied user needs often requires different tools to manage data coming from multiple sources. For example, financial and ad-hoc reporting requirements may rely on data extracted and stored in a staging location utilizing Microsoft Azure data services such as Azure SQL Data Warehouse, as well as manufacturing or sales information in Dynamics 365 apps.

With a centralized data and business intelligence strategy, the IT department is usually responsible for data, ETL processes, and reporting solutions. In a self-service approach, the IT team implements a data analytics platform that enables users without statistical, analytical, or data-handling expertise to access and use data. Teams should understand their organization's goals and strategy before implementing an enterprise data platform.

Any reporting solution requires ongoing maintenance and updates as underlying system requirements and business priorities change. The project team should clearly define the delivery, maintenance, and operational responsibilities. There should also be a plan for making historical data from legacy systems available for reporting, business intelligence, or audit purposes.

Every organization needs people who understand how to analyze and use data. Training and communication are key to ensuring that business users know how to use data and insights to optimize their work, drive efficiency, and make informed decisions.

Data and analytics processes change periodically to improve how information is provided. Make sure users are aware of any upcoming change and understand how it will affect them. Constant updates without appropriate change management can confuse users, but effective change management can improve the adoption rate of analytics.

Gain deeper insights from data

For most organizations, business intelligence and analytics are critical to their growth and digital transformation. With technology such as Azure Machine Learning and Adaptive Insights, organizations can gain deeper insights from their data. Having an early understanding of these technologies also can help organizations design their systems for future-state data consumption.

Public, cloud-based business intelligence solutions are a good fit for many organizations, as they are on-demand, self-service, and

scalable solutions. To choose the best technology to meet requirements today and in the future, the implementation team must understand their organization's current state and the future vision for enterprise business intelligence and analytics.

Understand reporting requirements

Understanding reporting requirements is key to delivering successful reports and helping improve the entire business. Reporting requirements must be examined in the context of the business process for the purpose of the report—such as operational reporting, financial or regulatory reporting, ad-hoc inquiries, and dashboard and analytical reporting—and should not be an afterthought.

As part of the business requirements gathering process, organizations must address reporting requirements, and collect information about how reports will be used, such as:

- Printed on paper and sent with a shipment on a truck.
- Used to create a pivot table in Microsoft Excel and build another report.
- Submitted to government agencies at the end of the year.
- Provided to a bank on a periodic basis for audit purposes.

Organizations must also identify any critical reports that require data mash-up with other sources, such as front-office apps or transportation systems, to develop an appropriate data integration and solution strategy. Data-volume reporting requirements help determine how reports will be designed and shared with users.

If reports will be validated by a third party, organizations need to determine how reports will be generated and shared, and what data must be included in the report before it can be approved.

Requirements gathering also includes security policies—encryption, access controls, and row and column level—and requirements related to data retention, data residency, and personally identifiable information (PII).

Document printing and electronic reporting

Document printing requirements are another factor to consider. To

comply with local laws, businesses typically must submit regulatory and compliance documents in a pre-defined printed or electronic format provided by government agencies. The data required for these documents often resides in enterprise resource planning (ERP) systems.

The Dynamics 365 Finance and Dynamics 365 Supply Chain Management apps have an Electronic reporting (ER) tool to help organizations adopt new regulatory requirements and generate documents in the required format to electronically exchange information with government bodies, banks, and other parties. Targeted at business users instead of developers, the ER tool lets users configure different document formats instead of writing code, making the processes for creating and adjusting formats for electronic documents faster and easier. It works with the TEXT, XML, Microsoft Word document, and OPENXML worksheet formats, and an extension interface provides support for additional formats.

Customer-engagement apps allow users to use Word templates to create standardized documents. The Dynamics 365 Customer Engagement app includes system reports for sales insights, while Dynamics 365 Sales lets you individually print quotes, invoices, or other records—and create PDF files from sales records.

Business document management

Business document management is built on top of the ER framework, and enables business users to edit document templates by using Microsoft 365 or a Microsoft Office desktop application. Edits might include changing business document designs and adding placeholders for additional data without source code changes and new deployments. No knowledge of the ER framework is required to update templates of business documents.

Financial reporting

Finance and business professionals use financial reporting to create, maintain, deploy, and view financial statements. The Finance app's financial reporting capabilities move beyond traditional reporting constraints to help you efficiently design distinct types of reports.

It includes complex currency-reporting requirements and financial dimension support, and makes account segments or dimensions immediately available—no additional tools or configuration steps are required.

Financial reporting in the Finance app provides default financial reports that organizations can use as is or as a starting point for their financial reporting needs. In addition to traditional financial reports such as income statements and balance sheets, the app's default reports include examples of the many types of financial reports you can create and customize.

Dashboards, charts, and analytical workspaces

The Sales app uses dashboards to provide an overview of actionable business data across an organization, and give insights on sales data and team performance. Sales representatives and managers can use an out-of-the-box sales pipeline chart in the Sales app to visualize the revenue for an opportunity based on each pipeline phase.

Dynamics 365 apps deliver rich, interactive reports that are seamlessly integrated into application workspaces. By using infographics and visuals supported by Microsoft Power BI, analytical workspaces let users explore the data by selecting or touching elements on the page. They also can identify cause and effect, perform simple what-if operations, and discover hidden trends—all without leaving the workspace. Power BI workspaces complement operational views with analytical insights based on near-real-time information. Users also can customize embedded reports.

Reporting categories

Reporting needs for an organization can be classified into two categories: operational reporting and business reporting.

Operational reporting

Examples of operational reporting include orders received in a day, delayed orders, and inventory adjustments in a warehouse. This kind of reporting supports the detailed, day-to-day activities of the organization. It is typically limited to a short duration, uses real-time, granular

data, and supports quick decision-making to improve efficiency. It also helps organizations identify their issues and achievements, as well as future strategies and actions that may affect the business. Operational reports can empower businesses to determine their next steps for improving organizational performance. Organizations can fulfill operational reporting requirements using elements such as native controls, SSRS reports, dashboards, and business documents.

Business reporting

Business reporting refers to reports detailing operating expenses and financial key performance indicators (KPIs) to business stakeholders so they can understand the organization's overall health and make more informed decisions. This kind of reporting delivers a view of current performance to enable the stakeholders to identify growth opportunities and areas for improvement, and track business performance against the planned goals for the organization.

The data estate and data modernization

A data estate refers to the infrastructure that helps organizations manage all their data, regardless of where or how it is stored. The massive growth in data and the increasing need for immediate insights demand a fast, agile, accessible, secure, and highly scalable data platform.

A common risk for businesses is the quality of their data. If data is weak, unstructured, unsecured, or difficult to use or access, additional work will be required to move, scrub, protect, and improve the quality of the data. Poor data quality also can lead to lost opportunities, failure to consistently attract customers, increased time and expense for cleaning data or processing corrections, and inaccurate or inconsistent KPI measurements.

Organizations typically used legacy applications to both store data and serve as the interface for data collection. Data modernization restructures how data is collected, stored, and managed to take advantage of innovative technology, and is backed by AI and data analytics solutions that optimize the data-review process. Organizations seeking improved business performance must design and manage their data estate based on a modern platform that breaks the data silos and unleashes the data's full potential.

Build a data strategy

For business solutions to deliver the intended results, an organization's data strategy must articulate a clear vision that effectively aligns business intelligence investments with business goals to maximize impact, drive growth, and improve profitability.

Some organizations emphasize collecting data more than analyzing it to drive insights. It's also important to identify gaps and blockers for business objectives and outcomes, and not focus solely on the data, structure, analytics, tools, or technologies.

To define your data strategy, start with the business questions you need to answer to meet your organization's goal of using the data to make more effective decisions that improve the business outcome.

With customers now having access to more information and channels, an organization's data strategy should reflect the customer journey. From a data management perspective, all channels and checkpoints across the customer journey should be represented.

Modernize your data estate

Your data estate modernization strategy should map the current data estate, goals, business processes, and regulatory requirements to aid gap analysis of the existing and desired future state. It should identify key analytics and metrics to improve the business and effectively align business intelligence investments with business goals.

Cloud-based solutions offer modern capabilities based on machine learning and AI to analyze portions of data and automate other processes, such as identifying and solving data quality issues. In terms of analysis, prebuilt models can serve the most common use cases, eliminating the need to build custom models.

Implementing or investing in sustaining legacy solutions brings the risk of a future gap. AI and machine learning solutions take a modern approach to data and analytics that grows as technology evolves, making it possible for organizations to add new features and capabilities in the future—and reduce the risk of being outperformed by competitors.

The modern data estate and the digital feedback loop

Gartner, Inc., predicts that investments in data and analytics strategies will critically affect businesses in the years ahead.

How organizations modernize their data estate is influenced by the four data signals found in the digital feedback loop (**Figure 13-1**).

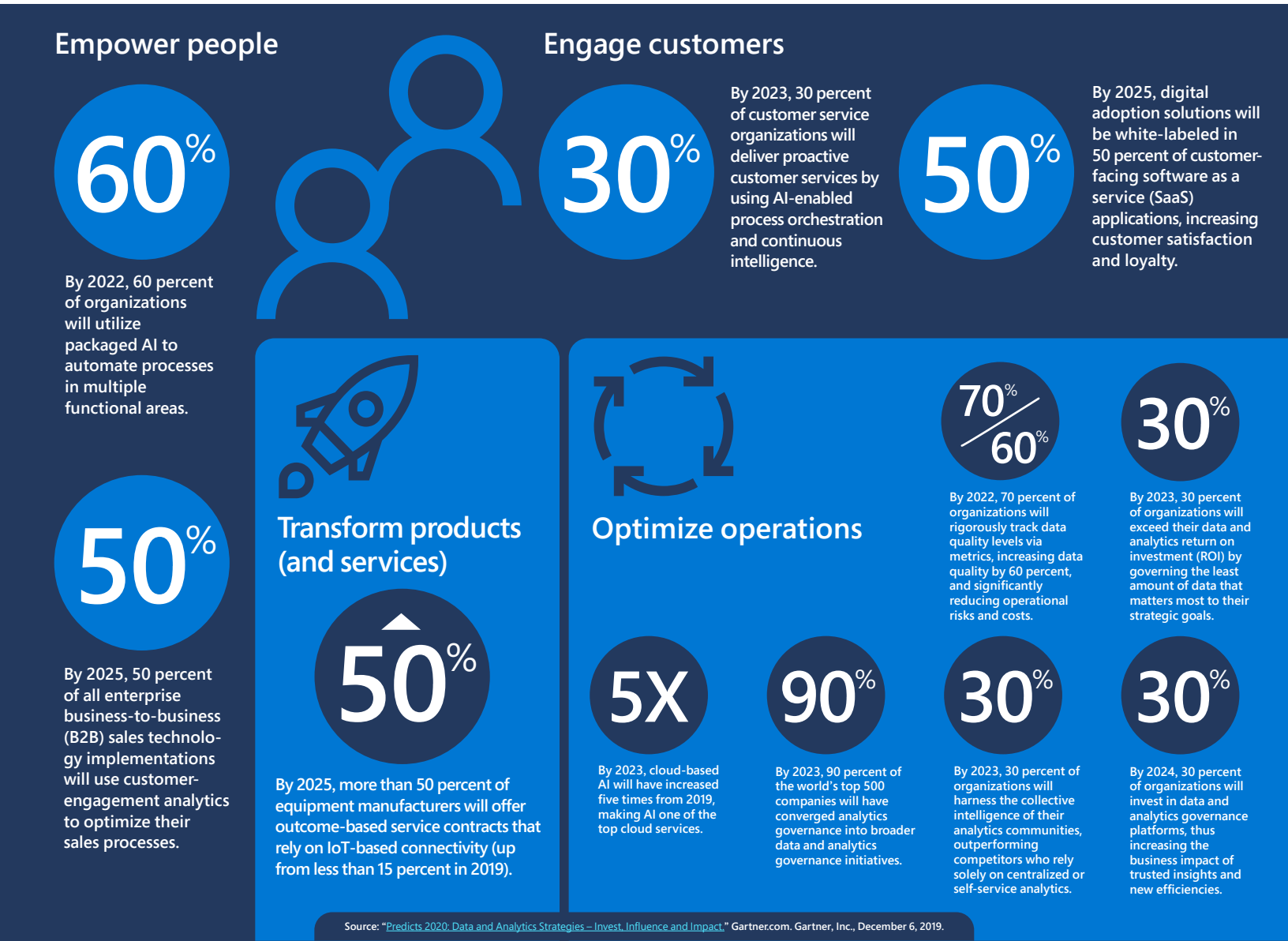
Empower people

To do their jobs more efficiently, employees need tools and resources, as well as timely access to information. Utilizing AI to further automate business processes contributes to better and faster results, which then empowers your people (**Figure 13-2**) to make the best decisions and deliver value to customers.

Engage customers

Modern applications are already capable of delivering insights by using AI and data analytics to optimize business processes and shed light on customer activity. For example, Dynamics 365 apps can provide a customer service agent with insights into a customer's interests and purchasing behavior information in real time, allowing the agent to make suggestions tailored to the customer. These types of insights help businesses intelligently engage customers (**Figure 13-2**) to provide a superior customer service experience.

Fig. 13-2



Source: "Predicts 2020: Data and Analytics Strategies – Invest, Influence and Impact." Gartner.com. Gartner, Inc., December 6, 2019.

Transform products (and services)

Products and services have become sources of data that provide better insights into lifecycle-related information that can be used to benefit organizations and their customers. By analyzing those insights, organizations can in turn transform their products and services (**Figure 13-2**) to take advantage of opportunities and expand into new channels and markets. For example, a smart air conditioner that allows you to set an optimal temperature via an app can also apply what it learns about your household's daily routine to make sure the house is cool when everyone gets home. The air conditioner's manufacturer then uses data collected from all users to enhance the product's smart features, leading to a continuous improvement cycle.

Optimize operations

Cloud-based AI usage is an increasingly common investment area for most organizations—and not just for customer-facing technology. For example, Dynamics 365 for Field Service can use AI to anticipate hardware failures on a manufacturing floor and automatically dispatch technicians before the malfunction. Organizations that optimize their operations (**Figure 13-2**) with augmented analytics, AI, and embedded intelligence will be more competitive in the marketplace.

Components of the modern data estate

The Dynamics 365 platform can be an essential part of your modern data estate architecture. Your business data is securely stored within Dynamics 365 as a function of your day-to-day operations. In addition, Dynamics 365 can export data to or ingest data from various sources to be used in reporting, workflow, applications, or in any other way that is required by your business.

You can also generate insights and analytics from data created and managed inside each Dynamics 365 application. Apps using embedded intelligence, such as the audience insights capability inside the Dynamics 365 Customer Insights app, enrich your data and allow more informed decision-making.

Common Data Model

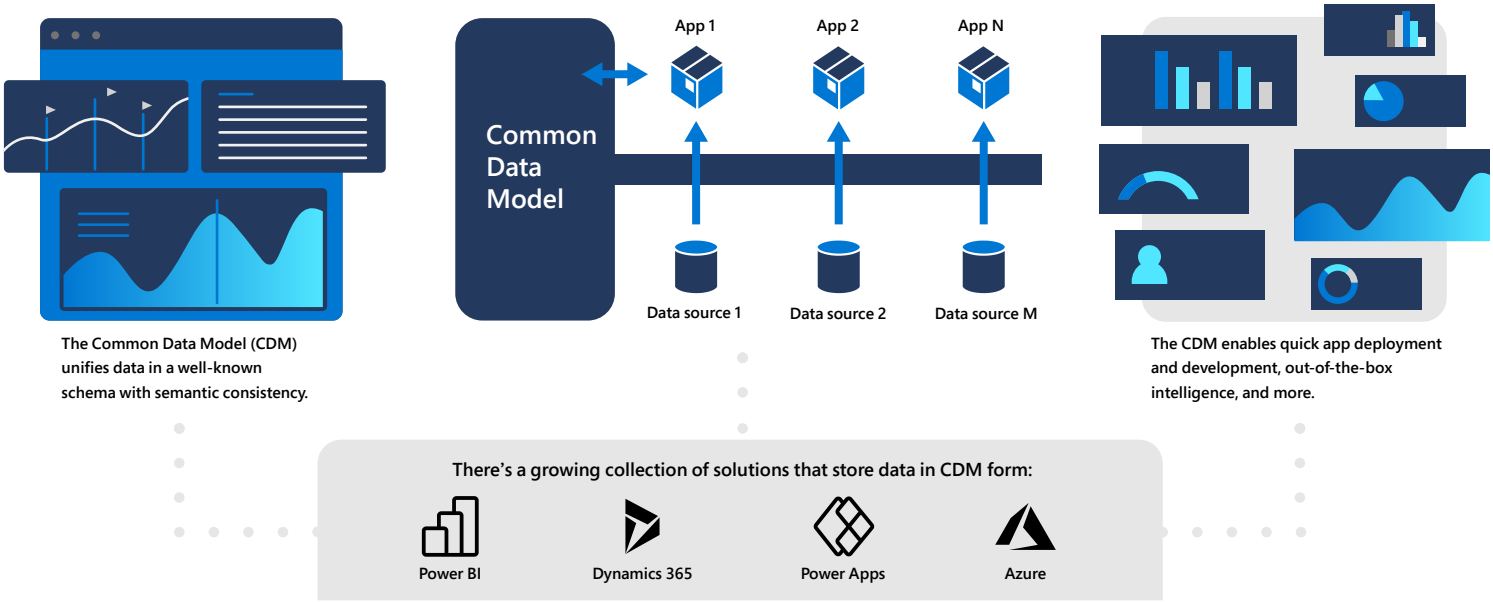
Dynamics 365 data estate components can ingest, store, prepare, model, and visualize data to produce insights that will support, boost, and even transform operations (**Figure 13-3**). The Common Data Model (CDM) is one of the key technologies enabling access to many kinds of information from heterogeneous services and data stores.

The CDM—a shared data language used by business and analytical applications—offers a set of standardized, extensible data schemas to enable consistency of data and its meaning across applications and business processes.

The CDM specification defines out-of-the-box standard entities representing common concepts such as people, budgets, and campaigns. Having your data in a common format for entities with the relevant metadata makes it possible to build apps and services that light up based on the data. The CDM format also makes it easier to consume data in the Microsoft Power Platform, which uses Power BI to get insights from your data. Normalizing the data sets up your organization to better identify opportunities and puts your data in a format that could be utilized for future projects. Dynamics 365 uses Microsoft Dataverse—which is structured according to the

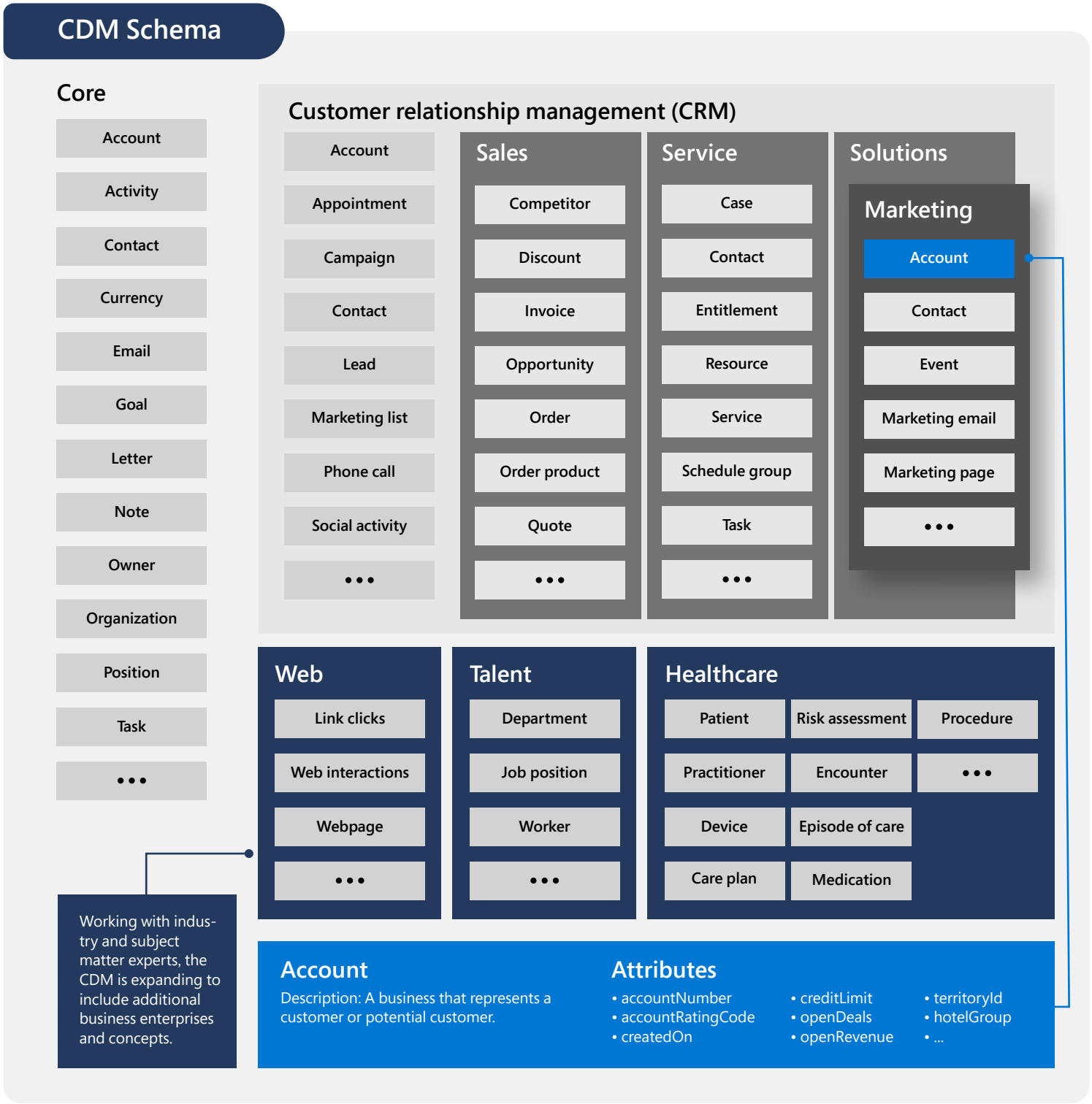
Fig. 13-3

Components of a modern data estate



CDM—to store and secure app data. The CDM structure is defined in an extensible schema, as shown in **Figure 13-4.** This allows organizations to build or extend apps by using Power Apps and Dataverse directly against their business data.

Fig. 13-4



Data unification components

Services and applications that ingest data from multiple sources serve a vital role in a modern data estate. Aggregation from data stores and services provides users with business-critical information supplied in dashboards and reports. The resulting data and events can also be used to trigger workflows and act as inputs to the apps running on the Dataverse platform. Many data unification components are built into Dynamics 365 applications—and organizations can design their own integrations to fit their business needs.

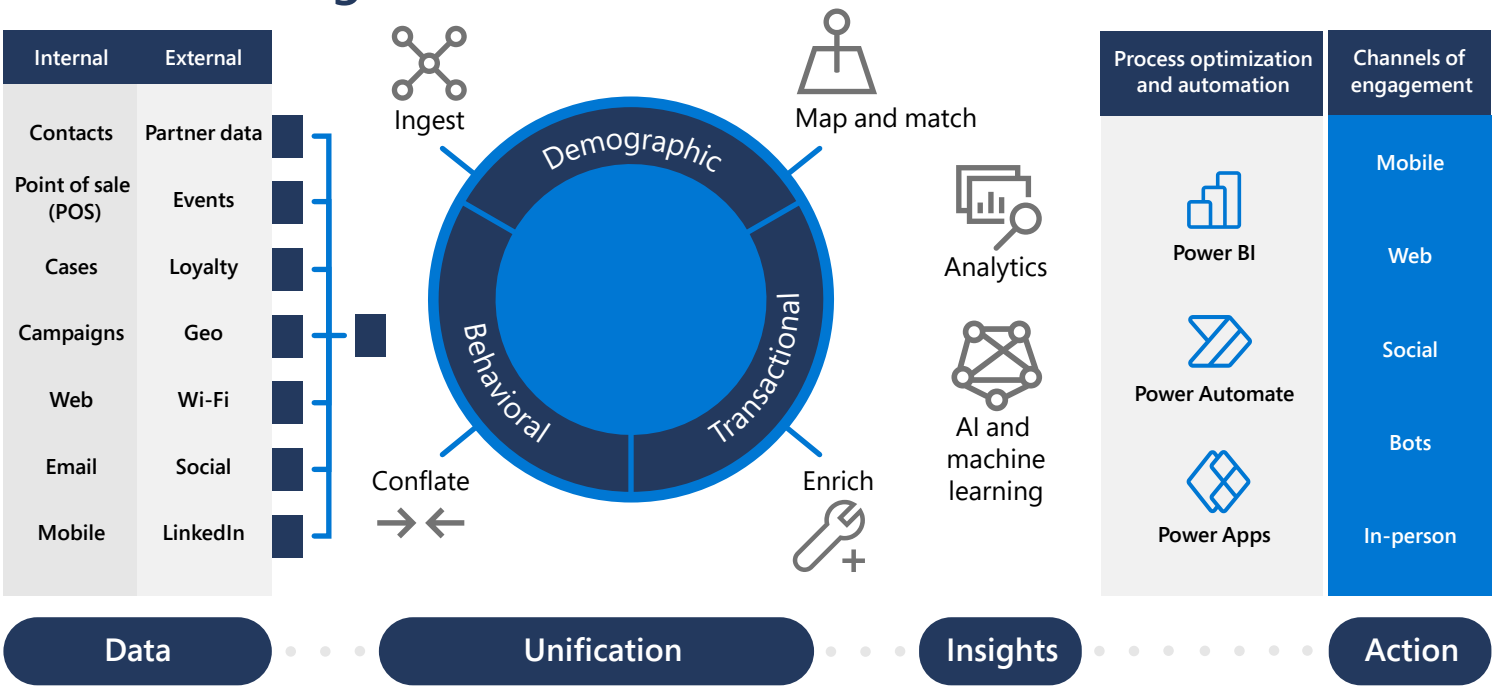
Customer Insights as a customer data platform

Customer Insights is a real-time customer data platform that brings together transactional, behavioral, and demographics data from various sources to create a 360-degree view of your customers.

The Customer Insights solution (**Figure 13-5**) offers prepackaged customer-analytics capabilities such as segmentation, churn analysis, and product recommendations, with Power BI embedded for easy data exploration and visualization. It also incorporates AI and machine learning models, so data scientists can create custom models in the Azure Machine Learning platform.

Fig. 13-5

Customer Insights solution



The addition of Azure Cognitive Services provides text, speech, image, and video analysis, and enriches data via Microsoft Graph.

Dataverse applications

Building an app typically involves accessing data from more than one source. Although it can sometimes be done at the application level, there are cases where integrating this data into a common store creates an easier app-building experience—and a single set of logic to maintain and operate over the data. Dataverse allows data to be integrated from multiple sources into a single store, which can then be used in Power Apps, Power Automate, Power BI, and Power Virtual Agents, along with data that's already available from Dynamics 365 apps.

Data export components

A key characteristic of a data estate is the ability to export data to services as needed. Because data in the Dataverse conforms to the CDM standards, businesses have the flexibility to export data from Dynamics 365 to other cloud services and applications using an industry-standard format, greatly simplifying the standardization of exported and imported data. To facilitate this business need, Dynamics 365 has several built-in export services.

Export to Azure Data Lake

The Export to Azure Data Lake service is a pipeline and the preferred option to continuously export data from Dataverse and the Finance and Supply Chain Management apps to Azure Data Lake Storage Gen2. Designed for enterprise big data analytics by delivering high availability with disaster recovery capabilities, Azure Data Lake Storage Gen2 provides a scalable storage facility for data that can be used in conjunction with AI services.

When connecting from Dataverse, data is stored in the CDM format to provide semantic consistency across apps and deployments. When connecting via the Finance and Supply Chain Management apps, the data is stored in a native table format and as entity metadata that can be used for further analysis.

Data Export Service

The Data Export Service replicates data from the Dataverse database to an external Azure SQL Database or an SQL Server on Azure virtual machines. This service intelligently syncs all data initially, and thereafter syncs the data on a continuous basis as delta changes occur in the system, enabling several analytics and reporting scenarios on top of Azure data and analytics services.

Making data available to all users is risky. Security in reporting and analytics should be aligned with your business solution. Be sure to have your data governance in place, and periodically review who needs access to which information. This may be a time-consuming process, but your data must be protected, secured, and only accessible to those who truly need it.

Bring your own database (BYOD)

The bring your own database (BYOD) feature lets organizations export data from the Finance app into an external Azure SQL database. The app's default operational reports already take advantage of embedded Power BI and Entity store (an operational data warehouse), but there are scenarios where organizations may need to export data for unification or other purposes.

Embedded intelligence

Dynamics 365 apps with embedded intelligence, such as Sales Insights and Customer Service Insights, allow organizations to use AI without depending on highly skilled resources. These apps continuously analyze your data and generate insights to help you understand business relationships, evaluate activities and interactions with customers, and determine actions based on those insights.

Unified data results in a 360-degree view of customers to ensure high-quality decision-making and predictions informed by the best, most recent data.

Power Platform

The Power Platform (**Figure 13-6**) enables organizations to analyze, act on, and automate the data to digitally transform their businesses. The Power Platform today comprises four products: Power BI, Power Apps, Power Automate, and Power Virtual Agents.

Fig. 13-6

Microsoft Power Platform

The low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications

Power BI
Business analytics

Power Apps
Application development

Power Automate
Process automation

Power Virtual Agent
Intelligent virtual agents

Dataverse

Power BI

Power BI is both part of the Power Platform and stands on its own by bridging the gap between data and decision-making. Power BI lets business analysts, IT professionals, and data scientists collaborate seamlessly, providing a single version of data truth that delivers insights across an organization.

Power BI helps you analyze your entire data estate within the Dynamics 365 or Azure platforms, or external sources. Power BI can connect individually with siloed sources to provide reporting and analytics, or it can connect with data stores within or outside of Dynamics 365. As data can come from multiple sources, organizations should analyze how Power BI will connect with those sources as a part of their data estate pre-planning.

Power Apps, Power Automate, and Power Virtual Agents

Built to work seamlessly with data in the CDM format, Power Apps allows you to consume, create, and complement information to contribute to scale and efficiency for the users consuming that data.

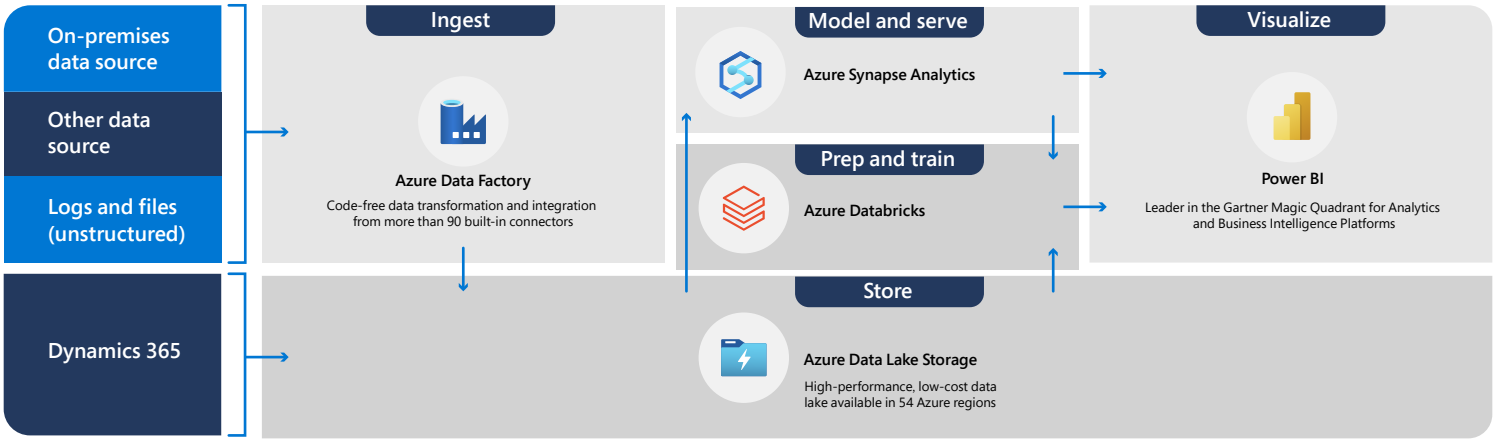
Organizations can also get accurate insights by adding low-code AI tools to their process automation via Power Automate. Power Virtual Agents help you create and manage powerful chatbots—without the need for code or AI expertise—and monitor and improve chatbot performance using AI and data-driven insights.

Microsoft Azure

With Dynamics 365 at the center of the data estate, Azure provides an ideal platform (**Figure 13-7**) for hosting services for business workloads, services, and applications that can easily interact with Dynamics 365. Built-in services in Dynamics 365 let you export data as needed or scheduled. Power BI can aggregate information from Dynamics 365 and Azure sources into an integrated view, and Power Apps can access both Dynamics and Azure sources for low-code, custom applications designed for business use.

Fig. 13-7

Azure features and services with modern data warehouse architecture

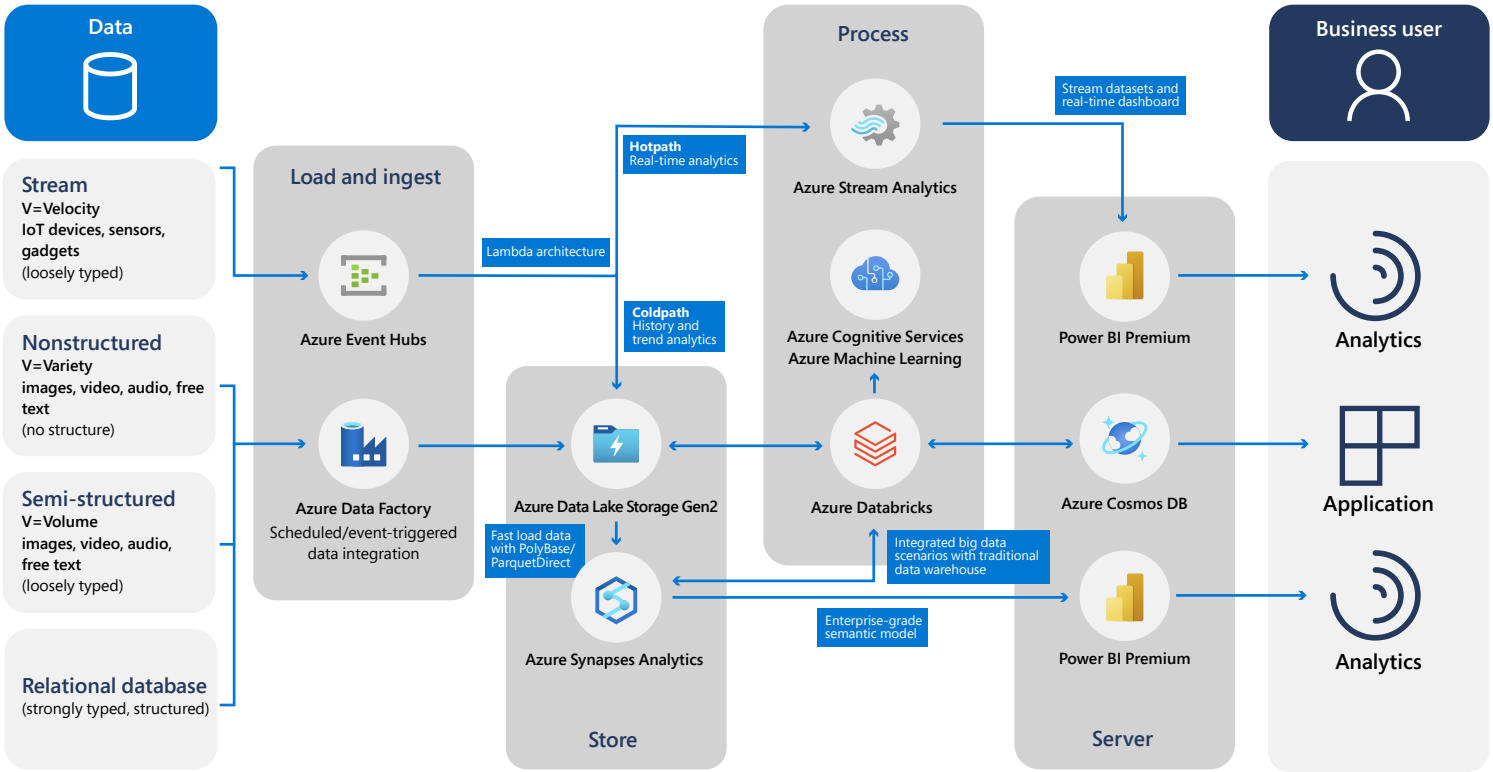


Azure cloud platform

For data estates that include Dynamics 365 and other platforms, Azure can help with unification of data to generate insights. There are multiple options within the Azure cloud platform to ingest, store, prepare, model, and visualize data (**Figure 13-8**), as well as build intelligence on top of that data—whether it's siloed or unified.

Fig. 13-8

Modern data platform reference architecture



Azure Stack

Azure Stack is a portfolio of products that allows you to use embedded intelligence to extend Azure services and capabilities to your environment of choice—from the datacenter to edge locations and remote offices. You can also use it to build and deploy hybrid and edge computing applications, and run them consistently across location boundaries.

Data load, ingestion, and pipeline orchestration

Azure Data Factory is a hybrid data integration service for creating, scheduling, and orchestrating ETL and extract, load, and transform (ELT) workflows. Organizations can use Azure Data Factory to combine CDM data and other data sources, and process that data for unique insights. Azure Event Hubs can ingest data streams generated by a client application.

Data store

Azure Blob Storage offers massively scalable object storage for any type of unstructured data—such as images, videos, audio, and documents—while Azure Data Lake Storage eliminates data silos with a single and secured storage platform.

Machine learning and AI

Azure Databricks provides a cloud-hosted Apache Spark cluster where data engineers, data scientists, and business users can collaborate to train their models and get analytics.

The Azure Machine Learning service gives developers and data scientists a range of productive experiences to build, train, and deploy machine learning models faster.

Azure Cognitive Services puts AI within reach of every developer—without requiring machine-learning expertise. It only takes an API call to embed the ability to see, hear, speak, search, understand, and accelerate decision-making in your apps.

Model and serve

Azure Analysis Services employs enterprise-grade analytics as a service (AaaS) to govern, deploy, test, and deliver business intelligence solutions.

Azure Synapse Analytics is a fast, flexible, and trusted cloud data warehouse that lets you scale, compute, and store data elastically and independently with a massively parallel processing architecture. It can scale across proprietary SQL and open-source databases and manage analytics workloads to provide fast, cost-effective power over any data—whether it is structured, semi-structured, run in real time, or globally distributed.

Azure Cosmos DB is a fully managed NoSQL database service for modern app development.

Synergy

Getting maximum value from your data requires a modern data estate based on a data strategy that includes infrastructure, processes, and people.

Your data can flow inside a cloud solution or via synergy with other components and platforms to provide an infrastructure and processes for analyzing data and producing actionable outcomes.

People are a big part of the process, and the data journey will often start and finish with them. The insights and actionable outcomes will allow them to make better decisions—for themselves and the business.

Conclusion

In this chapter, we discussed how organizations are becoming more competitive, expanding their global reach to attract customers, and using business intelligence solutions to make the most of their data.

While seeking data on how customers interact with their products and services, organizations are acting on that data to give customers access to more content, new purchasing channels, and brand options. By deeply understanding customer behavior, organizations can engage customers proactively and make predictions about their future behavior.

Successful organizations will use data to empower their employees, intelligently engage their customers, transform their products and services, and optimize their operations. They'll also use data to reduce operational risks and costs, and to respond to opportunities and market changes to meet their customers' needs.

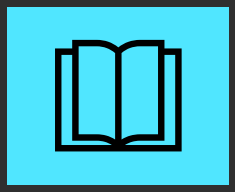
Business intelligence, reporting, and analytics should not be an afterthought. Focusing on building a unified and modern data estate gives organizations access to augmented analytics and embedded intelligence solutions, which will be the differentiator in the future. By changing the ways that they use data to understand and engage with their customers, organizations can reach each customer at the right time, on the right channel, with the right message, for the right outcome.

Due to budget, time, or skills constraints, some organizations decide to deliver a business solution as a first step, with a plan to improve insights and analytics capabilities at some point in the future. Insights and analytics should be addressed in the early phases of a business solution, even if it doesn't yet include all scenarios. Leaving these vital elements until later can affect the business and the user experience, eventually reducing adoption, increasing costs, and giving a negative perception of the value the solution has to offer.

Checklist

Reporting and analytics strategy

- [ ] Map out the organizational data estate to develop a holistic view of different data sources, the type of data they hold, and the schema used.
- [ ] Define your analytics strategy and the tools to support it. Ensure the approach meets the current and future reporting requirements while considering the data volumes and different sources of data.
- [ ] Create an intelligence strategy that considers the reporting needs at various levels, such as strategic business reporting, operational and process reporting, end-user intelligence, and administrative reporting.
- [ ] Take advantage of the out-of-the-box embedded intelligence capabilities in the app.
- [ ] Align the organization insights and analytics roadmap to the business capabilities being delivered.
- [ ] Align to the Common Data Model to take advantage of the standardized business applications schema for better inoperability across systems.
- [ ] Understand the security implications when exporting data from the app to external analytical data stores.
- [ ] Align the reporting and analytics to the broader master data management strategy.
- [ ] Use customer data platform offerings such as customer insights to unify the customer data from various siloed data sources.
- [ ] Focus on not just delivering a report but actions for the users in the context of the application.



Case study

Premier yacht brokerage cruises into smarter marketing and boosts sales by 70 percent with Dynamics 365

To differentiate itself from the competition, a large company that supplies luxury vessels decided to invest in reporting and intelligence. In an industry where relationship-building efforts must be incredibly precise and personal, any insight into a customer's mindset is invaluable.

The company implemented a solution based on sales, marketing, and finance data, using Dynamics 365 and Power BI to help identify trigger points that would help increase sales to charter clients, strengthen customer relationships with improved marketing activities, and seamlessly move customers through the sales cycle.

This solution allowed the company to build intelligence on top of a centralized system running in the cloud, and facilitated a move from legacy systems where data was siloed in different applications, channels, and departments.

Dynamics 365 apps made it possible to separate the features each department would use while accessing the same data. The marketing and sales teams started customizing their content and campaigns to nurture leads toward a sale, and used Power BI reports to generate insights that helped them identify the best prospects and create winning proposals.

Integration with external applications brought in data from other sources to improve the company's bespoke offerings. For example, the sales teams could collate even more information about their products and send it to customers in just a few minutes—instead of spending hours manually putting together the data.

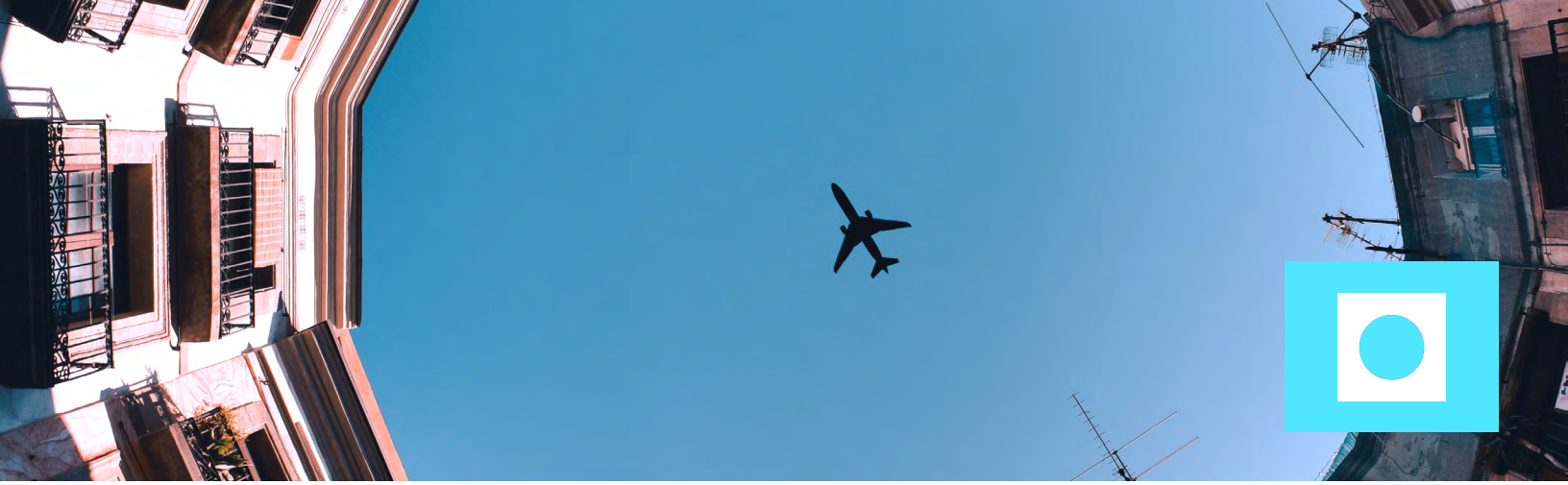
After deploying the Dynamics 365 solution, the company experienced a 70 percent increase in sales. They next brought in more departments to create a central hub for all activities, simplifying every area of their operation. For example, the sales team's process integrated everything that the accounting department needed to efficiently close a deal and reduce the possibility of human error.

The company is also using Dynamics 365 and Power BI to uncover market trends, as augmented analytics help the company learn about their customers in ways that were not possible before—and build authentic relationships that will last long after the initial sale.

14

Guide

Testing strategy



Overview

During the implementation of a solution, one of the fundamental objectives is to verify that the solution meets the business needs and ensures that the customer can operate their business successfully.

The solution needs to be tested by performing multiple business execution rehearsals before the operation begins and the system is taken live.

It is imperative that proper testing of the solution is completed before deployment to production. This helps avoid costly challenges that could delay the adoption, create negative perception of the final product quality, and reduce the confidence to run the business on the new solution.

The goal of testing is to detect defects in the process in order to ready the solution, processes, and people to operate the business in production application efficiently and accurately as soon as possible. Testing allows us to detect defects as early as possible in this process, in a systematic way and at the recommended time per the test type.

This chapter explores the components of:

- The value of testing
- Types of testing
- Testing at the right time with clear scope
- Defining the strategy

Defining a comprehensive test strategy for your implementation is a combination of considering the project scope, testing planning with a test plan and test cycle schedule in combination with the phases and solution versions availability, and selecting all the test types and how to execute them.

- **Define scope based on your processes and detailed test cases**
- **Create a test plan with the necessary test cycles**
- **Select the adequate test types for your project**

We explore each of these areas in more detail in this chapter.

You can also find recommended practices for executing the testing, how to deal with the outcomes, and finally, reaching a business sign off that confirms the readiness of the solution.

Having a properly planned and executed test strategy eliminates the practice of doing parallel testing in production since it is no longer a recommended risk mitigation approach during customer implementations.

We look forward to identifying the key components of your testing strategy in this chapter.

Focus on quality and keeping scope

Testing is a continuous task under application lifecycle management. Not only is it critical during the implementation of the solution, but also during the Operation phase. It is a continuous evolution that keeps bringing fixes or extending the solution. In the beginning testing is completed manually, but over time with automation we can make the process far more efficient. Our objective is to ensure the quality of the solution is always meeting customer expectations.

To align with the scope of the solution, testing needs to focus on validating the business processes in it to stay on track. Microsoft Dynamics 365 Business Applications are rich with features that can distract testers easily during testing, so it is important to focus on the features that add value to the business process. By only focusing on those features and the requirements needed for them, we can get to the starting line of initiating operations for the solution being built.

The approach for testing

Depending on the methodology you apply, agile, waterfall, or a hybrid of the two, the testing frequency differs for each iteration.

Regardless which approach you follow, you need to look for a high-quality testing approach to confirm readiness.

On top of your implementation methodology, you need to have a clear strategy for testing, as illustrated in **Figure 14-1**. This strategy is composed of:

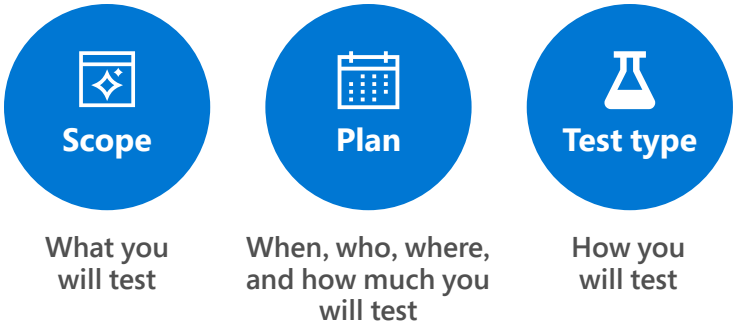
- A clear testing scope defined by your processes and requirements to validate.
- A comprehensive plan that brings clarity to when, why, who, how much, and where to test.
- The different types of testing with descriptions of how the testing is executed based on the complexity of the solution and minimum quality gates allowing us to detect defects early in the process.
- A comprehensive tracking mechanism that allows you to monitor overall progress, pass/fail rates, ownership, and issues tied to specific test cases.
- A clearly defined process for adding or modifying test cases during each individual test pass.

Explore each component to define the testing strategy for your Dynamics 365 implementation.

Defining the testing strategy

After observing thousands of Dynamics 365 deployments, we can see that most customers do a reasonable level of testing before go live. But the difference between each of those implementations is how thorough the implementation team is at defining their strategy, and the quality and depth of the testing.

Fig. 14-1 Test strategy



In general, a good strategy does not change in approach between implementations. What changes is the tailored artifacts used for each implementation, for example, the process to define the scope can

Make sure that every implementation you execute has a clear and solid testing strategy.

be the same across projects, but the scope itself differs. Investing in a good strategy and making sure the implementation teams use it consistently increases the quality of the project outcome.

Scope of the testing

The scope of the testing is defined early in the project after defining the solution, and it is refined as the solution blueprint takes shape. What changes throughout the project is the type of testing that comes into focus as you move through the implementation lifecycle and the ability to actually test the solution.

During the implementation, the number of test cases in the testing scope keeps increasing as you progress in building the solution. After golive, the scope is focused on maintaining the quality as you update the solution, but testing can increase if you continue to expand.

Consider the testing journey as an incremental process, as illustrated in **Figure 14-2**. Regardless of the timeline, you need to keep two things in mind:

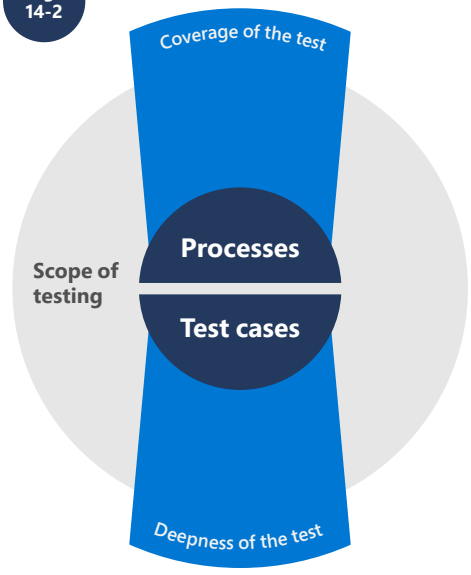
- A clear view of the end-to-end business processes.
- A clear definition of the depth and detail of the test case design, but also the variations of testing, for example edge cases and negative testing.

Testing connected to processes

To progressively build your solution you need to determine the implementation processes and the requirements connected to each of them. As we build the solution and make it ready for validation, the scope of the testing increases as we add more processes. During the project progression, we go from a simple functional test to an end-to-end testing scope during user acceptance testing (UAT). Always connect the test to the process being validated since this is how the end user confirms if they can perform a connected action to execute the business process in place.

This connection is primarily important for testing connected functional requirements, there is testing that checks other aspects of the solution out of the process, for example testing quality of data, testing latency in the connectivity, etc.

Fig. 14-2



Connect the test to the process being validated. This facilitates tracking, reporting, and final sign off on readiness by the business team.

Testing is not an artifact of the development process; it is an artifact of the business process created when you implement business applications. Plan your testing by connecting those tests to the processes and include processes that are executed outside of the new application being implemented.

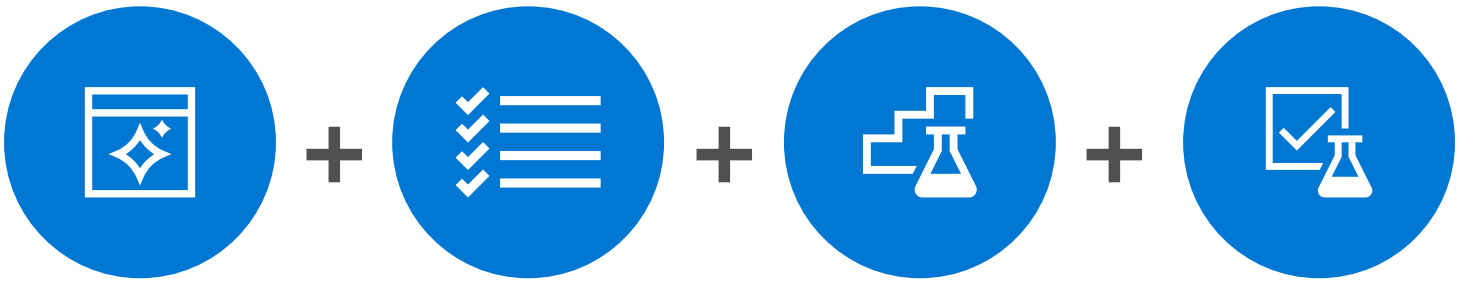
Test case definition

An important part of the scope is defining the test. Proper test documentation is key, and test cases (or test scripts) are the artifact that achieves it. The nature of a test case varies depending on the test type. Clear and detailed test cases guide the tester to validate the expected outcomes without distractions and align to the scope of the solution.

Writing a good test case requires a series of tools and tasks that help you validate the outcome. Dynamics 365 Business Applications provides some of the tools to build test cases faster, for example, the task recorder in the Operations apps where you can dynamically capture the steps needed to execute a task in the app.

Test cases should reflect the actual business execution in the system and represent how the end user ultimately operates the system, as illustrated in **Figure 14-3**.

Fig. 14-3 Components of the test case



Test cases should be composed of, at minimum, the following:

- The process and requirements that the test case is covering.
- The prerequisite, or entry, criteria to execute the test, which can be dependent on other tests or necessary data to produce the expected outcome.
- The reproduction steps.
- The expected outcome, or exit, criteria that helps to confirm the readiness.

Finally, when you design your test cases, design for what you expect to happen based on the process objective but also on what it should not do. Plan for tests that can break the process.

Azure DevOps is great tool for documenting test cases and connecting them to the solution development lifecycle. Having this artifact in Azure DevOps allows for tracking any bug discovered during testing and helps to trigger the fix and plan for the next testing cycle while also determining the coverage of your test; for example, process, user story, requirement, design, etc. This is depicted in **Figure 14-4**.

Planning

We have defined what is needed to determine the scope of the testing using the processes as a foundation. With this valuable information, we can determine how to execute the testing. This is where the planning becomes important.

Planning for testing is a fundamental part of the testing strategy. The next section describes the minimal components needed to define a recommended testing plan. This strategy can be used to implement any of our business applications.

Fig. 14-4

Process
Prospect to cash

Requirement
Create sales order based on servicing system part request

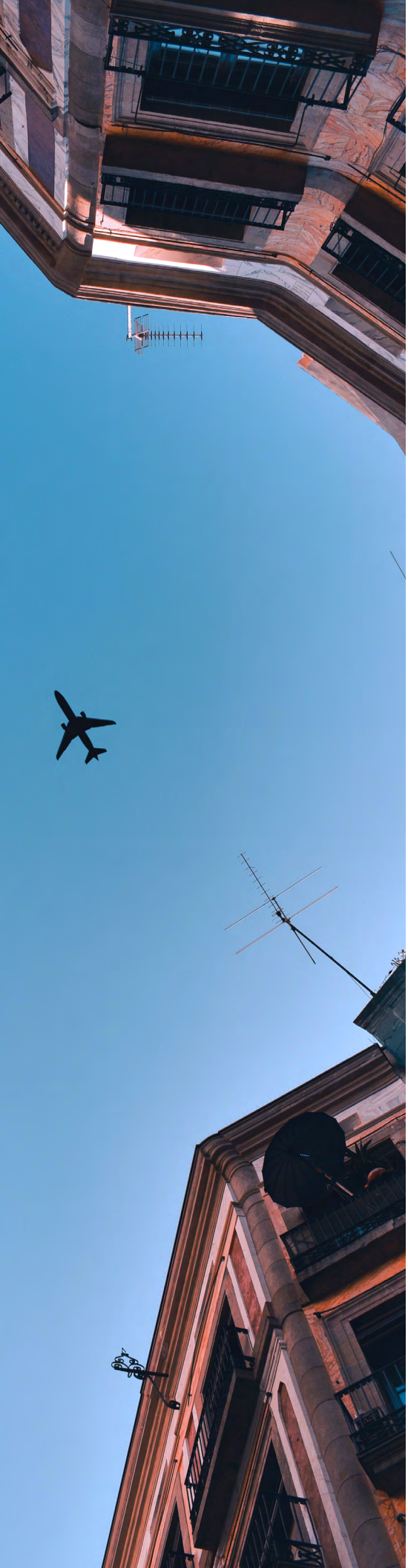
Prerequisites
Customer with no credit line available
Stock available

Description
Dealership request parts for recreational vehicles out of warranty

Test case ID	Test steps	Test data	Expected results	Actual results	Pass/Fail
PC-20	1. Create sales order header 2. Check credit line in order 3. Create order line 4. Validate and post sales order	**Customer:** CU050 - Great Adventures Dealership **Part:** P001 - All weather speakers **Qty:** - 4 pieces **P001** - All weather speakers	Sales order cannot be posted	Sales order is posted	Fail

Tester notes

Credit check is not stopping the order to be posted. Customer setup with wrong credit limit. Data quality issue.



During the planning stage of the testing, you need to be able to answer and provide clarity for the following questions. This helps design and define the frequency of the testing based on the necessary iterations or test cycles to secure a high-quality solution.

- When do you need to start testing?
- How do you control the versions of the solution to be tested based on the progressive readiness of it?
- How do you document the outcome of the testing?
- Who participates in the testing?
- Where does the testing happen in terms of environments and type?
- How is the ownership of the outcome defined?
- How deep you are going to test?
- What types of testing do you execute based on the project needs and solution complexity?

This helps to plan the quality control portions of the solution and must happen at the start of the Initiate phase of the project.

The plan for testing must be documented and signed off on by the business team prior to its execution. This is important because it leads into other types of planning, like environment planning, that determines where the test is done.

Creating a test plan

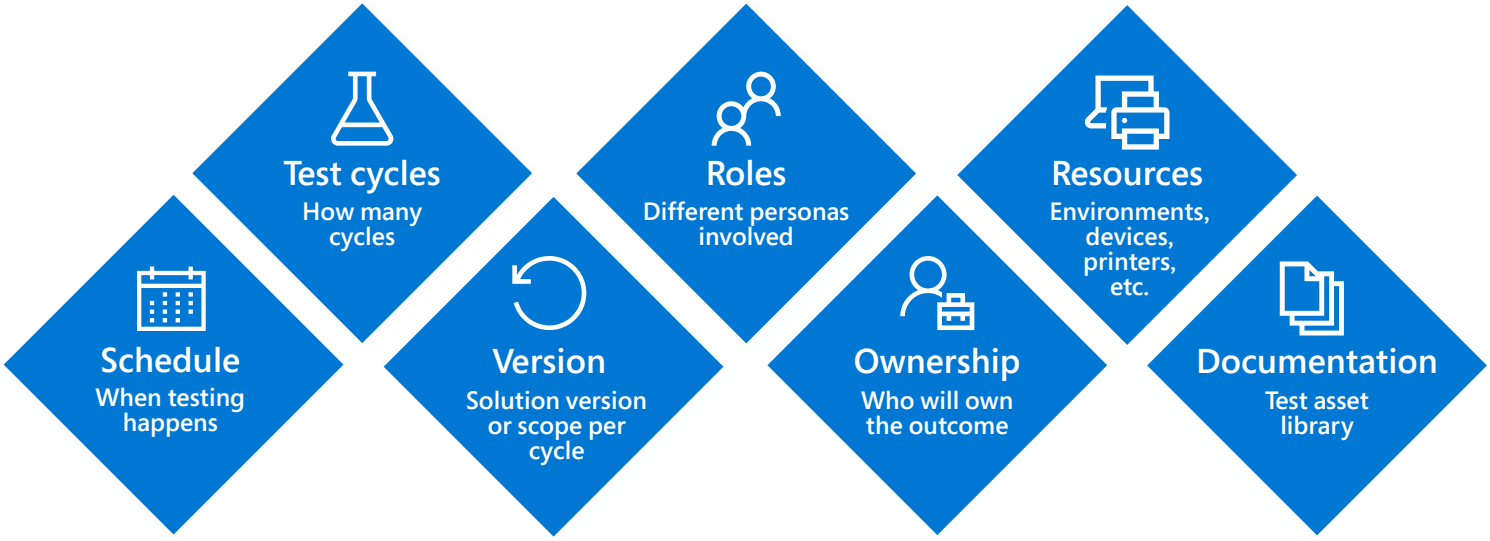
The test plan is the blueprint for executing your tests. It establishes the guard rails to stay on track in terms of quality assurance for the solution, schedule of testing, and resources needed, and it contains the scope of the testing.

Depending on the complexity of the project, this artifact can be developed by the same implementation team. In larger, more complex organizations, this is prepared in conjunction with the quality assurance team to secure a neutrality at tracking and reporting the outcome of the test, as illustrated in **Figure 14-5**.

Always create a test plan for your project regardless of if your implementation is simple, complex, or if you are implementing a standard solution or extending the solution.

Fig. 14-5

Components of the test plan



The test plan brings transparency and helps keep your goals and objectives on securing a quality solution. It contains important information that guides the team on how to conduct the testing.

Always create a test plan at implementation and obtain sign off on it by business stakeholders.

When does testing happen?

It's simple: the testing should start as early as possible. It is never too early to start testing. Your testing cycles are incremental in terms of the scope. Your first test cycles could be to test the standard functionality to validate how far are you from the overall desired outcome.

Do not wait too long, or until the end of the implementation when you think you have built the entire solution, to start testing. Testing late adds serious risk to your implementation by having issues and gaps that are hard to fix in a short period of time. This becomes a constraint, especially when you have reduced time available for time sensitive projects tasks like preparing to go live. Always plan to avoid testing close to your go live date since it leaves no time to resolve issues. Poor planning or no testing can ruin a project.

Testing cycles or iterations

Testing cycles are the frequency of the testing aligned to the availability of the solution in scope for the testing at a given period time in the implementation. Test cycles act as comprehensive testing milestones.

The successful outcome of the test cycle confirms the readiness of the solution to meet the requirements of the business. There are different terms used to describe this testing event, for example conference room pilots, testing iterations, testing cycle, etc. The important message here is that the testing event is comprehensive.

To implement Dynamics 365 Business Applications, it is recommended that the scope in each testing cycle is based on business processes. Test cycles by application module, or organizational department in isolation, risk losing the big picture and thereby limit the effectiveness of the test.

Every testing cycle represents a key milestone in building the solution. Consider every test cycle as a mini go live, where you are rehearsing the business operation at every test.

Solution versions and alignment to them

We discussed the importance of planning the testing cycles process as part of the scope, this scope also triggers the readiness of the code and data that supports the test cases connected to those processes. After determining the planned scope of the test cycle, you should plan your development efforts to cover those processes and the readiness of the required data in terms of master data and configurations.

Your solution version needs to be controlled at every test cycle so you can have a clearly defined milestone. For every new milestone, the solution keeps expanding in scope and complexity.

The solution version is a combination of the version of your:

- Code
- Configurations
- Master data
- Migrated data

Different teams need to coordinate and plan how to combine their efforts to prepare for the testing cycle.

Each component of the solution version is very important, and all of

Align your solution version to the testing cycles and coordinate the readiness of all the dependent parties to provide their own artifacts. Missing one necessary artifact can cause delays in the implementation, introducing risk.

them need to come together as part of the solution. This is especially important with data, the earliest you bring migrated data into the mix the better. One of the most common causes of errors during the first days of operation is poorly migrated data.

Ownership definition

Having clear expectations of who takes care of the testing outcome is key. Testing can result in bugs, but also it can result in discovering configuration issues, gaps, or new opportunities where improvements can be applied in future cycles or solution releases. Based on the type of outcome, we need to define who takes ownership of the fix and what test plan is needed to account for that.

- Bugs go back to developers for resolution.
- Configuration issues go to the consultant connected to the test case.
- Gaps go to the project manager for further discussion with the stakeholders and implementation team.
- Standard product issues go to Microsoft support or any other third-party solution provider connected to the issue.
- Conceptual design issues go to the architects on the implementation team.
- Process improvements go to the business SMEs.

One common pattern for implementations of standard functionality is the project team challenging the need to thoroughly test when the application has not been modified. The reality is that implementation teams are testing the specific business solution being implemented, not the standard software. You are not just testing software, you are testing the people being trained, the processes that represent the operation, the data that keeps the business alive, and finally the technology.

Documenting the outcome of the test cycle and assigning clear ownership for the fix facilitates tracking and resolution. Plan for testing thoroughly, even if you are implementing standard product.

The right environment for the right type of test

Where to test is dependent on the type of test being executed and this definition impacts the environment planning discussed in Chapter 9, “Environment strategy.” Refer to this chapter for the different types of environments available and their overall purpose. Our focus is on the environments where you can do proper testing based on the test type.

This environment can be a different variation of a development or test environment, but it is important that you never complete or plan for regular testing in production environments.

Production environments are meant to run the real business operations. Doing tests, or making changes to it without being tested first, is a high risk for the stability of the solution because of the unknown variables it can bring. The only test type that can be executed in a production environment is a mock cutover test, and this test happens for a very limited period before the solution is live.

Plan for non-Dynamics 365 testing environments where you have a dependency in the solution that requires validation.

You can require testing be done in an environment different than Dynamics 365 environments, like in the case of integrations. Plan for the availability of the testing version of third-party systems that will integratewith Dynamics 365; integrations or other dependencies will need to be validated. This is a commonly missed opportunity during implementation.

Plan for additional hardware required to emulate the real operation like scanners, printers, handhelds, etc. These resource needs are dictated by the test case.

Documentation

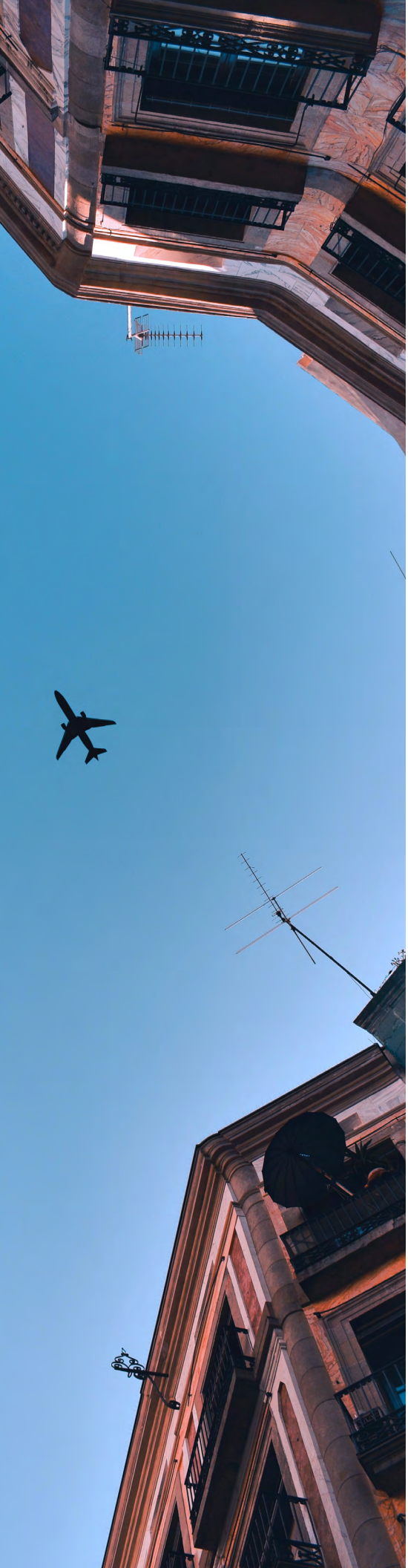
Documenting testing has two angles, the test cases and tracking the outcome of the test.

Planning to document the results of your testing cycles helps to highlight the wins, but also the bugs and patterns. This report determines the next course of action and can impact future milestones. It allows stakeholders to make decisions to correct the paths if necessary.

Use Azure DevOps to build dashboards that can help to report progress and quality of the testing.

Having a model to document the outcome of a test case and test cycle allows the implementation team to validate the performance of the test cycle. Preparing to document and report the outcome highlights other challenges as well, though not necessarily ones connected to the test cases itself. These issues can be solution performance, connectivity issues, usability, continuity of the transaction, gaps in the product, etc.

Other important benefits of documenting the outcome are that it tracks the progress of the testing and keeps the big picture in sight. It allows us to detect and build the backlog for new non-high-priority opportunities that are discovered during the testing. It can also trigger a change of the statement of work when the newly discovered requirements are critical. This is highlighted as project risk.



The different roles involved in testing

The people involved in testing depends on the type of testing being done. For example, performance testing types require more technical roles and consultants familiar with that type of testing, while user acceptance testing types require your end users.

Testers can be developers, functional consultants, customer subject matter experts, quality assurance testers, or end users.

It is important to plan for the resources required for building your solution and for the different types of tests that require special skills. Consider the volume of test cases involved and the business areas impacted so you have good representation of the business.

Roles for testing are determined by who to involve in tests depending on the following variables:

- Test preparation (system, data, test cases, training, etc.)
- Test execution
- Test administrator or manager
- Test results triage
- Test reporting
- Test defect fixes

The business team involvement during testing is critical, they own the solution and plan for enough time to allow for proper testing by the key business users. One common pattern of failure is poor involvement from this group of testers.

Test types

In the previous section, we covered the different roles and environment types that may be needed for testing, but we mentioned the need to consider the test type. Let's now combine the concepts in this section so you can see how to use different test types to implement Dynamics 365 under the Success by Design framework.

In **Figure 14-6**, we show when we recommend executing the most common test types when implementing Dynamics 365 apps across

the different Success by Design implementation phases. Note that some test types are more relevant to specific phase than others.

The minimal recommended test types are:

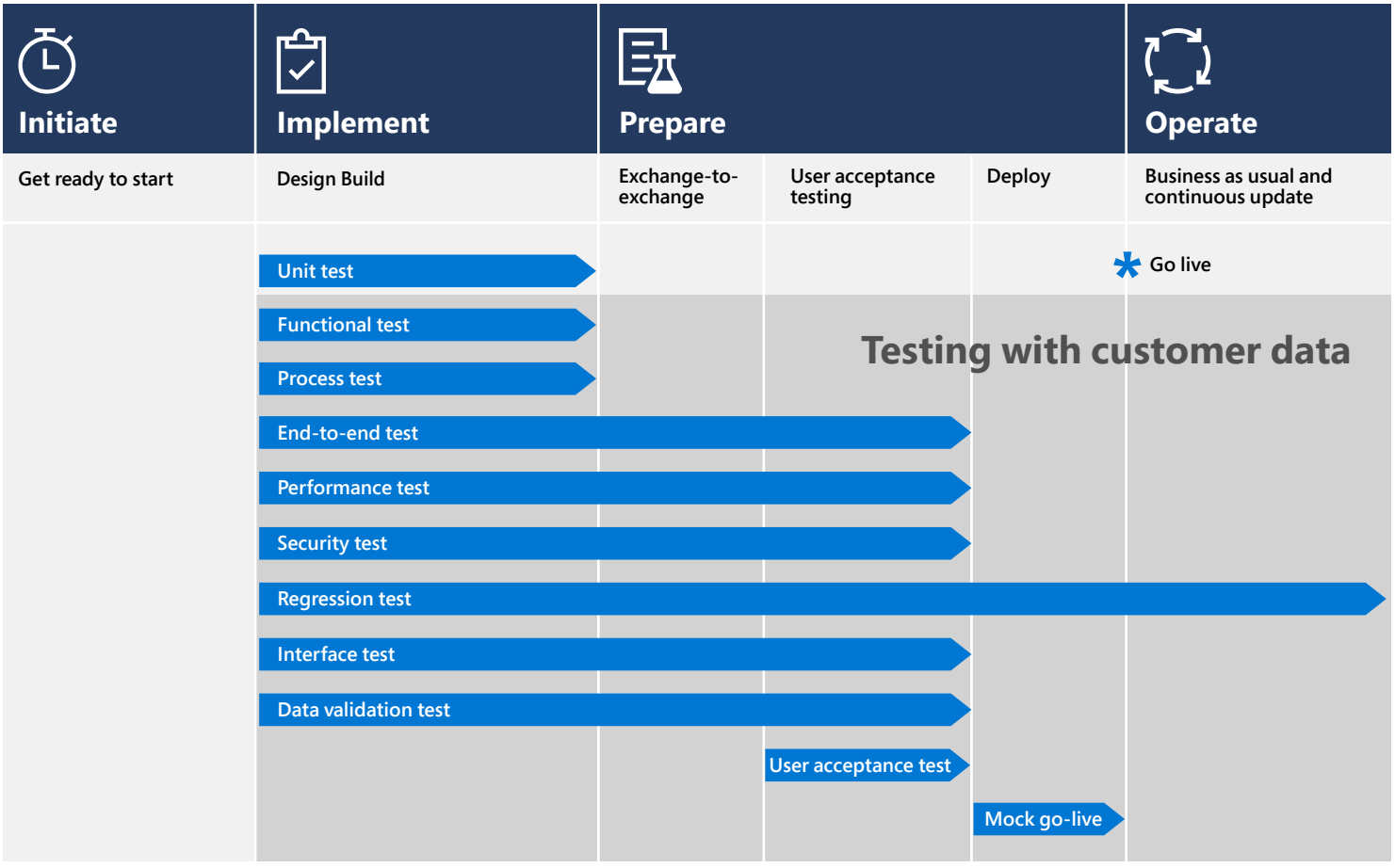
- Unit tests
- Functional tests
- Process tests
- End-to-end tests
- Performance tests
- User acceptance tests
- Regression tests
- Mock cutover

Each test represents an incremental testing approach. Some are required for any implementation and others depend on the specific implementation circumstances based on the complexity of the solution, performance, and security requirements. Some test types are especially relevant for specific stages in the implementation and others can run for the life of the solution. For all cases you need to test with real customer data and migrated data as soon as possible.

Test types need to be planned with consideration to the complexity of the solution, performance, and security requirements, and should be executed during specific times within the implementation.

Fig. 14-6

Testing throughout the solution lifecycle



See it in action

Let us assume we are building a solution for a customer that ships goods and uses a specialized transportation system that is part of the final solution. For this example, we use unit testing to test specific validations during a sales order confirmation and interface with the transportation system using Power Automate. The unit tests are written and executed for each validation requirement and interface in this scenario.

Unit test

This test type focuses on individual function, code, and configuration testing. It is done by developers and is the lowest level component of the testing. In this test type, the developer verifies the requirements, validates, improves the code design, and finds and fixes defects.

This testing happens in a development environment and is expected to be done during the implementation mainly, or during any bug fixing. It is a fast, iterative process. This test type is required if you are customizing or extending your solution and is one of the first opportunities to introduce a quality check in the solution. At this point you should validate performance behavior of the individual components in addition to keeping security in scope. Being diligent from the start with this test type can save you important time during implementation by avoiding rework or bug fixes for issues that should be detected during this test.

See it in action

Following the previous example regarding the sales order validation, the functional testing of the interface with the transportation system focuses on the configuration and setup, like customer data, products, pricing, warehouse setup, and the dependencies with payment processing and other module-related configuration and parameters.

Functional test

Functional tests can be either a manual test or an automated one. They are done by the functional consultants, customer SMEs, or testers. The purpose of functional tests is to verify the configurations of the solution or any custom code being released by the developers. The primary objective is to validate the design as per the requirements. This is done in a test or developer environment during the Implement phase of the Success by Design framework. At this point, testing automation can also be introduced.

This test type is the first test to be done by consultants and customer SMEs. Consultants still need to do it, as it is important that the first line of testing is done by the consultant's previous customer testing at least to verify stability of the feature.

At this point the consultants need to map the test to the requirements and processes and the test case is detailed. The link with process is agreed upon with the business so the test case is not focused only on the gap, but also to fit in the whole picture of the solution.

The data used to test beyond this point requires it to keep evolving continuously using customer data and should reflect the reality of the operation.

Process tests

The solution continues to be built, with unit testing being done by developers and functional testing by consultant and customer SMEs. The work from the development team is validated, and bugs and corrections are completed while the test is mapped to the process.

This is the point at which running connected test cases is ideal. The point where we raise the bar in our testing approach by connecting a collection of test cases that are all connected to our process.

Process tests can be manual or automated at this point. The business has visibility and is actively participating in this testing. It is our first opportunity to go beyond testing that focuses on just functionality or modules, and it gives different people the opportunity to start playing an important role in the test by handing over the outcome from one test case to another. This should be your first comprehensive testing cycle, starting during the Implement phase of Success by Design.

The objective is to verify the solution allows us to operate business scenarios, and the testing needs to be done in a test environment.

See it in action

At this point we know how our sales order function works, but now we want to test how the sales order works in connection with test cases preceding and following the process. Let us call this process prospect to cash. We want to test the collection of money after the shipment of goods happen. We involve other teams in the implementation team that handles that part of the process, and we validate that we can generate the invoice and collect money while closing the cycle. From the business point of view, the customer can confirm they can sell, ship, invoice, and collect the money so the process prospect to cash works.

Tracking the outcome becomes more critical at this point, so bugs fixes and other dependent improvements must be tracked. During this time, tracking tools like Azure DevOps become crucial as part of your application lifecycle management.

For this test type, role security becomes crucial. We want to make sure we have the right segregation of duties and that the different personas can operate as they should when they are running the solution in production.

This testing should look for unexpected outcomes of the process, commonly known as negative testing, and not just the happy path.

Performance of the solution starts to become more evident with this

test type since more testers are involved, but also because it touches more areas of the solution so the solution can start to enter into stress mode when executing parallel functions. Nonfunctional requirements need to be observed; for example, proper batch processing jobs are properly configured.

End-to-end tests

After validating all the individual processes, it is time to connect all of them and increase their complexity with new process variations. This is the first test cycle that looks like a complete operation.

This test is manually executed, but automation starts becoming more relevant in preparation of future iterations, regression testing, and system maintenance because of continuous solution updates even after go live.

The test is done by functional consultants who are preparing the cycle and guiding the team, but the overall cycle is mainly executed by customer SMEs and testers.

The main objective of this test type is to validate all full business processes in scope and it needs to be done in an integrated test environment since now it is connecting to other systems that interact with the solution. It is iterative and the prerequisite to being able to execute your user acceptance test (UAT).

This test type starts in the Implement phase and goes through the Prepare phase as per the Success by Design framework.

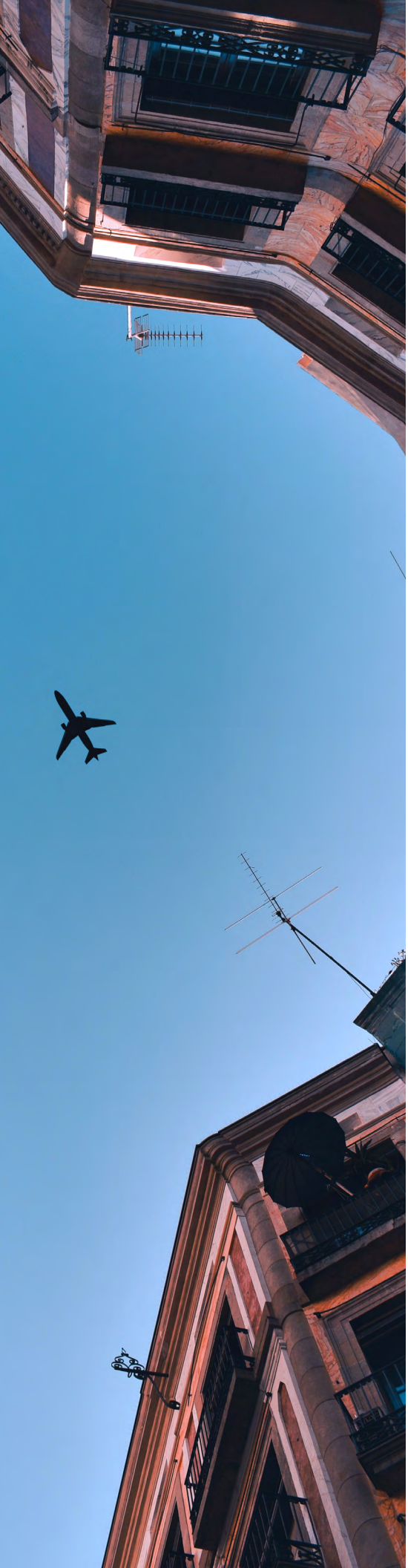
It is important to execute as many of test cycles with end-to-end testing, doing only one and at the end of the build of the solution is not recommended since it add risks to confirming readiness by having less time to react to final fixes.

Another important aspect here is that you run this test not just with customer real data but with migrated data coming from the legacy solutions as soon as possible.

See it in action

On previous test cycles, we were able to collect the cash of our sale. Now we want to connect other processes dependent on this one; for example, accounting process to report taxes, update and interact with other systems like the transportation system, optimize inventory, and introduce new products. This allows us to combine different Dynamics 365 apps to work together.

Plan for this test by making sure you include real customer data and migrated data.



This test is key to validating the entire solution works in conjunction with other systems that are part of the business, and testing is done by having role-based access control enabled so it validates a real end-to-end test.

Performance tests

Successful testing is not complete until we not only make sure we can run the business on top of the solution, but also that we can do it at scale. We are implementing Dynamics both for today and for the future, and we need a solution that lasts and performs.

Performance testing is required, especially if there is concern over operational volumes, peaks, variety of integrated systems, usability challenges on the UX design, etc. In this section we focus on some of the considerations you need to keep in mind for performance tests, but we have dedicated an entire chapter just for this in Chapter 17, “A performing solution, beyond infrastructure.”

We put special emphasis on this test type since there are many misconceptions defining whether this test needs to be executed. Our experience has shown that performance is one of the most common reasons for escalation since teams often miss this test when it is needed.

In the end, a performing solution is a combination of data, code, configuration, and infrastructure. Microsoft, partners, and customers play important roles for this test type, but we look to the implementation teams to play their important role on validating performance beyond infrastructure and at the right time of the implementation—so plan for it.

This test is executed by developers, functional consultants, customer SMEs and testers mainly.

The objective of this test is to ensure the solution performs while focusing on critical processes that require scaling with load and growth. Not all the processes are involved in this test.

This test is completed in a dedicated performance testing environment.

See it in action

In previous example we involved invoicing and the integration with the transportation system. Now we want to test if we can process the operation peaks considering the seasonality of the business. Invoice posting operation is crucial since we need to have almost real-time interaction with the transportation system as per the business needs, so we need to test a day in the life with relative volumes across key business processes, not just prospect to cash.

The basic version of performance testing starts during unit testing so developers can influence proactive improvement of performance. Regular dedicated test environments can also be used depending on the load to be tested.

Performance testing happens as soon as the critical processes are available to be tested throughout the implementation.

Performance testing needs to be visible at the start of the implementation and be part of the test plan. It requires agreement on the performance test objectives, and it can require proper environment planning to provide the necessary resources.

Do not delay performance testing till the end of the implementation or avoid the review of the need for performance testing. This test can bring to surface important challenges that can require fundamental architectural fixes, so doing it late in the implementation puts the entire solution at risk of wasting important resources.

It is important to mention that the infrastructure required to execute this test type, especially when the plan requires a higher spec environment, does not need to be available during all the implementation. You bring them a needed environment or repurpose ones for other test types during the lifetime of the project. This does not mean you should not log performance bugs observed during other test types at any time.

User acceptance testing (UAT) can be a great opportunity to start building automated testing for recording the tests so you can have repeatability on testing. This sets the path for regression testing, optimizing the investments to build that automation.

UAT is the last comprehensive test before going live, and every test case emulating real operation is executed so end users can record their test as a proof of the test case validation, therefore making this the first step toward automation.

Remember, you are not just testing the standard product in this test type; you are testing the solution that you are implementing in combination with your business operation needs. Some very basic implementations can justify not executing a performance test, but before deciding you can learn more about what you need to have a performing solution in Chapter 17, “A performing solution, beyond infrastructure.”

User acceptance tests

We are now getting ready for prime time. Our solution works in unison, it performs, and it is ready for final validation by the actual people that execute the business, our end users. User acceptance tests are the final calls for the business to declare readiness for go live.

This test is manually executed—never automated. It is executed with customer data, including migrated data, and with the latest solution version. This test is the closest to being like running live operations, it is an actual business operation simulation.

The objective is obtaining business sign-off of the solution, collecting end user feedback, helping to manage organizational change, and it is done in a dedicated and integrated test environment.

UAT is done during the Prepare phase of the Success by Design framework and it happens prior to preparing for go live, which is a prerequisite to reach this milestone. For more details, see Chapter 18, "Prepare for go live."

See it in action

The business team now brings in a select group of people who run the operation. This group consists of order processors, the accounting team, the accounts receivable team, shippers, and others. The selected group of people run a real-life operation simulation. All the processes are executed in parallel and in-sync between teams. For the prospect to cash process, they involve a selection of different end users connected to this process to run the test. The team tests all the variations of the processes in scope.

This test type needs to be executed by users from the business, the implementation team is just a facilitator. The business team is the main owner of this test, failing to fully participate or test thoroughly is a risk.

The business users must be trained prior to UAT, not just on the solution but also on how the pending process works once the solution is deployed to production; otherwise this group causes false error reports because the lack of training.

At the end of the successful test, the end user connects the new solution to its reality of their daily tasks and confirms the readiness of the solution, but also their own readiness at being familiar with the new system after being trained. The new solution fulfills the business operation need.

You are required to plan for the UAT milestone and to have business stakeholders sign off on the acceptance of the complete, satisfactory results of this test prior to final preparations to operate (go live).

The final iteration of this test type requires the business sign off on acceptance. We describe the importance of this step in later sections in this chapter.

This is one of the most important milestones in terms of testing types. This test happens at the end of the implementation and prior to the go live. It is the entry criteria to prepare for operations. UAT is an end-to-end testing approach where all the processes in scope are tested and it requires sign off on the acceptance by the business stakeholders.

Regression tests

Now we need to look at how to scale our future testing cycles, but also to make sure prior positive test results continue as you keep evolving the build for the solution.

Regression testing is testing that requires you to repeat previously executed tests, due to changes or updates, to the solution to ensure it stays healthy. Keep in mind that the solution is dynamic, new features are introduced or new processes are required. Running regression tests manually works of course, but it is also important that you consider the need to prepare to automate this type of testing as soon as possible. It is an investment that pays off with time.

The objective is to ensure the solution still performs as expected after a change and that it does so in a pre-production environment, test environment, or development environment.

A regression test happens when you have change in the code, or any configuration or new solution pattern that can impact different processes. This test type is done by testers, developers, and end users.

It is important to perform this test type before change is introduced to the production environment.

As we realize that this necessary quality gate is needed, there are tools available to help you to automate. You can find links to more information about these tools at the end of the chapter in the “References” section.

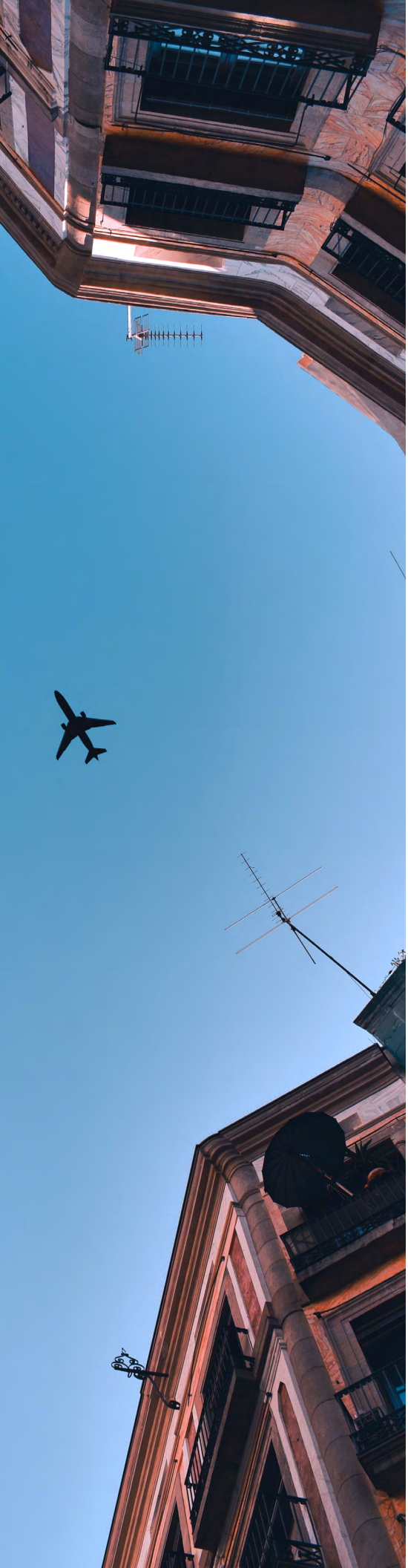
There are different techniques you can follow to run your regression test.

You can opt to test almost 100 percent of the processes, which can provide you comprehensive coverage, but it is expensive to run and maintain especially if there is no automation.

You can prioritize the test based on business impact. This technique ensures you have coverage of the processes that are mission critical for the operation, it is also a more affordable approach and the only risk is that you cannot guarantee a solution 100 percent free of regression.

See it in action

Microsoft has released new functionality that enriches the order processing feature in Dynamics 365 Finance and Operations. When the new update becomes available, teams can execute regression testing to key business processes connected to this new feature. Testing the order and warehouse processes following a new configuration change is done because it can impact the picking process. Once the update is done in the test environment, the team runs an automated test to confirm the solution produces the expected outcomes.



Another technique is to focus on the areas that you expect to be impacted based on the change. This technique is targeted where the change is happening and requires less effort, but it has the limitation of not being able to confirm that you are free of regressions and the impact cannot be visible on the direct change itself, but instead downstream on other processes.

You can also combine all the previous techniques, making sure you test critical business processes, and you can target more testing on the features being changed.

Always keep in mind that bugs discovered late in the implementation process have a lifecycle to get fixed but also require testing. Testing allows you to detect those bugs early, and if the bugs are detected late in the process, let's say in the last test cycle, to think twice on the value of the fix versus the risk it introduces with not being able to do proper regression testing. The fix can be more costly at that point, being so close to go live, than waiting and doing a proper regression testing or delaying the go live.

Every time you test successfully, you put money in a trust. Every time you have a failed test or fail to properly test, you deduct money from that trust leaving a debt in the solution completeness. At the end, you need to reach the end goal with the trust full of money.

Again, automation is key, and you should plan for testing automation. Your solution is alive and ever evolving so change is a constant, not just coming from the application, but from the customer business needs.

Start building your automated testing progressively, focusing first on key business processes then expanding further over time. Do it as early as possible during the implementation process and always test it after a change and prior to production deployment.

Regression is important but it can be costly, especially if you do not automate in the long term. Automation brings the benefit of testing faster, having better testing coverage, and providing repeatability of the test. Automation takes as much planning as was needed for the manual testing planning, but it is a long-term investment that pays itself off with time.

Finally, keep in mind that finding bugs during regression testing could be due the change of the solution design, a standard product update, or just the resolution of previous bugs redefining how the solution works, which can require recreating the test case.

Mock cutover

This is a special test since it is executed in a special environment, the production environment. This type of test is especially important when you test Dynamics 365 for Finance and Operations apps.

This is a temporary use of the production environment to do testing activities and it happens during the Cutover phase. To learn more about the cutover activities refer to Chapter 18, "Prepare for go live."

This test brings important value because it helps to validate aspects like connectivity, stability of the access by users and data, integration end points configuration, device connectivity, security, network, and many of the test cases in your processes may be environment dependent.

During this test, you are able to validate confirmation of the estimated times between environment preparation for production for all the planned tasks during go live. It is a confirmation that all the planned activities run smoothly once you do the actual final execution.

Once the testing is completed you rollback your data up to the point that there are not transactions being created because of the testing, allowing you to finish the environment preparation and start running your system for real. If the mock cutover fails, there is a risk of delaying the go live.

We recommend you always plan for a mock cutover test type.

Other types of testing

We have described the incremental approach for testing through the different test types, scale needs, and continuous validation of the solution. But there are other types of tests that can be unique for every implementation based on the design of the solution.

See it in action

Continuing with the previous examples in this chapter, the team has tested the entire solution and they are ready to move forward. The team wants to confirm the cutover plan in terms of sequence of activities, timing, and production environment stability. They execute the cutover plan by rehearsing and documenting so they can adjust it. After executing the mock cutover, they found the time to load data and sequence it required adjustment due to other conflicting cutover tasks. The team adjusted that sequence and confirmed the readiness to execute the updated Cutover plan knowing it works without surprises.

These test types are below.

- **Smoke tests** Where you test if basic functions work as expected so you can confirm the component can continue to test further with other test types.
- **Data acceptance tests** You validate if the migrated data goes into the target environment correctly and if it is usable for actual operation.
- **Interface tests** Here you confirm if your interfaces can be utilized as intended.
- **Environment testing** In this test type, you validate if the new environment is fit for the purpose.
- **Reporting testing and business intelligence testing** Validates that reporting and Business Intelligence can be executed and operate properly with the expected outcome and performance.
- **Device testing** Focused on validating devices are operational, connect, and perform as expected, for example, RFID readers, scales, and warehouse devices, etc.
- **Network/infrastructure testing** Validation of relevant underlying networking, firewalls configuration, Wi-Fi, printers, etc.

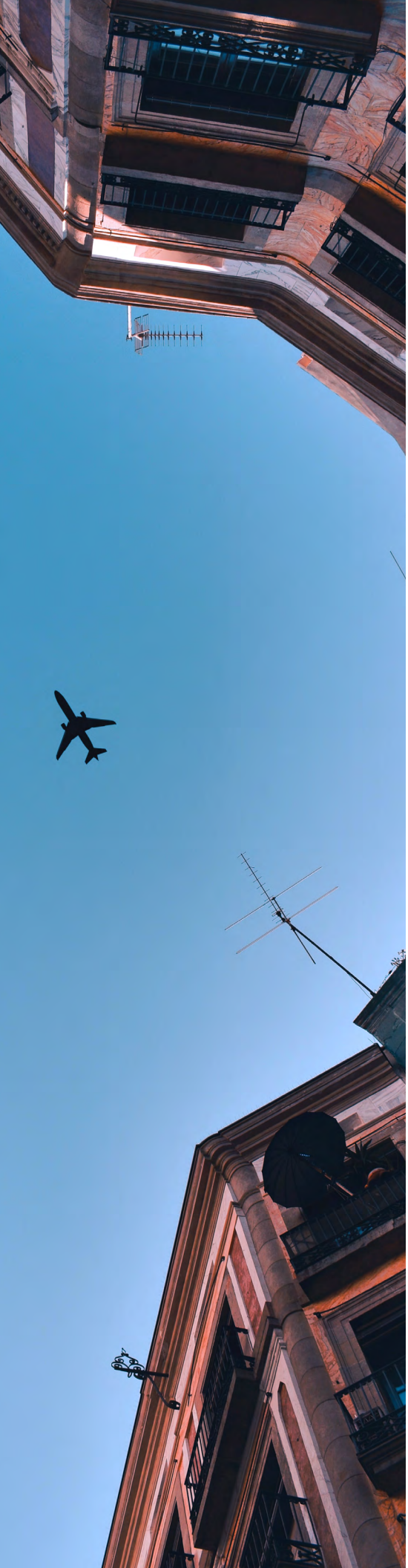
While some projects are too small to justify separate planning to execute some of these types of testing, the concept and meaning of the tests should be folded into other types of testing.

Test types and outcome ownership

When you define your test plan, define ownership for the type of fixes but also who is accountable to drive the fix overall by the test type.

We described the importance of defining the owner of the outcome at testing, now we need to connect this ownership role to the test type.

In other words, who takes accountability to drive the resolution of the outcome derived from testing based on the test type. It is important we have clear ownership to avoid defects bouncing around with unclear roles driving the actions, even though the fix can come from different roles. As an example, unit testing outcomes tend to be owned by partner technical architects or dev leads. User acceptance testing outcomes tend to be owned by customer steering groups devolved to lead SME or PM.



Test types and test plans

Test plans are living documents that keep evolving since project needs can change in terms of adding new test cycles or test types. This is due to business needs changing or the need to increase the quality due to unknown factors at the start of the project. For the most important test types, like user acceptance testing or performance testing, you need to create specific test plans or keep enhancing your master test plan while always bringing enough details to provide clarity for each of them.

The bottom line on defining a test strategy

Defining a test strategy for your implementation is a combination of considering the project scope, testing planning with a test plan and test cycle schedule in combination with the phases and solution versions availability, and selecting all the test types and how you execute them.

Testing is iterative and incremental; it grows as you progress in the implementation and it is a permanent activity once you are live based on the regression testing technique you select.

Always test under the umbrella of the processes, in the end the processes are the language of the business and the testing is proof that the solution "can speak" that language.

Executing testing

In a previous section, we focused on the importance of defining a testing strategy. Now we are going to explore the minimum components you need to keep in mind to execute the testing.

The prerequisites required to move to execution are having a test plan, test cases, and clear testing cycles that are defined according to the progressive readiness of the solution.

Part of a successful test cycle is to set the expectations with the participants, so everyone keeps the focus on the objective. Consider the test cycle like a mini go live for the scope in place for the cycle.

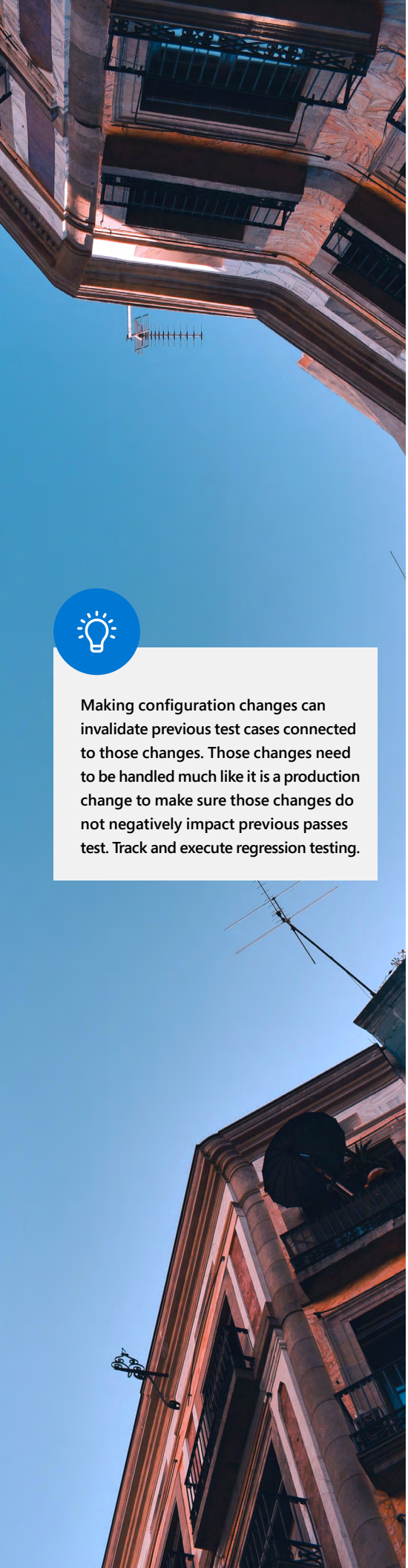
Now we focus on the tactical components for prepping the execution, what to keep in mind to communicate the scope, and how to track the progress and outcome.

Communicating the plan for the test cycle

Aligning the teams prior to testing execution is important so that everybody stays focused on the expected outcome of the test cycle being executed. This is especially important when you test with several people involved, like process testing or functional testing.

During the communication effort you share the test plan for the test cycle. You need to communicate the following with the team during testing.

- **Scope** Before you start testing, you describe the scope and purpose of the testing cycle, and what processes, requirements, and tests cases are included in that test cycle. Every new testing cycle requires detail on how the iteration has been designed in terms of incremental testing and what is expected to be tested. The scope of the test cycle is aligned to the solution version being used.
- **Schedule** The time expected to be used to run the test cycle.
- **Roles** Who is testing and how are the test cases distributed? How do they report test case execution so dependent teams can continue the testing? Who is the orchestrator of the test? Who resolves questions to test cases per area?
- **Resolution process** One of the objectives to test is to identify any defect in the solution. The communication plan needs to specify how those bugs are reported but also how to document the feedback from the tester.
- **Progress** How is the progress recorded and communicated so everybody can see the current state of the testing cycle?
- **Resources** The communication needs to specify where the testing happens and how to access the apps. It determines any additional equipment needed for testing, for example printers, bar scanners, network requirement, etc.
- **Test sequence** Especially on process test, end-to-end test, and user acceptance test types. You need define and align how the different teams interacts.
- **Test objectives** Here you explain the purpose of the testing cycle.



Tracking during testing

Detecting bugs and properly documenting them is key. It facilitates the resolution. But before reporting bugs, the tester needs to understand the scope and objective of what the test is. Building the solution is progressive and it is easy to report issues where the scope of the test does not include a solution for that and is part of future iterations. Keep the focus with the team testing.

Tracking captures the outcome and performance of the test cycle; it is how the development team and consultants understand the quality of the test cycle. Azure DevOps is a recommended tool for this tracking since it integrates into the entire application lifecycle management and any bug reported is visible to the team that needs to fix it, but also shows what is or is not working in general.

Let us explore some tactical components of the tracking of issues discovered during testing.

Making configuration changes can invalidate previous test cases connected to those changes. Those changes need to be handled much like it is a production change to make sure those changes do not negatively impact previous passes test. Track and execute regression testing.

Keep track of changes in configuration

It is likely some tests result in tuning configurations as you test. Adjustments to the solution in configurations can change the behavior of the app, and those changes can happen during tests. This brings flexibility into the resolution, but also raises the risk of the team not documenting those changes. Keep in mind that for the next test cycle the data can be reset and if you do not document and make sure those changes are not reflected in your golden configurations, you lose the important opportunity of what you fixed during testing. In the end, you do not want to repeat the same research over and over, so document the configuration update.

Keep record of the progress

At every test cycle you come with a scope. Depending on the type of test you have some dependencies. Teams need to know when they should start testing each type of testing. A testing heatmap can help you to show the progress of testing by process, test cases, and its statuses. For example, in the scenario used to explain the test types in previous sections we used the analogy of testing a prospect to cash.

A meaningful process test type makes sure the process being tested has continuity, from the order to the invoice and shipment and later to the collection. If you are testing a sequence of processes, you do not go ahead of the game and straight to the collections part. If you do, you lose the opportunity to properly test the larger process sequence.

Keep record of the bugs

You know how to report the bugs and that the reports need to be accurate. Make sure you can report the issue, but also that the case being tested is in scope for that test cycle. Not doing that creates rework in the investigation for resolution. Also, be factual. Is the test passing or not based on the test case documentation and the expected outcome when following the exact steps?

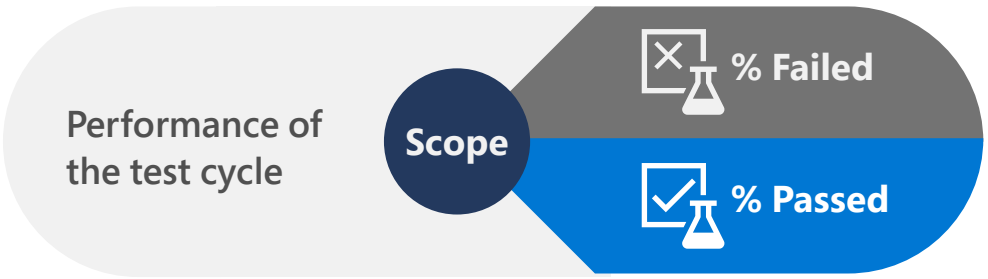
Bugs versus gaps

One common issue during bug reporting is that we can mix a bug with a required feature or gap. Stick to the test case, if you identify a gap, great! Just make sure the feedback is properly classified. We do not want to lose the opportunity to capture that great idea or finding, we just need to funnel it properly to the right tracking system.

Dealing with the outcomes

Finishing the test cycle requires reporting the results and maintaining the bugs discovered during the testing, but also funneling the feedback received. In the end, all these outcomes add onto the top of the next wave of processes or features to be introduced in future test cycles, as shown in **Figure 14-7**.

Fig. 14-7



Reporting the outcome measures the performance of the test cycle and helps to identify areas of opportunity for the implementation team since it triggers clear actions to improve the quality, so the next test cycle provides better performance.

For the next test cycle, we start over, as even passing test cases during this cycle need to be re-tested in the next one.

Solution acceptance and operation

Finally, the last test cycle takes the main objective of why you started implementing the solution to confirm the readiness to run the business using the new solution.

We described the importance of reporting the outcome of the test cycles for user acceptance test types, this report helps to determine the readiness to prepare for go live and is the foundation of the go or no-go decision by the stakeholders. The performance should be high at this point, and if not? Then the solution is not ready.

This confirms the readiness of the solution but also the readiness of the production environment if you run a mock cutover test. It is important that for any final test the business team signs off and confirms they can operate the business with the new solution.

Accepting the solution does not mean it is 100 percent free of bugs. You need to assess the value to bring a fix, at this point if the issue is very low risk for the operation it is often better to go live with known nonblocking bugs and fix them later than to introduce risk by making an unnecessary rushed fix and not having time to re-test properly.

From here you move to the maintenance mode of the solution if you are implementing only one phase, or you continue adding new workloads or expanding the solution. Keep the discipline for testing and scale using the tools for automation. Testing is a continuous practice, the difference is the frequency, scope, and tools used to test.

Sign off by the business team on the overall pass of the final test cycle is a required step prior to executing cutover preparation for go live. This creates accountability on the acceptance of the solution by business. Any non-blocker bug needs to be document and they should be low risk for the operation at not bringing a solution at going live without them.

Conclusion

Testing is about quality, it is used to confirm the effectiveness of the path taken by the implementation team to solve the challenge of how to operate the business with the new solution, it confirms the initial vision by putting it into action. It is a required step in the process to implement Dynamics 365 apps, regardless of how simple or complex your solution is, or if you are implementing standard functionality or extending or customizing it. If you run into a large number of quality related issues after you are live, it is generally attributed to ineffective testing.

Testing during implementation is the step that builds the trust for the business to run their operation. Always test and do so as early as possible. It is never too early to start testing.

References

Finance and Operations

Test Strategy TechTalk

Performance benchmark

Performance patterns and antipatterns

Performance troubleshooting tools

Performance testing approach

Regression suite automation tool

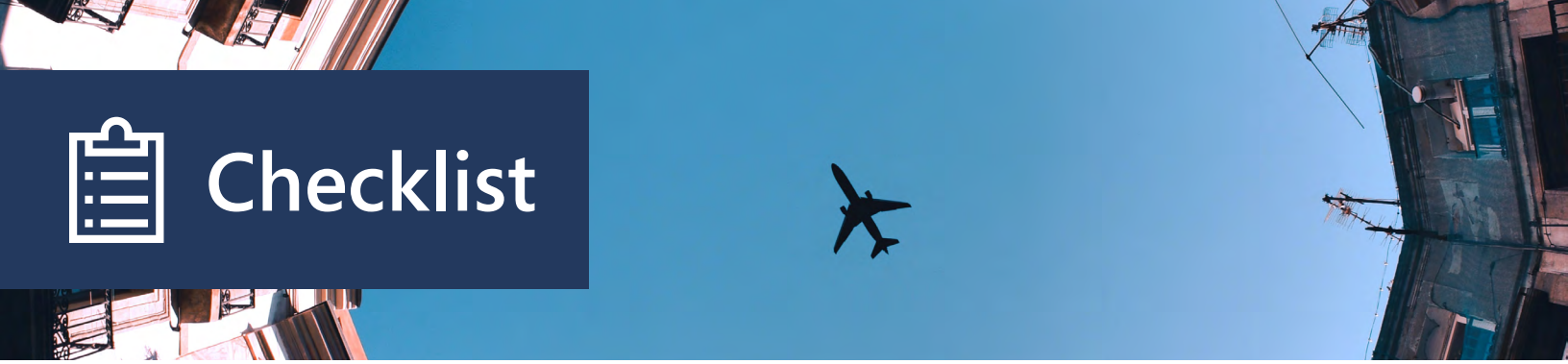
Customer Engagement

FastTrack automated testing in a day workshop offering

EasyRepro – Getting started

EasyRepro in 60 Seconds

Automated and manual testing with Azure Test plan in Azure DevOps



Checklist

Planning

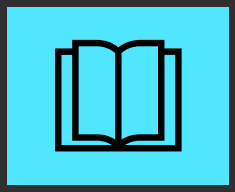
- [ ] Plan the test case creation, forecast the resource requirements (people, hardware, environments, devices), and implement the test cycles in accordance with the processes in scope.
- [ ] Have a tracking mechanism that monitors the overall progress, pass/fail rates, ownership, and issues tied to specific test cases, and have a process defined for remediation, retesting, and deployment when errors are found and corrected.

Implementation

- [ ] Ensure the SDLC includes unit testing by the developer in a development environment, with focus on function, code, and configuration of all extensions and integrations.
- [ ] Ensure the functional testing is performed by functional consultants or SMEs in a test or developer environment with the primary objective to validate the design against the requirements.
- [ ] Ensure process testing is performed by SMEs in a test environment, running multiple processes consecutively, focusing on testing whether the solution allows business operations and monitoring for unintended outcomes.
- [ ] Carry out performance testing throughout the Implement and Prepare phases on critical processes to ensure the solution performs while focusing on scaling with realistic load and growth.
- [ ] Throughout the Implement and Prepare phases, perform iterative and cumulative end-to-end testing of all full business processes in scope and connected integrations and solutions in a test environment with increasingly “real” migrated data.
- [ ] Perform UAT in a test environment during the Prepare phase to test the preparedness of users, processes, and the solution to operate the business.
- [ ] Consider automation for functional, process, end-to-end, performance, and regression testing.
- [ ] Establish a test communication plan with clear scope and objectives, specifying the schedule, sequence, roles involved, and issue resolution process.
- [ ] Perform regression testing in a test environment, or development environment for updates, throughout the Implement and Prepare phases as well as post-go live to ensure the solution performs as expected after a change.

Go live

- [ ] Plan a mock cutover in the new production environment to validate confirmation of the estimated times between environment preparation for production for all the required tasks during go live.



Case study

The right environment for the right test: A missed opportunity

A customer in the manufacturing industry that manufactures recreational vehicles is implementing Dynamics 365 for Finance and Operations and has created a testing strategy that included four testing cycles, including user acceptance testing, as part of their plan.

The team included the most common test types except for performance testing under the assumption that the first rollout would not require it since the extension of the system was low, and the customer will be implementing just manufacturing business unit. Next rollouts will include the servicing business, including the warranty management of recreational vehicles plus other operations. The servicing operations will continue to be executed as a third-party solution for now and their dealership network will be using this system to order parts and honor warranties.

Very soon, during the first testing cycle, the team started to test across the first wave of the solution where there were no integrations, so completing testing was satisfactory in terms of the cycle performance.

As the team got ready for their second wave of testing, they started to introduce some of the integrations as part of the scope for the test cycle, but those integrations were emulated with dummy data and the team introduced some migrated data. Integrations worked for their purpose and the volume of migrated data was small.

For their third testing cycle, things looked great and they were ready to do a more comprehensive testing by using end-to-end testing, this time with real integrations and real volumes of data migration.

At combining the testing for having a larger volume of users for the test and integrations running with similar volume of operations to what they expected in production, the team hit the first major blocker. The solution was not keeping up with the needs of the test cycle, and they were not even in production. The first reaction from the team was an underperforming infrastructure, which raised the concerns of the business stakeholders upon learning the test cycle outcome.

The team was ready to prepare for UAT and decided to continue expecting that this would not be an issue in production due to it having higher specs. They assumed the production environment would be able to solve this performance challenge, so they decided to continue, complete UAT, and move to production. The customer signed off and preparation moved to the next stage to get ready for go live.

The big day came and production and all the departments switched to new system. The first day was normal and everything was working great. The team decided to turn on integrations for the servicing solution on the second day. When the second day came, they were ready to go interconnect with the service department, and integrations started to flow into Dynamics 365. This is when they had their first business stopper: they had a sudden decrease in performance, users were not able to transact in Dynamics 365, service departments from the dealerships were not able to connect effectively, shipments started to slowdown, and the shop floor was having a hard time to trying to keep inventory moving to production.

The implementation team was providing hyper-focused care and immediately started to check for what could be blocking the operation and they found the problem. The integration was creating a bottleneck by making important OData calls to check inventory, creating sales orders, and invoicing while connected to the dealership's partners using the servicing solution. They stopped the integration and things went back to normal, but it was clear that the integration was the cause.

The team found a solution after troubleshooting, but it will require a change to the integration design pattern and fast movement to solve it and keep operating. Part of the revenue from the recreational vehicle business comes from servicing and providing parts to the dealerships. It took days of manual work to keep things working during this painful first week of go live, which impacted the perception of the organization in the solution.

In retrospect, validating designs for integrations with a volume of data was key to preventing this scenario and the issue they had during real operation. The team's morale was affected after working hard during the implementation season, all because they missed one important test—the performance test. The team thought emulating early and validating in production would be OK, but the cost of that decision was high.

This team learned that including performance testing as early as possible, with actual volumes in the right environments that perform at the same volumes expected in production, would help to detect a design problem and correct it in time. The team also learned that "testing" in production for the first time, and with the real volumes, was not the best idea since the time available to make any fixes was reduced due to the live operations and correcting as they go introduced bigger risks. The assumption of the team that they were implementing almost standard functionality led them to think that the solution would perform fine, but the actual customer challenge made the solution unique which requires extensive testing.

For the next phases, this customer included performance testing. They dedicated an environment to stress test the solution with their own business needs and they executed these tests earlier with additional scenarios and UAT that included parallel processing of people testing and integrations running. They were able to have second go live to include their accessories manufacturing business and it was a breeze in comparison.

It is never too early to start testing and do so according to the project needs. The later the testing, the bigger the risk.

15

Guide

Extend your solution

Introduction

Business solutions offer a rich set of capabilities to help drive business value.

Still, in some situations, you need to extend the solution and adjust off-the-shelf functionality to accommodate organization or industry specific business processes. These adjustments can change how a feature works or bring additional capabilities to meet specific requirements.

While business solutions natively offer rich capabilities, they also offer powerful options to customize and extend them. Extending the solution can open even more opportunities to drive business value. It is important to note, however, that extending should not compromise the fundamental advantages of an evergreen cloud solution, such as usability, accessibility, performance, security, and continuous updates. These are key to success and adoption.

Complex business requirements lead to highly advanced solutions with customizations and extensions to applications.

Advanced implementations bring an increased risk that user experience suffers because of the introduction of performance, stability, maintainability, supportability, and other issues.

In this chapter, we delve into key factors to consider when determining the scope of depth of solutions.

Defining your extensibility strategy

When organizations shift toward software as a service (SaaS) cloud solutions, one of the main reasons they do so is to enjoy the many

advantages, including out-of-the-box security, native integrations with other cloud solutions, cost savings, and improved agility. The evergreen cloud approach also enables the continued ability to evolve with the latest modern features. In addition, it is often easier to use off-the-shelf solutions with embedded modern processes and further extend it. This tailors a solution to meet business needs, rather than having to build a custom solution from scratch with all the associated hassles and expense to meet the precise requirements of your organization.

What is extending?

When organizations implement a solution, there typically is some degree of customization and/or extensibility, which we refer to as extending. Extending can vary from minor changes to the user interface of a particular feature to more complex scenarios like adding heavy calculations after certain events. The depth of these extensions has important implications on how much the off-the-shelf product needs to change to meet specific business requirements.

We can also categorize these extensions into distinct types:

- Extending by modifying off-the-shelf functionalities.
- Extending by introducing new functionalities on top of a native one.
- Creating custom solutions on the platform.

Determine the need

It is important for organizations to be aware of the impact of extending the solution. They want to strike the right balance between the features required to empower a user or bring business value, as well as efficiency and the value it brings compared to what it costs.

A key driver in the decision to extend is understanding the potential risks of extension to key solution characteristics, like its robustness, reliability, performance, usability, security, and ability to take updates. Consider the example of extending an invoicing system that could pose risks to the operations or could disrupt the business because of increased complexity; such risks are best avoided. When opting to extend the solution, organizations may also depend on a third-party to maintain and evolve it,

and in some scenarios block the organization from adopting new functionalities.

Extending to integrate with other solutions is a common scenario that we cover in Chapter 16, “Integrate with other solutions.”

Do not replicate your legacy solution

When gathering solution requirements, organizations may want to reproduce every piece of functionality from the legacy solution or even try to mimic the user experience. This could come across as a strategy to reduce user resistance and drive adoption, but it can also lead to a highly customized solution that fails to leverage the strengths of the new platform in favor of familiarity.

Legacy solutions may have taken years to develop and evolve and may not use the latest and greatest functionality available on the market. As an example, Dynamics 365 natively uses artificial intelligence and machine learning to provide insights that help users make the best and most informed decisions.

Major deviations from the native features or repurposing existing features can limit the value of an evergreen platform by compromising flexibility and the ability to evolve with changing business needs. That increases technical debt and maintenance costs.

Understand the power of the platform

Another challenge arises when an organization does not fully know the power of the new solution and opt to extend before even understanding the impact of their decisions. The platform may have numerous ways to achieve the same functionality as the legacy solution already. Reasonable customization of the off-the-shelf solution might be required, but the point is to find the right balance. Thus, each scenario should be evaluated, explored, and in some cases piloted. And of course it’s important to involve key business users in these decisions.

Leveraging ISV solutions

Leveraging independent software vendor (ISV) solutions from the app marketplace instead of extending the solution to achieve the same results may save development cost and time, as well as testing and maintenance resources. ISVs typically support and maintain the solution at scale for multiple organizations. Their experience can be an advantage for organizations that require additional functionalities that are already provided by ISVs.

When deciding to use ISV solutions, organizations should be aware of the ISV support model and its alignment with continuous updates as well as protection clauses.

Extensibility scenarios

Extending off-the-shelf solutions occurs when functionality is changed to fulfill an organization's requirements.

Customizations and extensions performed to out-of-the-box applications could stretch from simple setting and configuration updates to customized platform as a service (PaaS) solutions that extend their functionality. In this section, we explore levels in which customization is possible and discuss their relevance.

Some organizations prefer to stay as standard as possible. That does not mean that they cannot make simple changes with configurations or low-code/no-code customizations using modern tools. Staying standard allows for slight changes, as long as those changes do not highly impact key characteristics of a solution.

App configurations

Configurations are the out-of-the-box controls that allow makers and admins to tailor the app to the needs of a user. These setting changes are low effort, requiring no support from professional developers. They are a powerful way to make the application your own, for example changing a

theme to reflect business branding.

In some cases, additional due diligence might be required for configurations, even though they are tested and shipped as part of the core service. Examples include security model configuration, out-of-the-box integration setup, and enabling embedded intelligence. Also consider new capabilities delivered via the service updates that might be controlled using the app settings.

App settings are the safest and the least impactful way of tailoring the solution to your needs and should be the preferred approach before exploring another extensibility technique.

Low-code/no-code customizations

A differentiator for Dynamics 365 and the latest generation SaaS products is the powerful customization capabilities made available through "what you see is what you get" (WYSIWYG) designers and descriptive expression based languages. This paradigm helps significantly reduce the implementation effort and enables businesses to get more involved with design and configuration.

This low-code/no-code approach also guides makers to take advantage of the optimized patterns and best practices, avoiding the potential pitfalls and antipatterns that are observed in a complex code. Because the platform takes care of lower-level details and automatically optimizes the logic, testing is focused on functional components. The impact on performance, usage pattern and service boundaries, however, still needs to be taken into consideration.

Typically, customizations result in user interface modifications and changes to the field settings, security model, business processes, visualization components, etc. Changes implemented through customizations are performed to ensure data quality, reduce repetitive tasks, and adjust visualization components that determine how information is organized and displayed to the users.

With customizations, applications can be tailored to more closely fit

organization requirements that are specific to an industry or unique business processes, including specific functionality focused on specific roles or personas. This allows personalization that streamlines the user experience so a user can focus on what is most important.

Low-code and no-code customizations are the preferred extensibility approach when the requirements cannot be satisfied by app configuration changes.

Extend using pro dev

Over the years the low-code/no-code base configuration capabilities have advanced, reducing the dependency on professional developers for application changes. Edge cases or scenarios that cannot be achieved using the configuration-based approach can still exist. In that case, the powerful custom code execution capabilities of Dynamics 365 can be leveraged.

Dynamics 365 apps support extension programming models that enable organizations to leverage existing expertise within the organization for advanced scenarios. These models, depending on the specific app and functionality extended, generally supports .NET based languages, Software Development Kits, and open-source libraries around JavaScript and HTML5.

The key aspect to keep in mind when using code-based customization is to understand the extensibility framework of Dynamics 365 and only use the documented extension patterns. Use of unsupported techniques breaches the service level agreements and can have a potentially severe impact on the live service.

It is important that your developer teams have the latest software development toolkits available to them. They should also know about community based third-party tools and samples. These can drive productivity even though Microsoft may not provide direct support for them.

Extending into PaaS

In some scenarios, organizations leverage PaaS components to extend

solutions, which adds powerful capabilities that help address complex requirements. Dynamics 365 has a core design philosophy that allows our SaaS applications to be extended by leveraging the underlying Azure PaaS capabilities. This is referred to as the no-cliffs extension approach. The approach enables businesses to start with SaaS, and then for the most complex long-tail scenarios, extend into the Azure PaaS. Doing so alleviates the fear of being limited by the platform.

This no-cliffs extension provides the best of both worlds. The SaaS application provides the off-the-shelf functionalities as well as the options and methods to extend them. The PaaS extensions further enrich the solution architecture by providing rich and powerful mechanisms that scale and allow heavy processing of operations outside of the business solution.

An example of how this approach is natively used is the Internet of Things (IoT) Intelligence add-in for Microsoft Dynamics 365 Supply Chain Management. This add-in integrates IoT signals with data in Supply Chain Management to produce actionable insights, as illustrated in **Figure 15-1**.

The same happens with Connected Field Service. Connected Field Service for IoT Central provides a direct integration of Dynamics 365 Field Service with Microsoft Azure IoT Central, an add-on solution

Fig. 15-1

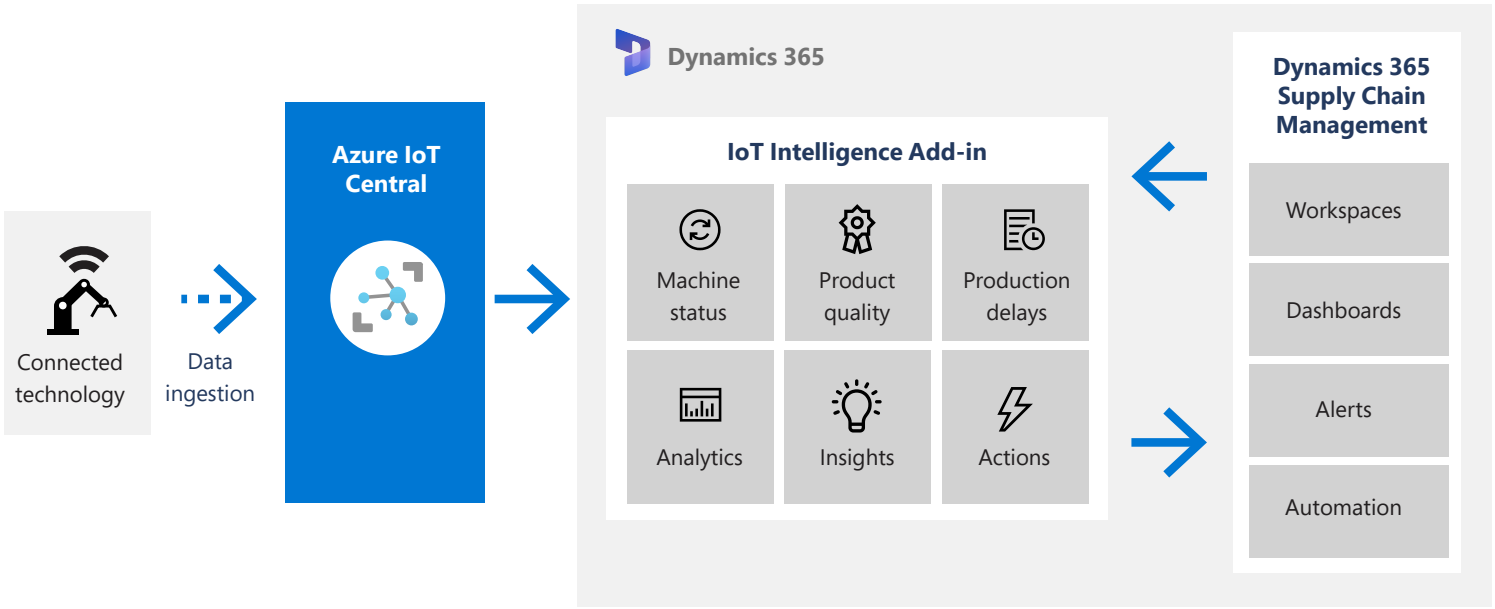


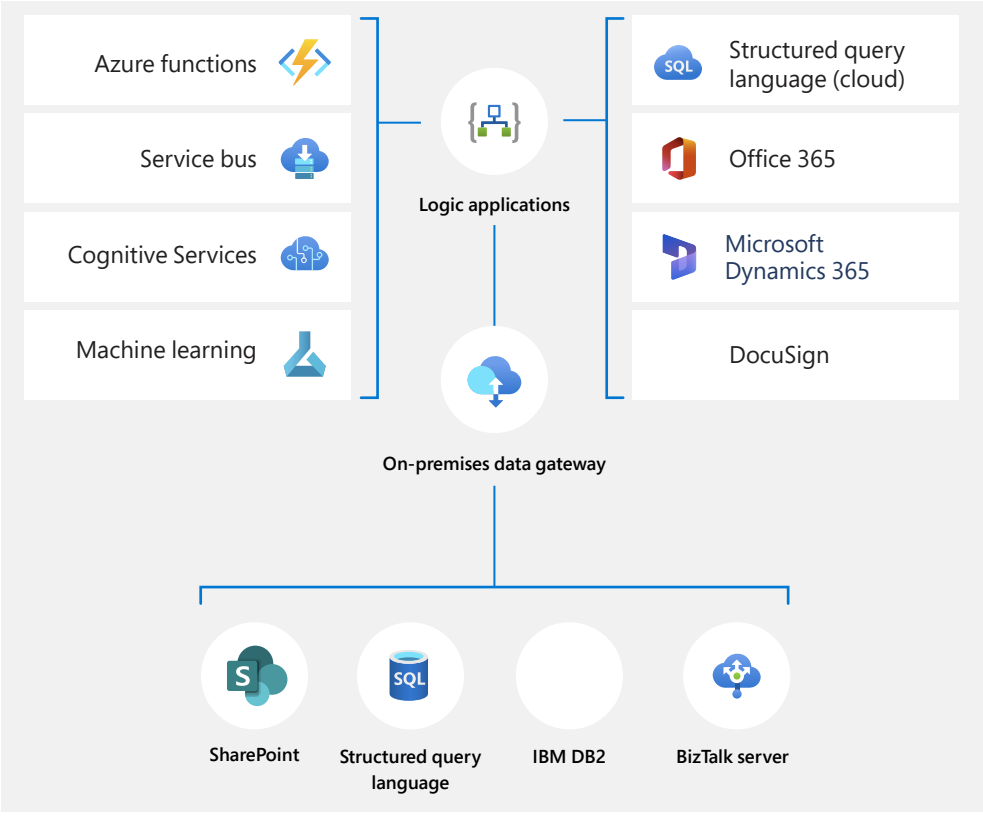
Fig. 15-2

that brings in Azure IoT PaaS on top of Dynamics 365 Field Service, as shown in **Figure 15-2**.

In both examples, organizations can leverage PaaS to further extend the solution. The same can be applied to specific organization requirements that can use PaaS features as part of the extension's architecture. This approach can be particularly valuable because it reduces the maintenance requirements for parts of the solution compared to a fully custom-built extension.

One example of this approach is when organizations leverage Azure Logic Apps. Logic Apps provide a serverless engine to build automated workflows to integrate apps and data between cloud services and on-premises systems. Logic Apps provide the ability to trigger workflows based on events or timers and leverage connectors to integrate applications and facilitate business-to-business (B2B) communication. Logic Apps are integrated seamlessly with Azure Functions, as illustrated in **Figure 15-3**.

Fig. 15-3



Azure Functions is an event-driven serverless compute platform that can also solve complex orchestration problems, by using it you can move some of the heavy computing processes away from Dynamics 365.

With Azure Functions organizations can code their own workflow definition and simplify complex, stateful coordination requirements programmatically in event-driven applications, as well as taking advantage of numerous native connectors.

A multitude of examples demonstrate where PaaS can be leveraged to further extend the capabilities of the solution. Leveraging a platform helps

eliminate the costs and complexity of configuring and managing the required infrastructure, while empowering development teams to focus on building the apps and services that drive value.

Considerations

Every piece of an extension should focus on bringing efficiency or value to the organization. Inherent costs do exist when implementing changes and can vary from the cost of building, testing, maintaining, and support. These should be taken into consideration when planning to extend a solution.

In this section, we delve into the key considerations and impacts that extensions can have on a solution.

Usability and accessibility

User experience is a key characteristic of a business solution. Providers invest heavily in the user experience to ensure that the users have a pleasant experience when interacting with the solution, and that the solution operates seamlessly between devices, such as laptops, tablets, phones, and platforms like Outlook and Microsoft Teams.

We earlier discussed that the purpose of extending off-the-shelf functionalities should be to add further value to the business. One way to achieve this is to improve user experience. This is achieved by considering and adapting user requirements and expectations.

While the main purpose of extending may not be to improve the user experience, it should not negatively impact the user experience and how the solution behaves across different devices and platforms. In addition, extending should not negatively impact the responsiveness and performance of the solution.

Security, privacy, and compliance

Security and compliance are important aspects of a solution. When

extending, it is crucial to honor the security and compliance requirements.

Access to data, either implemented by security roles or any other feature, should not be bypassed when customizing or extending the native functionality. Breaking the security model can cause a large negative impact, and security breaches are not to be taken lightly.

The same happens with compliance, for example the General Data Protection Regulation (GDPR) policies. Compliance functionality implemented natively also need to be inherited in any customizations of extensions. Failing to do so may have consequences for organizations that do not comply with regulations.

GDPR is just one set of regulations. Regulation of privacy and data use exists in many different forms across several markets. While there is a great deal of overlap in terminology, privacy and security are not identical. Security is about preventing unauthorized access to any data, while privacy is ensuring, by design, proper acquisition, use, storage, and deletion of data defined as private under local regional and global regulations.

While security is natively built into Dynamics 365 business applications and highlighted as best practices, privacy requirements tend to have a higher probability of being overlooked by developers when extending native apps.

Performance

Although cloud solutions provide a high degree of scalability and performance, when extending a solution it is important not to compromise performance.

When extending the user interface and/or the business logic, additional efforts are added to create, retrieve, update, or even delete data. Those additional efforts may have an impact on the user experience, depending on the amount of extended functionality added.

Service protection limits also exist to ensure consistent availability and performance. These should also be considered when extending the

solution to avoid random or unexpected surges in request volumes that threaten the solution's availability and performance characteristics.

Performance goals should be identified, agreed upon, signed-off on by the business, and measured as part of the testing strategy to identify any deviations from those goals. These goals are especially important when the solution is extended, as those extensions can highly impact them.

Depending on the environment strategy, organizations may have globally distributed solutions where factors like latency, firewalls, network traffic, organization proxies, and routing by Internet Service Providers (ISPs) are important to ensure a good user experience. When extending a solution, especially integrations with external systems, the number of external calls, round trips, and volumes, on top of all previously mentioned factors, are crucial to ensure that the solution meets the performance goals.

Scalability

The scalability of a business solution is also a key consideration to determine how you extend it. While the cloud platform includes scalable servers and micro services, other aspects of the platform need to be considered to determine the impact of your business solution architecture.

Parameters, such as the number of parallel connections, requests, and concurrent users, can determine how well the business solution scales, for example when new business units, countries, or regions are added.

It is better to assess the impact of the chosen architecture as use of the solution grows, for example, when the scope is small or when the solution is first released.

When extending a feature, it is also important to understand how the functionality applies to different business units, countries, or regions, and what is the level of adjustments needed for each of them. If each business unit, country, or region requires its own level of customization, the solution at the extension level is not scalable by itself and needs further adjustments.

Service protection and limits ensure consistent solution availability and performance, as well as a level of protection from random and unexpected surges in request volumes that threaten the availability and performance characteristics of the platform or solution.

When extending the solution, it is important to understand how use grows over time and the impact of the design on parameters like:

- How much storage is required by the extensions and how does it grow over time?
- How many additional application programming interface (API) calls do the features require?
- What are the limits on workflows, code execution, or API calls?

Impact on application lifecycle management

When extending the business solution, all code and customization must be packaged and shipped from their various environments. This means extensions need to be included in application lifecycle management (ALM). Typically, custom code, customizations, and development created from the user interface or pro dev tools need to be developed, tested, approved, and released into production. In addition, for easier maintenance, they also should be added to a repository where they can be backed-up, merged, and properly handled.

This means that all extended functionalities are added on top of ALM practices, which also increases the complexity. As an example, if the extended features required different configurations, or are only applicable to specific business units, countries or regions, there needs to be a separation at the ALM process, which may just be that they are shipped in different packages (or solutions).

Aspects of the packages should be considered, depending on the number of extended functionalities added to the native solution. Examples include the order these functionalities are introduced, how they are split, and how the size affects the ALM operations.

Maintainability, supportability, and future-proofing

In a cloud world where updates are continuous, and new features are released with frequency, it is important to think about the impact and costs of maintaining extended functionalities.

When extending a solution, additional maintenance requirements are added to the business solution, so it is important to understand and be aware of deprecations and roadmap. This helps avoid building something that might soon be offered natively by the solution or being forced to replace parts of the extended solution because of deprecated functionality.

Supportability

Extending a business solution can also complicate the support requirements of the solution. Normally, the first line of support is at the organization itself or a vendor. Support resources must have expertise on the business solution and extensions built on top of off-the-shelf capabilities. This requires a specialization of resources to support the extended business solution.

The solution provider is typically the last line of support, so it is important to use proper techniques when extending a solution. Not complying with these rules can void the warranty or support terms of your contract.

Because some apps provide the ability to customize them to meet user needs, these customizations can also impact the time required to maintain them. For example, when a new user is onboarded, the time it takes to onboard and whether existing security roles and apps can simply be applied or if additional requirements must be added that require more maintenance time.

As the solution is extended, the importance of having a good testing strategy grows because it is necessary to validate how the solution behaves with each update.

Update and future-proofing

In a cloud solution, organizations must embrace the change and

innovation that comes with frequent new features and capabilities. These capabilities, in turn, add value to the business.

When it is necessary to extend functionality, it should always be done in a way that makes it easy and safe for other developers to extend them down the line.

It is good practice to take the following perspectives into account.

Firstly, having sound developer craftmanship. This applies to basically any language or technology and dictates that when APIs and extension points are created, they are well thought through, robust, and well defined. Also, that the extension is made in a way that allows for other extensions to use the same extension point or API side-by-side with ours.

Secondly, you want to be a "good citizen" in the new solution. That is to say that by your example, you encourage other developers who may expand your code later to follow the same principles. This includes writing code that others can easily read, APIs and extension points that others can extend without surprises, unexpected results, or side effects, and design structures and frameworks that enable others to utilize your effort in a clean and concise way.

Thirdly, it is important to establish a process that anticipates the future and works to minimize potential negative effects extending the solution can bring. This is because, while it is common for organizations to build custom extensions, deviations from native functionalities can have an impact on the ability to take advantage of future capabilities.

Product-specific guidance

In the following sections, we will look at Dynamics 365 Customer Engagement and Finance and Supply Chain Management individually.

Finance and Supply Chain Management

In Dynamics 365 Finance and Supply Chain Management, the

Visit the Extensibility home page for reference information or follow the learning path at Introduction to developing with Finance and Operations apps - Learn | Microsoft Docs.

extensibility model provides the ability to extend the out-of-box functionality to fit additional requirements.

The extension model itself, the toolset, ALM in Azure DevOps, and the SDK is well described in Microsoft Docs, Microsoft Learn, and in multiple community sources. There are two great entry points for learning about extensions and development in Finance and Supply Chain Management apps in the links below and a list of additional relevant links at the end of the chapter.

Introduction

In this section, we first look at what we can do to enhance functionality and UI without extending the application. Then we give a very brief overview of the extension model, and finally highlight some of the tools and practices that are available for professional developers when they extend the application.

Let's first look at what we can extend without changing applications components in **Figure 15-4**.

Fig. 15-4

	Extension example #1	Extension example #2	Extension example #3
	Requirement: Create a custom workspace with tiles and lists from multiple forms and queries across the manufacturing module.	Requirement: Add a table to hold a list of food allergens. Add the allergen field to the sales line record for the user to indicate that a product contains the specific allergen.	Requirement: Add code to automatically change the status of the new production order to "started" when the user firms a planned production order.
	Tools and components: Use the personalization feature and "add to workspace" functionality in the UI to put together the desired workspace and components. No change to code or application components is needed.	Tools and components: Use Visual Studio in a developer environment to add the extended data types, the tables and fields and the extensions to the sales line table and the Sales Order form. This change requires new and changed application components and must follow software development lifecycle (SDLC), build, and deployment guidelines.	Tools and components: Use Visual Studio in a developer environment to extend the X++ business logic, add appropriate event handlers, class, methods, and X++ code to catch the event that the planned order is firmed, and execute the correct code pattern to bring the new production order to started status. This change requires new and changed application components and X++ code, and must follow SDLC, build, and deployment guidelines. Considerations about scalability and performance when large numbers of planned orders are firmed using appropriate developer tools should be considered.
	This requires user level experience with navigation of the relevant module, navigation, and personalization.	Requires entry level professional developer experience in Dynamics 365 Finance and Supply Chain Management and familiarity with Visual Studio, building procedures, and best practices.	Requires medium- to expert-level professional developer experience in Dynamics 365 Finance and Operations and familiarity with Visual Studio, build procedure, best practices, and, ideally, frameworks like SysOperations Framework, Multithreading, and performance-checking tools and patterns.

Personalization and no-code options

As we mentioned above, even though Dynamics 365 Finance and Supply

Chain Management offer a rich extension model with minimal impact on future updates, it is always preferred not to extend the product in the first place if possible. This lowers risk, maintenance, and project complexity.

Before deciding that a certain requirement must be solved through an extension to the standard product, it is a best practice to consider if one of the options for noninvasive personalization or configuration can meet the required functionality. Dynamics 365 Finance and Supply Chain Management offer wide options for personalizing the UI, creating user specific experiences, and automating processes without having to do any programming at all.

Figure 15-5 is a table of these options.

As the table below shows, many options are available to find alternative approaches to extending the application. The below is not an exhaustive showing. Additional options for low code and no code options are mentioned in Chapter 16, "Integrate with other solutions." The decision of whether to extend comes down to user efficiency and value for your customers.

Fig. 15-5

Tools	What is it?	Description	Characteristics
Restricted personalization	The application remembers the last settings for sorting, column width, criterial values in queries and dialogs.	While this is hardly considered a way of extending, it does give the user the opportunity to store selections in dialogs, expansion of relevant fast tables, and aligning the column width so that more columns are visible on a given form.	Personalizations can be shared by users or managed centrally by an admin from the personalization page.
Personalization of forms/UI	Personalization bar in forms and workspaces.	Personalization allows users or admins to add or hide fields or sections, change labels, change the order of columns, and edit the tab order by skipping fields when pressing tab.	Personalizations can be shared by users or managed centrally by an admin from the personalization page.
Saved views	Saved views is a combination of personalization of the UI for a form and filtering and sorting of the form data source.	Saved views is a powerful tool that allows the user the ability to quickly switch between tailored views of columns, filtering and sorting on the same screen depending on the specific task at hand. For example, a buyer in a pharmaceutical company may need a simple view of the purchase order screen for nonregulated materials purchasing regulated materials used for manufacturing	Saved views can be shared by users or managed centrally by an admin from the Personalization page. May require the Saved views feature to be turned on.
Custom Workspaces	Users can use personalization and the "add to workspace" button to add tiles, views, or links to an existing or custom workspace.	This functionality allows users to tailor their experience to their needs. Workspaces provide glanceable information about the most important measures, actionable items, and relevant links to other pages.	Custom and personalized workspaces can be shared by users or managed centrally by an admin from the personalization page.

Tools	What is it?	Description	Characteristics
Custom fields	Users with certain security roles can add up to 20 custom fields to tables	Finance and Supply Chain Management have a rich set of features that apply across a wide set of industries. Some organizations require additional fields on certain tables; for example, the item or customer master or the sales order header. This feature allows the user to create these fields. Note that these fields are specific to the environment that they are created in and cannot be referenced by developer tools.	Custom fields and the personalized forms to show the fields can be shared by users or managed centrally by an admin from the personalization page. Deleting a custom field is irreversible and results in the loss of the data in the custom column.
Grid capabilities	The grid on forms in the system have some extended features that may eliminate the need for an extension.	The grid offers the following capabilities. ▪ **Calculating totals for columns in the grid footer** ▪ **Pasting from Excel** ▪ **Calculating math expressions. For example, if the user enters 3*8 in a numeric field and presses tab, the system calculates and enters the result of 24** ▪ **Grouping tabular data in one or more levels in expandable groups** ▪ **Freeze panes so important information does not scroll out of view when scrolling horizontally**	The admin can enable the New Grid Control feature from feature management. Note that there are certain limitations, see the reference at the bottom of the section.
Embedded canvas apps	The user can add a canvas app to a form or workspace as embedded into the UI or as a menu item that can pull up the app from the power apps menu.	The ability to embed a canvas power app enables citizen developers to use low-code/no-code options for interacting with data in Dataverse directly from the UI in the Finance and Supply Chain Management apps. It is important to note that if the app must interact with Finance and Supply Chain Management data, that integration to Dataverse must be in place and the app must of course support the required actions.	Please see more about UI integration for Finance and Supply Chain Management in Chapter 16, "Integrate with other solutions."
Mobile workspaces	Users can view, edit, and act on business data, even if they have intermittent network connectivity on an app for iPhone and Android.	IT admins can build and publish mobile workspaces that have been tailored to their organization. The app uses existing code assets. IT admins can easily design mobile workspaces by using the point-and-click workspace designer that is included with the web client. The application has 13 predefined workspaces out-of-the-box that cover Finance, AR, AP, and HR functions.	Simple actions can be done from the mobile workspaces. Most more advanced actions require extension.
Excel integration	Users can extract or edit data on most forms in the system by clicking on the office icon.	The Excel integration allows for a wide variety of scenarios for entering, presenting, and interacting with data in way that is not possible from the UI. In addition to the export-to and open-in Excel option, the user can create workbooks and templates for specific purposes. Excel has many features for presentation and offers data manipulation capabilities for larger datasets that users cannot do in the system UI.	With great power comes great responsibility. While it is easy to change a whole column of data and publish that data into the system, it is equally easy to make a mistake.

Extending the application

In this section, we provide a high-level introduction to the extension model and discuss required skill sets and practices used when extending.

We also highlight some of the tools and techniques that we recommend you use to ensure a clean, high-performing, and future-proof solution.

Figure 15-6 is an overview of the components, the consideration for extensions, and the related characteristics.

Fig. 15-6

Component	Editor	Considerations	Do Not
User interface	Graphical editor in Visual Studio with preview.	Forms must adhere to patterns. The editor in Visual Studio has a rich modeler that will help the developer applying the right structure to the screens. This will ensure performance when loading the form, adaptability across different resolutions and screen sizes and consistency across the UI.	Deviate from the predefined patterns and create monster all-in-one screen style forms. They are often a result of the designer trying to replicate a legacy system experience.
Data model and queries	Metadata editor in Visual Studio for Tables, Fields, Extended Data Types, Enums Queries and Views	Follow best practices and frameworks. Apply indexes and define delete actions. Normalize. Use effective date framework when applicable. Keep performance in mind. Use field lists when performance is a risk.	Create redundancy or replicate poor modeling from legacy apps.
Business logic	X++ Editor in Visual Studio.	Adjust compiler setting to alert about best practices, code with the goal of zero deviations. Use code patterns and frameworks. Practice good developer citizenship. Write clean easily readable code. Run CAR report and use compatibility checker. Unit test the code.	Ignore best practices, write long methods or over comment the code.
Reporting	SSRS report editor in Visual Studio	SSRS reports are good for certain precision designs and tabular lists. See Chapter 13, "Business intelligence, reporting, and analytics," for more information.	Do not reach for the SSRS report option if there is a better way.
Data entities	Metadata editor in Visual Studio	The out-of-the-box data entities are general purpose entities are built to support a wide variety of features surrounding business entity. In scenarios where a high volume, low latency interface is required it is recommended to build custom data entities with targeted and specific features needed to support high volume interfaces in the implementation.	Do not create data source proliferation. See Chapter 16, "Integrate with other solutions," for more information.

Development architecture

The architecture of the development environment, as shown in **Figure 15-7**, includes the software development kit (SDK), which consists of Visual Studio development tools and other components. Source control through Azure DevOps allows multi-developer scenarios, where each developer uses a separate development environment. Deployable packages are

compiled and packaged in a build environment or a build service and deployed to Dynamics Lifecycle Services (LCS) for further deployment to nonproduction environments. Deployment to the production environment happens through a service request, after proper testing has been done in the UAT environment and users/stakeholders have signed off as described in the SDLC.

See Chapter 11, "Application lifecycle management," for more information about the SDLC.

Microsoft Power Platform integration with Finance and Supply Chain Management

Finance and Supply Chain Management is a virtual data source in Dataverse that enables full create, read, update, and delete (CRUD) operations from Dataverse and Microsoft Power Platform.

Fig. 15-7

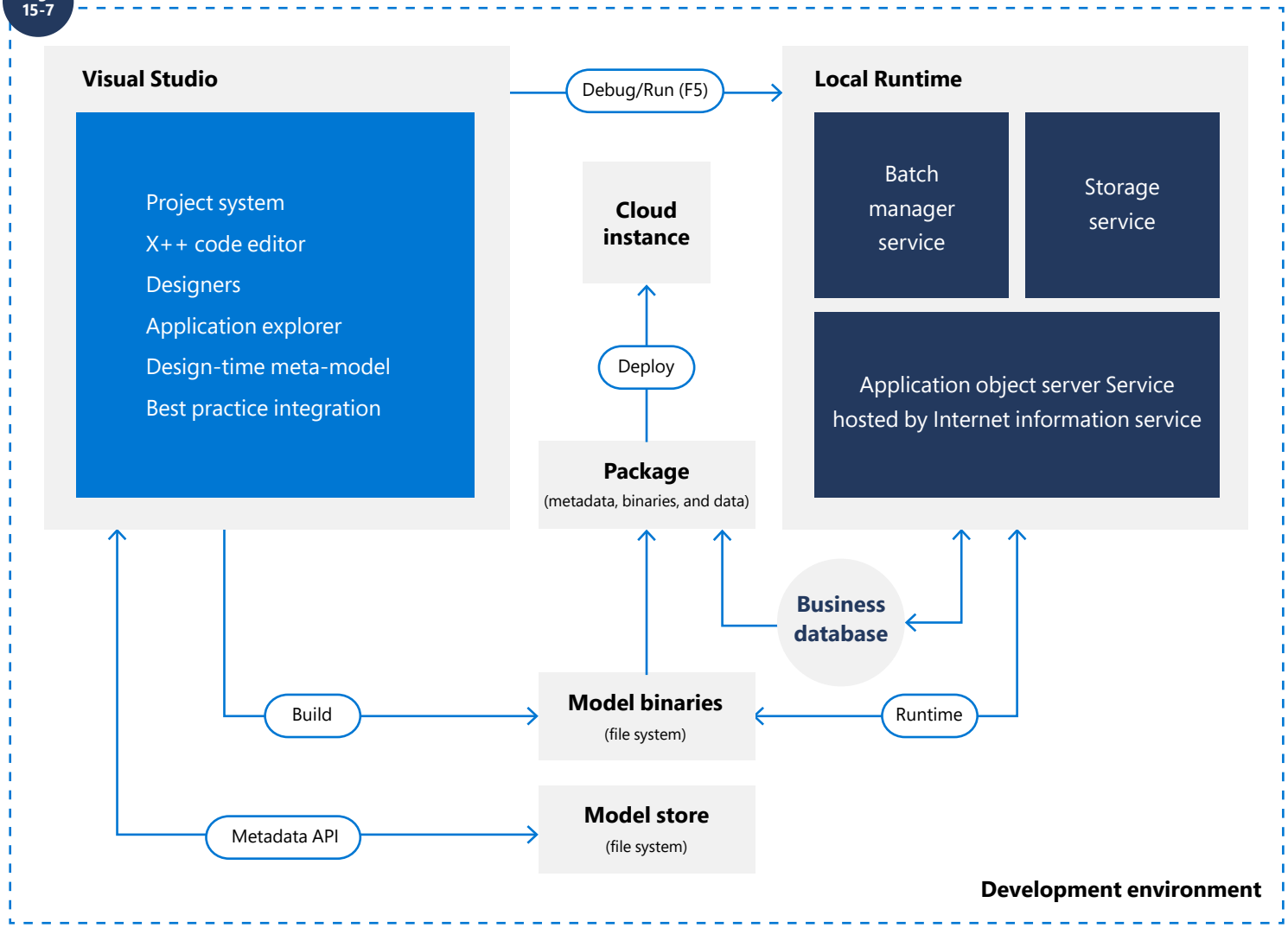
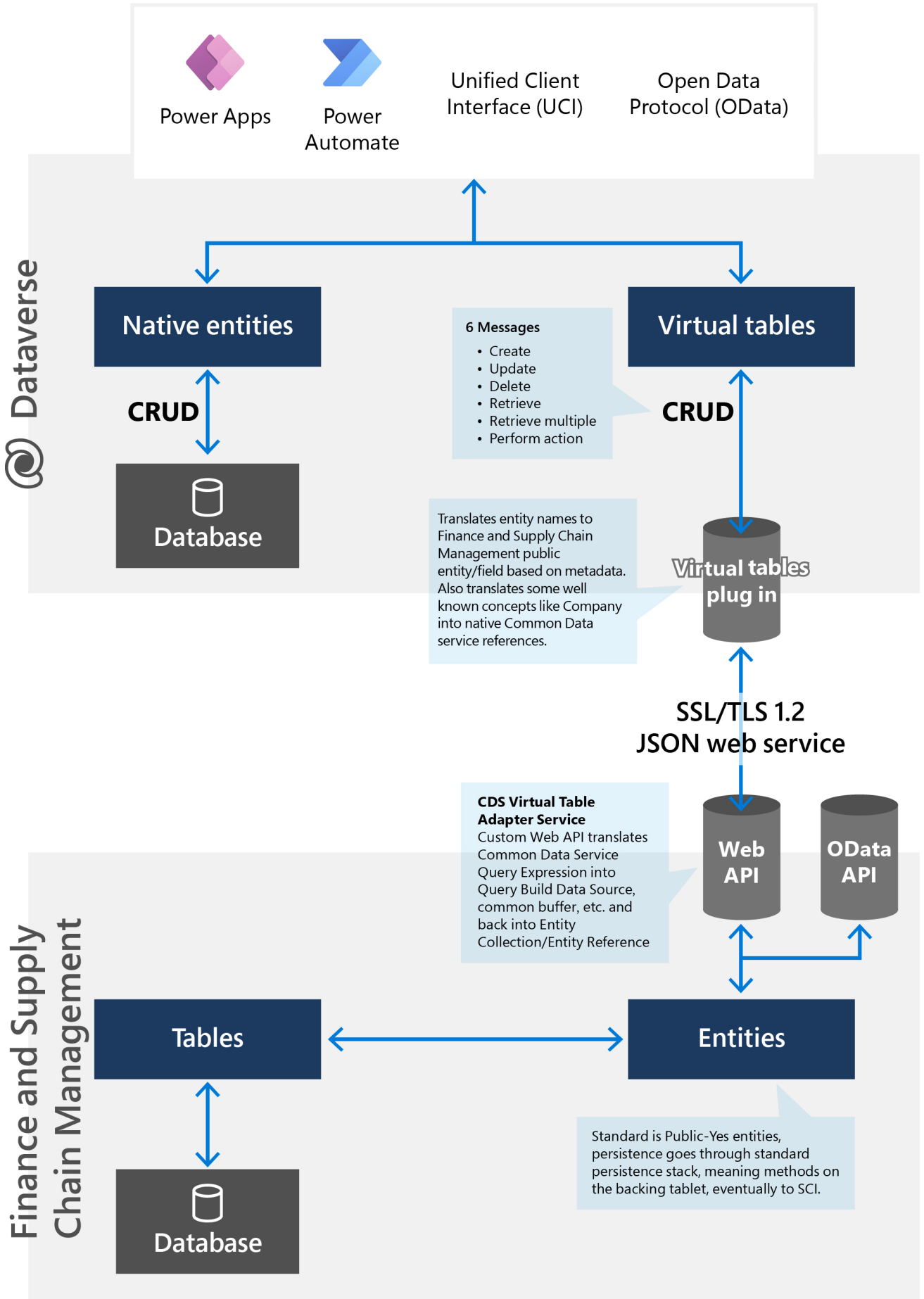


Fig. 15-8

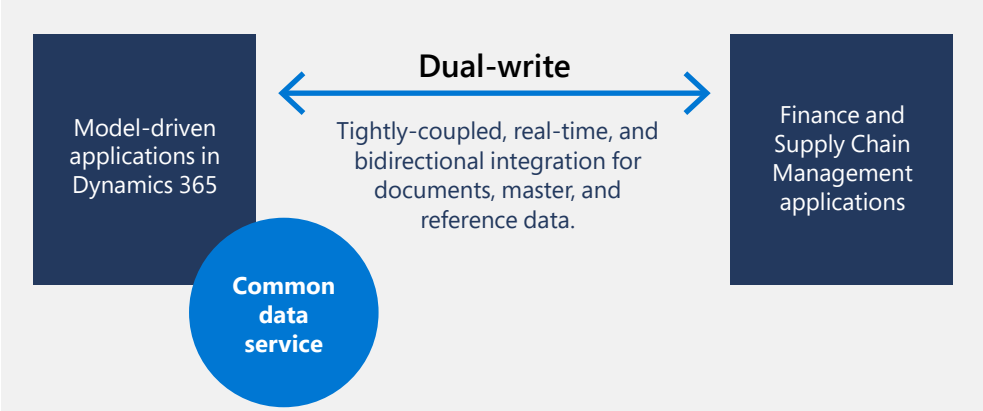

Power Apps
Power Automate
Unified Client Interface (UCI)
Open Data Protocol (OData)
Dataverse
Native entities
CRUD
Database
6 Messages
Create
Update
Delete
Retrieve
Retrieve multiple
Perform action
Virtual tables
CRUD
Translates entity names to Finance and Supply Chain Management public entity/field based on metadata. Also translates some well known concepts like Company into native Common Data service references.
Virtual tables plug in
SSL/TLS 1.2
JSON web service
CDS Virtual Table Adapter Service
Custom Web API translates Common Data Service Query Expression into Query Build Data Source, common buffer, etc. and back into Entity Collection/Entity Reference
Web API
OData API
Finance and Supply Chain Management
Tables
Entities
Database
Standard is Public-Yes entities, persistence goes through standard persistence stack, meaning methods on the backing tablet, eventually to SCI.

Virtual tables provide a mechanism to use Microsoft Power Platform with Finance and Supply Chain Management without having to physically copy data to Dataverse. Virtual tables and dual-write are complementary technologies, as illustrated in **Figure 15-8**. That means they can be used together if required, depending on the different scenarios.

Dual-write, shown in **Figure 15-9**, is a feature that provides near real-time interactions between customer engagement apps and Finance and Supply Chain Management apps for specific integration scenarios. It provides tightly coupled, bidirectional integration between Finance and Supply Chain Management apps and Dataverse.

By bringing data from Finance and Supply Chain Management apps into Dataverse, it is possible to leverage the Power Platform and use low-code/no-code developer models on Finance and Supply Chain Management data. Either as embedded power apps or PowerBI reports or dashboards, in standalone apps, or integrated with other Dynamics 365 Business Applications, without making changes through extensions to the application and data layer.

Fig. 15-9



Advanced practices and tools

As we mentioned earlier, it is always sound advice to be good citizen developers. In Dynamics 365 for Finance and Supply Chain Management specifically, you can ensure that you follow that principle by understanding, learning, and using the following tools and best practices as shown in **Figure 15-10**.

For more on these topics, see the "Reference links" section later in this chapter.

Additional components

It is important to note that the extended product architecture contains several additional components. Along with that are multiple avenues and tiers for approaching a requirement for extension. It depends on the

Fig. 15-10

Tools	What is it?	Description	Characteristics
Best practice check	Best practice checks for X++ and application components are built into Visual Studio. They can be errors, warnings, or informational.	Developers should strive for zero deviations. The best practices are made to ensure and updatable, performing and user friendly solution.	You can stop developers from checking in code with best practice deviations.
Compatibility report	The compatibility checker tool can detect metadata breaking changes against a specified baseline release or update.	The compatibility checker tool is available as one of the dev tools in Platform update 34 and forward. You can use it to ensure that your solutions are backward-compatible with earlier releases before you install or push updates.	Not all breaking changes can be detected by the tool. See the Compatibility checker docs page for specifics.
Traces and trace parser	The users can take a trace of runtime execution directly from the UI. The trace parser to read the trace.	You can use the trace parser to consume traces and analyze performance in your deployment. The trace parser can find and diagnose various types of errors. You can also use the tool to visualize execution of X++ methods, as well as the execution call tree.	The trace parser tool can be found in the PerfSDK folder in your development environments.
Performance timer	Performance timer is a tool in the web client that can help you to determine why your system's performance might act slow.	To open the Performance timer, open your web page with the added parameter debug=develop. You can see counters for client time and server time, and the total time. Additionally, you can see a set of performance counters, a list of expensive server calls, how many SQL queries were triggered by this individual call and which SQL query was the most expensive.	The tool itself has a performance impact.
LCS Logs	Under Environment Monitoring in LCS there a comprehensive collection of tools and information that you can use to analyze and diagnose your cloud environment.	The logs provide for example: ▪ Activity Monitoring: A visualization of the activity that has happened in the environment for given timeline in terms of user load, interaction and activity. ▪ SQL Insights: Logs that include advanced SQL troubleshooting. ▪ Information about slow queries, deadlocks, crashes etc.	The tools under Environment Monitoring are very effective at diagnosing potential or growing performance issues. Keeping an eye on these metrics can help pinpoint problems with extensions.
Customization Analysis Report (CAR Report)	The CAR report is an advanced best practice check tool.	The CAR report can be run by command line in a development environment. The output is an Excel workbook with recommendations issues and warnings.	A clean CAR report is a requirement for the go-live readiness review prior to enabling production.
Understand and avoid breaking changes	A breaking change is a change that can break the code consumers of your code and components make.	Breaking changes are, for example, changes to data model and extended data types, changes to access modifiers on classes and methods and many others. This is especially important in heavily extended solutions, if you are making a basis for a global rollout, if you are making an ISV, or if you have multiple developers sharing common custom APIs or constructs but should always be considered.	That although the application is massive, we tend to only extend the same relatively small subset of elements. It is not as unlikely that you may think that other developers use your components or code.

Tools	What is it?	Description	Characteristics
Log extensibility requests early	If you find a need for an extension point that is currently not available, log the request early. This is done via an extensibility request in LCS.	Extensibility requests are logged to a backlog. Microsoft engineers prioritize all requests, and then work on them in priority order. Please note that Microsoft is not guaranteeing that all requests will be fulfilled. Requests that are intrusive by nature will not be supported, as they will prevent seamless upgrade.	Extensibility requests are following the same cadence as the platform updates.
Proper unit testing	Sometimes developers put a lot of effort into building the extension, but little effort into unit testing it before determining whether it is ready to deliver.	Developers are the first line of defense against bugs, performance issues and semantic issues that may exist in the specification. By simply going through the intended functionality from perspectives such as: ▪ Will this code scale with high volumes? ▪ Does it do what I expect? ▪ Can I do things I am not supposed to? ▪ Could the user accidently do something unwanted? ▪ Does the requirement make sense or force me to break best practices or patterns?	It is a lot easier and cost effective for the developer to find and fix a problem before it is checked in, built and deployed.

specific area and nature of the requirement.

In addition to the components in the Finance and Operation application stack, there are components for Commerce and WMS that require different technologies as well as developer toolsets and skillsets. These include the Point of Sale (POS), The Commerce Scale Unit (CSU), the eCommerce components, the Warehouse Management Mobile App, and the Finance and Supply Chain Management Mobile App.

That means separate requirements for extensions may have to be applied to different components. Slightly different approaches may also be needed, and the complexity and skill sets required may vary. For all of these, it is critical that developers and administrators familiarize themselves with each of the specific SDKs, guidelines, application lifecycles, builds, and deployment methods and the skill sets required before deciding to extend the component.

Reference links

- Extensibility home page - Finance and Supply Chain Management
- Application stack and server architecture - Finance and Supply Chain Management
- Microsoft Power Platform integration with Finance and Supply Chain Management
- Microsoft AppSource – destination for business apps
- Commerce for IT pros and developers - Commerce

- Write extensible code - Finance and Supply Chain Management
- Breaking changes - Finance and Supply Chain Management
- Extensibility requests - Finance and Supply Chain Management
- Grid capabilities - Finance and Supply Chain Management
- Extensibility requests - Finance and Supply Chain Management
- Mobile app home page - Finance and Supply Chain Management
- Take traces by using Trace parser - Finance and Supply Chain Management
- Testing and validations - Finance and Supply Chain Management

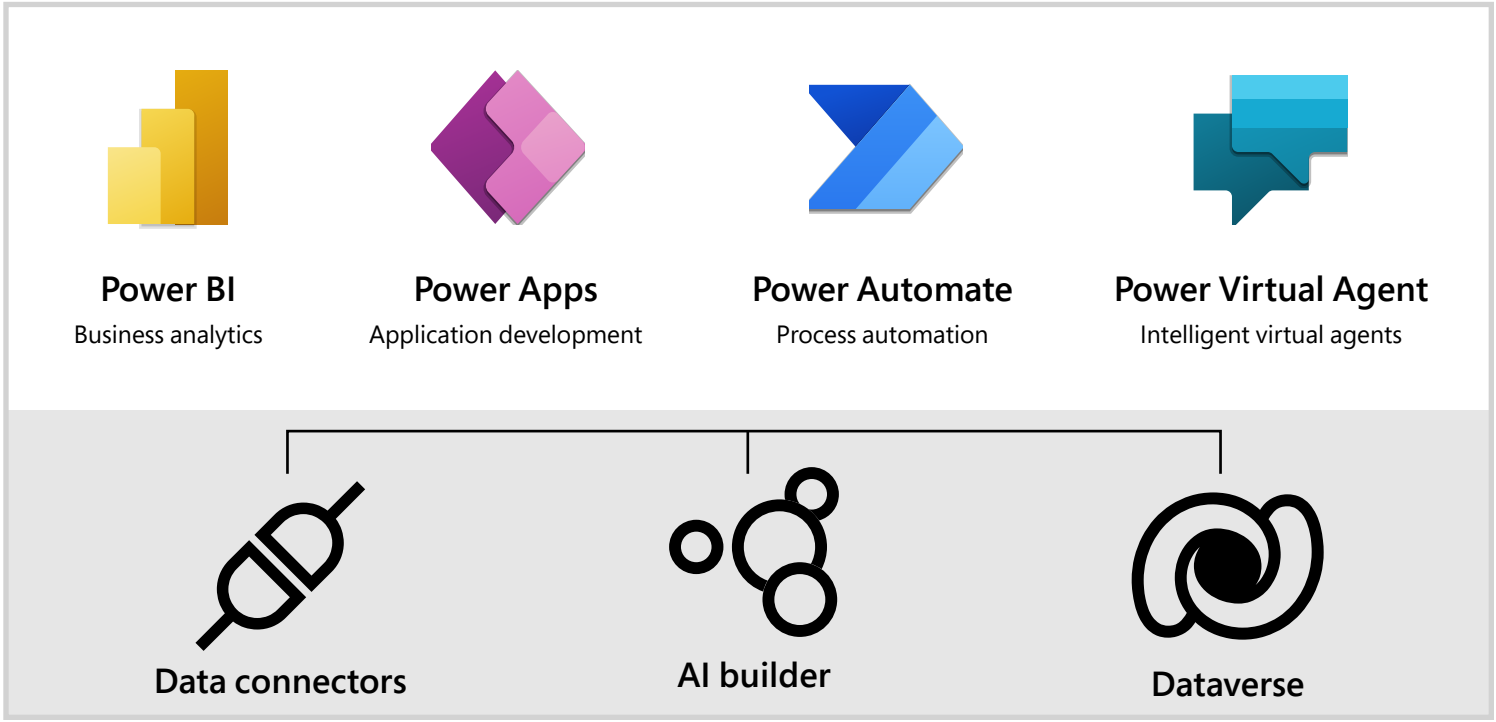
Customer Engagement

The Power Platform, shown in **Figure 15-11**, provides the ability to use configurations, low-code/no-code customizations, and still allows developers to extend programmatically first-party customer engagement apps like Dynamics 365 Sales. Custom business applications can also be created. You can even mix the two approaches to provide applications adjusted to specific organizational needs.

Fig. 15-11

Microsoft Power Platform

The low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications



Until recently, creating custom apps would require professional developers. But by using Power Apps, organizations can quickly build custom business applications that provide rich business logic and workflow capabilities to transform manual business processes into digital, automated processes.

Power Apps democratizes custom business app building experience by enabling non-professional developers to build feature rich, custom business apps without writing code.

When offering this ability to users, it is important to have a central governance entity to determine guidelines for certain aspects of the solution, like security, data model, etc. This prevents redundancy, gaps in the security model, and bad practices/architecture that impact how apps are maintained, secured, scaled, and shipped.

The Microsoft Power Platform Center of Excellence (CoE) starter kit provides a collection of components and tools that are designed to help organizations develop a strategy for adopting and supporting Microsoft Power Platform. This helps organizations implement a CoE to drive innovations and improvement while also providing standards, consistency, and governance.

Power Apps also provide an extensible platform that lets professional developers programmatically interact with data and metadata, apply business logic, create custom connectors, and integrate with external data.

Model the user experience

Power Apps is a high-productivity development platform for business apps and has four major components:

Canvas apps are intended to build a business app from a canvas in Microsoft Power Apps without requiring professional developers to create code. The apps can be built by using drag and drop elements onto a canvas. It also provides the ability to use connectors to integrate data from, and to, a variety of sources.

Model-driven apps are a component-focused approach to app

development, which also does not require code. These apps can be simple or complex. With Model-driven apps much of the layout is determined and designated by the components that are added to the app. The Model-driven apps can be completely custom or from first-party apps such as Dynamics 365 Sale.

Both types of apps are built to be responsive by adjusting and being accessible from different devices.

Power Apps makers can also create external facing Portals that allow users outside their organization to create and view data in Microsoft Dataverse. The Portals experience reuses components in other apps to determine the experience. It is also possible to customize the Portals experience, similar to how other apps are customizable.

Microsoft Dataverse securely stores and manages data that is used by business applications.

Business logic and validation within the platform

Business logic and validation to ensure data quality and reduce repetitive tasks can be applied within the Power Platform by using the following approaches:

- Business rules validate data insertion and provide validation messages within the user interface. As an example, with business rules it is possible to display a message when a certain field does not have data or does not have data in the expected format.
- Business process flows guide users to ensure they enter data and use the solution in a consistent way. An example would be displaying the information that is required to complete a business process, like lead to invoice, this provides a consistent approach for all users while it also improves usability of the solution by making the process more efficient and improving data quality.
- Workflow allows business processes to be automated within the process designer to apply a condition or perform a new action. For example, when closing a support case to send a survey to the customer.
- Business logic with code supports advances developer scenarios to extend the application with code, for example using java script

code for form events or plug-ins that apply business logic to data transaction.

Dynamics 365 applications, such as Dynamics 365 Sales or Dynamics 365 Customer Service, use Dataverse to store and secure the data they use. This enables organizations to build or extend apps by using Power Apps and Dataverse directly against the business data.

Dataverse data is structured according to the Common Data Model, which is a shared data language for business and analytical applications to use. By using Dataverse, makers can simply extend the data model of first-party solutions such as Sales, Customer Service, Field Service or others, and jump-start app development by using the Common Data Model with business logic, security, and integration already built in.

Dynamics 365 Customer Engagement and the Power Platform provide a powerful customization and app making capabilities by using several components that provide the ability to build the appearance and functionality of an app.

These components are distributed in four categories.

Data

The entity designer and option set designer determine what data the app is based on and allow changes to the data model by adding additional tables and fields as well as relationships and components that use predetermined options for users to select.

User interface (UI)

User interface components, like the app designer, site map designer, form designer, and view designer, determine how users interacts with the app and allow changes to be made to the components that display in the app UI.

Logic

The business process flow designer, workflow designer, process designer, and business rule designer, determine the business processes, rules, and automation of the app.

Visualizations

These determine what type of data visualization and reporting the app includes, like charts, dashboards, and reports based on SQL Server Reporting Services.

You can create a range of apps with Power Apps, either by using canvas or model drive apps to solve business problems and infuse digital transformation into manual and outdated processes.

Solution analyzers

Solution checker The solution checker can perform a rich static analysis of the solutions against a set of best practice rules to quickly identify patterns that may cause problems. After the check is completed, a detailed report is generated that lists the issues identified, the components and code affected, and links to documentation that describe how to resolve each issue.

It is important to include solution checker with the solution release cycle, and after releasing an updated version in a lower environment (sandbox), to make sure that any deprecations or issues related to extensions are identified.

Portal checker The portal checker is a self-service diagnostic tool that can identify common issues with a portal by looking at various configuration parameters and providing suggestions on how to fix them.

Power Apps checker web API The Power Apps checker web API provides a mechanism to run static analysis checks against customizations and extensions to the Microsoft Dataverse platform. It is available for makers and developers to perform rich static analysis checks on their solutions against a set of best practice rules to quickly identify problematic patterns.

Conclusion

Cloud-based solutions offer ready-to-use applications solutions that can be easily delivered, reducing the time required for an

organization to start taking advantage of it. SaaS solutions typically reside in cloud environments that are scalable and offer native integrations with other SaaS offerings. These solutions benefit from continuous upgrades that add innovation and new capabilities multiple times per year. This reduces costs and effort and eliminates the downtime associated with upgrades in a traditional on-premises model.

This means that organizations can start using the solution as-is. Still, in some scenarios, additional requirements are needed to add value to the business or to empower users and drive adoption of the new solution. Thus, modern SaaS solutions also provide rich capabilities to further extend it. These can range from simple configurations using a low-code/no-code approach or an extension by using custom code by professional developers.

It is important to consider that the level and depth of these extensions can impact key characteristics, like performance, or increase the cost of supporting and maintaining the solution, but they should hinder the ability to take advantage of innovative functionalities that are delivered as part of the continuous update strategy. Normally, when deciding to extend a solution, a trade-off is necessary between new functionalities and the impact of adding them. For example, if adding too much information to a form it impacts the time required to load the information included in it.

In this chapter we have discussed how these key characteristics can be impacted and the consequences of overextending the solution.

With Success by Design, during the Solution Blueprint workshop, such topics are discussed to establish a clear understanding of the impact that the extensions have on the solution, as well as to identify risks and issues that come with these decisions. One of the most important factors to consider when deciding to extend a solution is how performance is affected. During the Solution Performance implementation workshop, the FastTrack solution architect works along with the implementation partner and the customer to review how the extensions can impact the overall

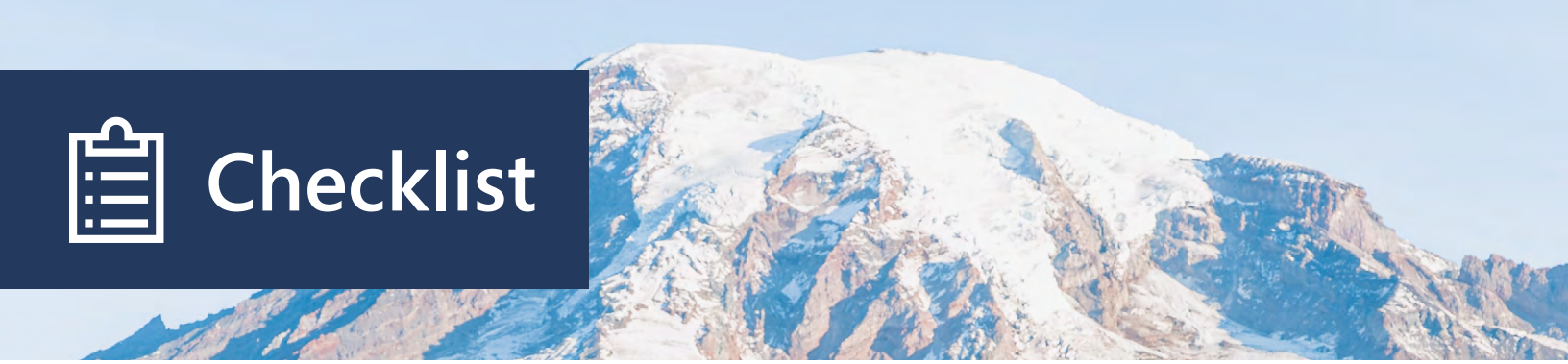
performance of the solution. Typical risks and issues are identified related to overextending forms, synchronous events that impact the user experience, impact on capacity such as service protection limits, resources, and allocation limits.

References

Business Apps | Microsoft Power Apps

Build Apps – Canvas Apps or Model-driven Apps | Microsoft Power Apps

Microsoft AppSource – destination for business apps



Checklist

Define your extensibility strategy

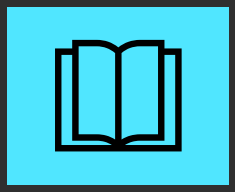
- [ ] Check that requirements to extend the solution are driven by the need to empower users and bring additional value to the business, or industrialize the solution.
- [ ] Ensure the solution doesn't mimic the ways of achieving the same results as the legacy solution or the system being replaced.
- [ ] Understand the platform capabilities and use its strength to simplify and optimize the overall process to get the most of the out-of-the-box experiences.
- [ ] Review if any potential ISVs were considered before deciding to extend the solution. The AppSource marketplace contains ISV-managed solutions that may replace the need to create a fully custom solution.

Considerations

- [ ] Validate that extensions don't negatively impact the user experience, responsiveness, performance, and how the solution behaves across different devices and platforms as part of the organization test strategy.
- [ ] Ensure the extensions honor the security mechanism, privacy, and compliance requirements.
- [ ] Ensure extensions are scalable, tested for high volume, and capable of handling peaks like holiday seasons.
- [ ] Align extensions with ALM automated processes to build and deploy them in an efficient and fast-paced approach.
- [ ] Ensure code and customizations follow only the documented supported techniques, and don't use deprecated features and techniques.

Case study

The power of making it your own



A wealth management services company that delivers personalized services had the need to standardize, simplify processes, drive efficiencies, expose more data, and increase visibility and interoperability to their users.

After evaluating solutions, the company decided to build a modern, completely customized, cloud-based customer engagement platform on Microsoft Dynamics 365.

The off-the-shelf functionalities gave the ability for the users to achieve a single view of the customer lifecycle across all products. Through customizations and further extensibility of the solution, the company added special wealth management services and fully customized the user experience to better serve their customers.

This helped the company streamline processes, increase revenue, reduce costs, enhance transparency, and manage regulatory compliance, and allowed the users to better understand each customer's story, goals, and capacity for risk. The company was able to deliver appropriate, innovative, high-value services and products to maintain successful end-to-end customer journeys across every account.

The power of using off-the-shelf functionalities and being able to further customize and extend the experience for their users set the foundation to improve their internal processes and empowered their users to deliver better services in a more efficient matter.

16

Guide

Integrate with other solutions

Introduction

Together, Dynamics 365 Business Applications provide a rich and mature feature set across a wide variety of industries.

However, there are situations in which you might want to go beyond the boundaries of the application suite and extend processes or incorporate data assets from other sources.

In this chapter, we examine how to integrate Business Applications.

We discuss the importance of keeping business goals at the forefront while aligning integration projects with your organization's overall solution strategy.

In line with the design pillars, we look at how the choice of integration platform should fit into the architectural landscape.

We also look at how to choose an integration design that offers users the capabilities they desire in the short and long term.

We discuss integration patterns available with Business Applications

This chapter covers the following topics:

- Defining business goals
- Choosing a platform
- Choosing a design
- Choosing a pattern
- Challenges in integration
- Product-specific guidance

and consider important characteristics and factors.

And finally, before diving into the product specifics, we walk through some of the common challenges people face when integrating systems.

Defining business goals

To ensure that your integration work aligns with the overall direction of the business, it's important to match each requirement of cross-system processes against the overall goals of the project and the business. To accomplish this, begin your integration work by defining goals that map to the business perspective.

Reasons for integrating

You might have many reasons to want to integrate systems in your solution. Given the array of tools available and the vast ecosystem provided by Microsoft Power Platform, integration across processes and systems is often readily available and a scalable and reliable choice. Here are some examples of when you might want to unify systems by using integration.

Integration to legacy system

You might have a highly specialized system—perhaps a custom-built or heavily customized system that fits a specific set of requirements in your business—that you want to keep in place or phase out over time. With this approach, you would integrate Business Applications into the legacy system to get the benefits of both and allow for business processes to span both systems.

Multi-phased implementation

You might be implementing Business Applications in a planned multi-phased approach, starting with a geographic location, a single division, or a single Dynamics 365 business application—Finance, for example. In this scenario, some level of integration with the parts of the legacy solution that will be implemented in future phases could be necessary.

Regulatory requirements

Regulatory, government, and industry data exchange or reporting are often standard practice or required by law in some industries and regions. It's often reported electronically, but more complex exchanges of data could require a more traditional integration approach.

Financial consolidation

Perhaps your organization is a subsidiary of a larger operation that requires data to be consolidated and reported in a corporate parent entity. This often requires extracts of data to be transformed and loaded into the corporate consolidation system. In some cases, it's the other way around: your organization might expect integration of consolidation and other data from your subsidiaries into your new system.

Multiple system architecture

Sometimes the Dynamics 365 business application is a link in a bigger chain of specialized line-of-business systems. For example, the Prospect-to-cash process might start in one external system where the opportunity is discovered, then go through quote and order management in Dynamics 365 Sales App, and then integrate to other systems for warranty tracking, billing, and financial reporting.

Extract to reporting solutions

Perhaps you might have invested in building a specialized analytics solution that includes a data warehouse with many years of historical data and tailored reporting on top of it. In that scenario, you might want to extract data from Business Applications for transformation and load it into the existing data warehouse.

Many more scenarios are not described here. Some might even be combinations of several of the examples.

When multiple systems "talk" to each other—systems that perform distinct functions in the business—it's not only about making it work. Integration can enable business processes, reporting, and analytics that would not have been possible otherwise.

Goals and benefits

Organizations that integrate systems when building their business solution can achieve several key goals and benefits, including:

- **Process automation** Integration enables the automation of repetitive tasks in the workflow.
- **Data availability at the right time** Communication between different departments is key. If the sales department depends on information from accounting, integration can help the data to flow at the time that the sales department needs it. Whether the data arrives in real time or in batches, you can program it to arrive at a certain time. The right pattern of communication will help satisfy the different business units.
- **Reduced human errors** An error in the data like a typo, an incorrect number, or a button pressed at the wrong time could significantly affect your process, or even worse, degrade the customer experience.
- **Increased productivity** The users become more focused on the processes that add value, since they won't need to reach for or search other systems to retrieve the information they need.
- **Process optimization** Integration simplifies your processes so that you spend less time executing tasks, thereby adding more value to your business.
- **Increased security** By automating processes and data movement through integrations, organizations implement better controls for data access and reduce the need for users to directly work with sensitive data.
- **Regulatory compliance** Automated integrations could help meet some of the controls needed to meet regulatory needs like HIPAA (Health Insurance Portability and Accountability Act) data or 21 CFR (Code of Federal Regulations) Part 11 processes.
- **Reduced cost of operations** Integrations might help reduce repetitive manual activities, human errors, and training activities, which could lead to an overall decrease in cost of operations.
- **Avoiding data silos** Integrations break down data silos and improve the value of the organization's data. With AI, machine learning, and IoT usage on the rise, this data might help drive better insights, data science, and predictive analytics in your company.

For supplemental information, read the Success by Design Integration architecture guidance on Microsoft Learn.

Integration planning

To properly incorporate business goals into the entire project lifecycle, we recommend you plan the integration architecture in the initial stages of the implementation project.

When planning integrations, it's easy to miss a critical point or underestimate volume or complexity. To prevent this, we highly recommend you create a blueprint of the design before starting any specification work and do some calculations of loads and transaction flow.

Success by Design highly recommends that you approach integration work the same way you would an extension project, by following a defined software development lifecycle (SDLC) that incorporates business stakeholders and collects their buy-in. The SDLC should include requirements, design, development, and user acceptance testing, as well as performance testing, deployment, and application lifecycle management (ALM). The work-to-define requirements should be business driven and process focused. For more information, refer to Chapter 7, "Process-focused solution," and Chapter 11, "Application lifecycle management."

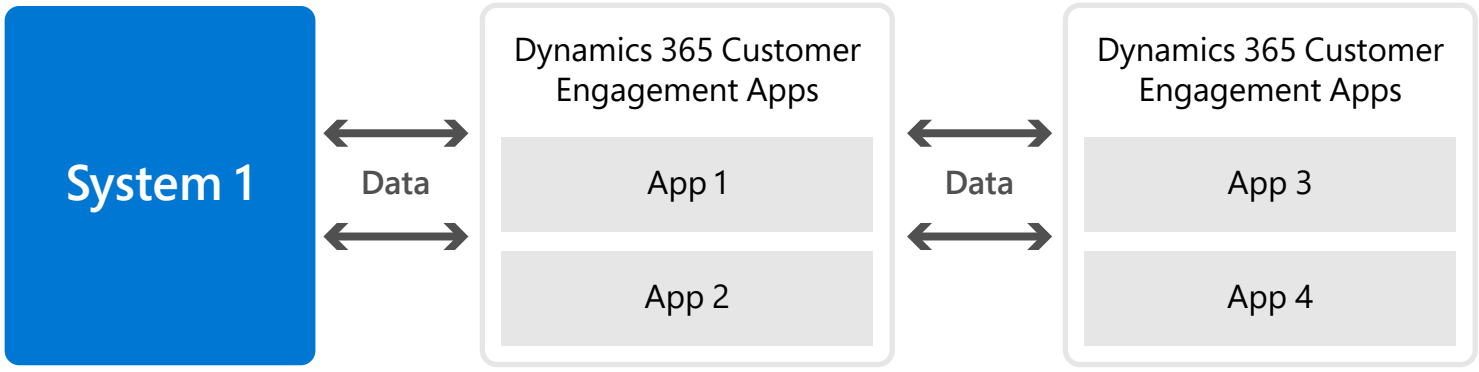
Conceptualizing

Creating a blueprint and thoroughly planning will also help you formulate the testing and performance testing of the solution later in the implementation. That's why integration architecture is a key part of the Success by Design solution blueprint.

To create a blueprint, you can leverage several types of diagrams, which we describe here.

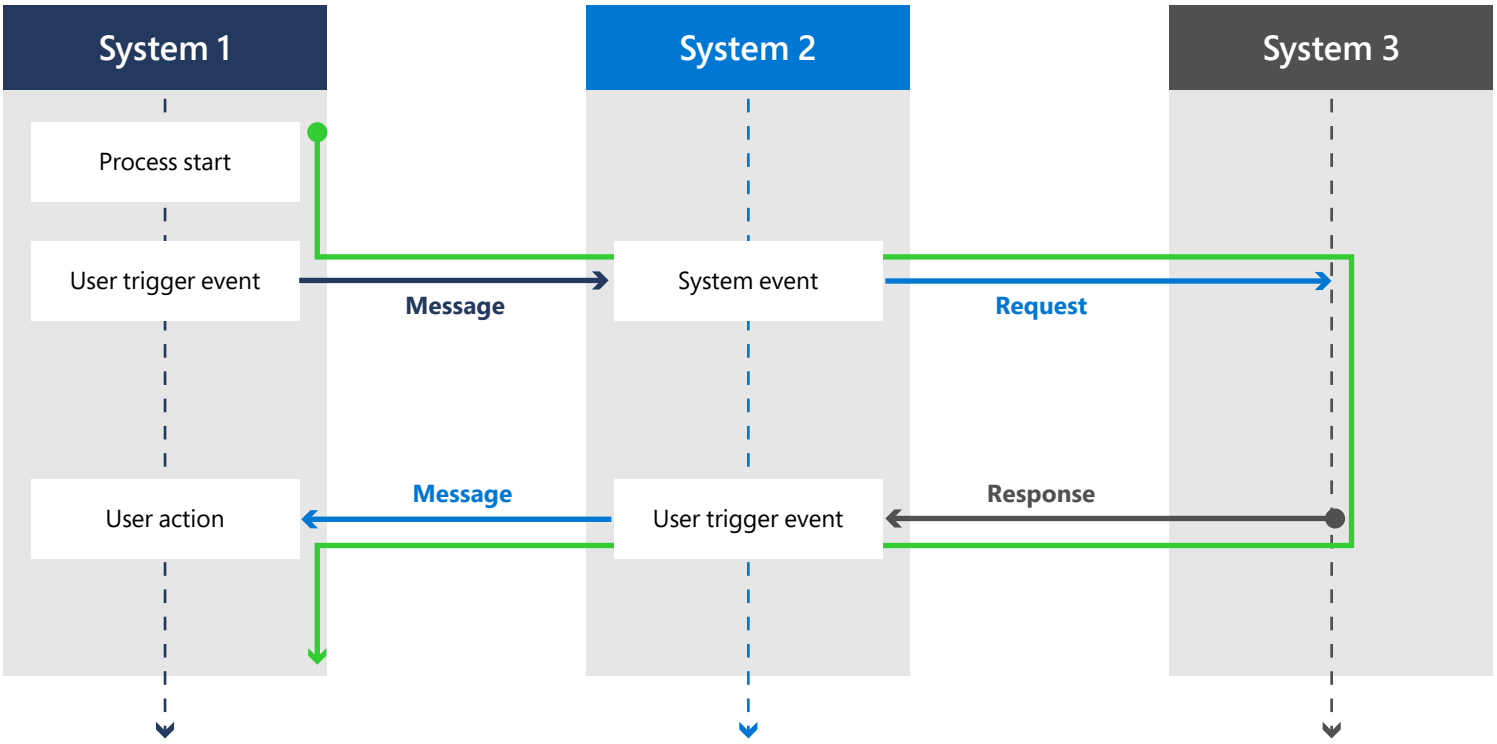
- **Application integration diagram** The application integration diagram is often a high-level representation of solution architecture. An example of one is shown in **Figure 16-1**. Many styles exist, but in its basic form it should provide an overview of which systems in the solution need to be integrated and, ideally, what data is exchanged and the direction of the data flow. Once the overview is established, details can be added about the specific interface touchpoints, frequency, and volume information, and perhaps a link to ALM information, such as a functional design document (or FDD) number or link.

Fig. 16-1



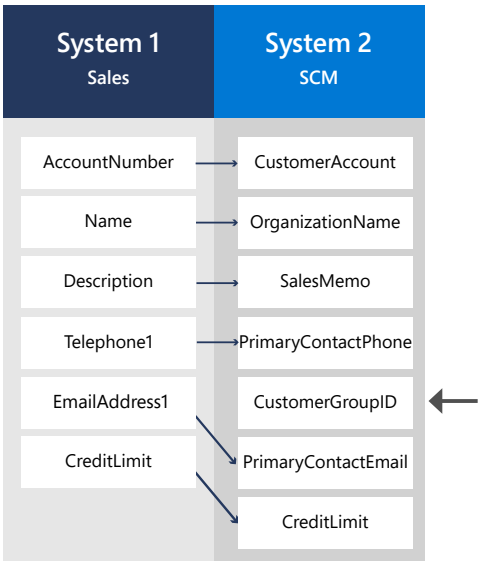
- **Interaction diagram** The interaction diagram defines system and user trigger points, data and process flows, and the sequence of events. It can also uncover unforeseen complexity. By following the flow, even nontechnical team members can understand and validate the design before any work has started. The diagram in **Figure 16-2** illustrates a simple three-system interaction diagram for a simulated process flow.

Fig. 16-2



- **Process documentation, mapping, and volume calculations** Once the high-level overviews are defined, we recommend you document the cross-systems processes in more detail, for example by defining user stories, before starting the system-to-system mapping. Mapping can be done as part of a functional or technical design document. (An example of mapping is shown in **Figure 16-3**.) The design document, along with the process definition, should be detailed enough to ensure that everyone agrees on the scope, process, desired outcome, test criteria, and benefits of the integration. Often such a document also contains information about the expected effort and duration. Calculating the volume of transactions that flow through the interface is important because it helps you decide what patterns to use, and ultimately the size of the different platform components and services needed for the integration.

Fig. 16-3



Choosing a platform

In this section we discuss what you should consider when choosing a platform and, if needed, a middleware solution to achieve the previously mentioned business goals.

Data storage has increased exponentially in recent years, and it will continue to intensify in this direction. The amount of data that's generated and transferred today is practically unimaginable. In addition to the new data that's being created, historical data might need to be preserved. The platform you choose will need to incorporate all of these realities, plus the fact that the amount of generated data varies from business to business and project to project. The platform you choose must be able to reliably handle the storage and transfer of vast amounts of data.

Cloud, on-premises, and hybrid platforms

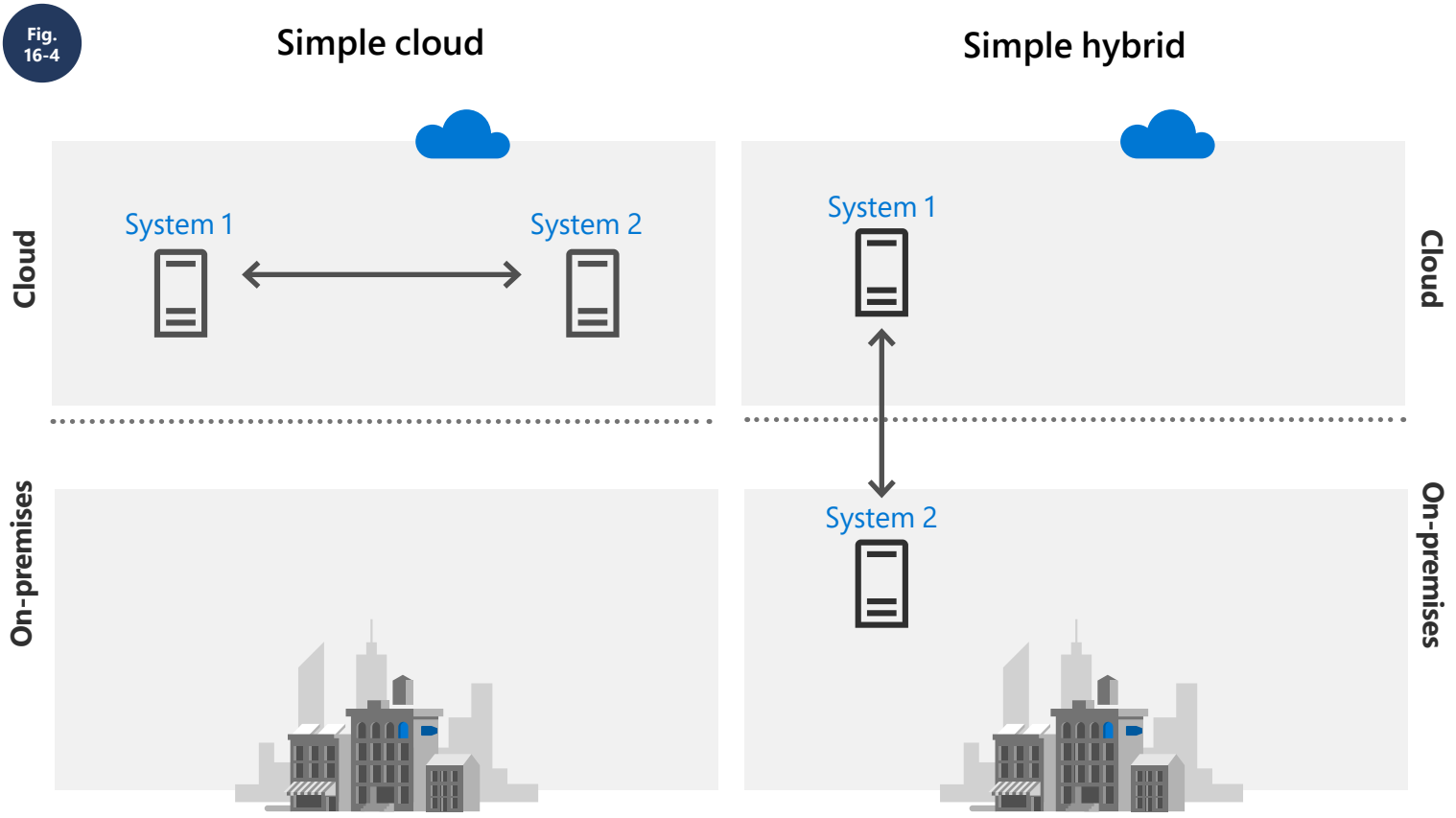
Your organization's size and maturity level are factors when considering various platform scenarios. Most organizations find themselves using some variation of one of the following types:

- **Simple cloud** In the scenario depicted on the left side of **Figure 16-4**, the organization might have acquired the first cloud software as a service (SaaS) application. Overall solution complexity is low. As part of the implementation project, they want to integrate with another cloud service.
- **Simple hybrid** The scenario on the right side of **Figure 16-4** is similar to the previous one in that the organization might have acquired the first cloud SaaS application. Solution complexity is also low. But in its implementation project, the organization wants to integrate to a system on-premises, adding the need for a strategy for crossing the cloud to on-premises boundary.

When implementing a simple cloud or simple hybrid system, be sure to use the capabilities that Microsoft Azure and Power Platform provide. In Business Applications, out-of-the-box low-code/no-code options are available for using Pull designs and RESTful service

endpoints. Such an approach will build a scalable foundation, allowing expansion for years to come.

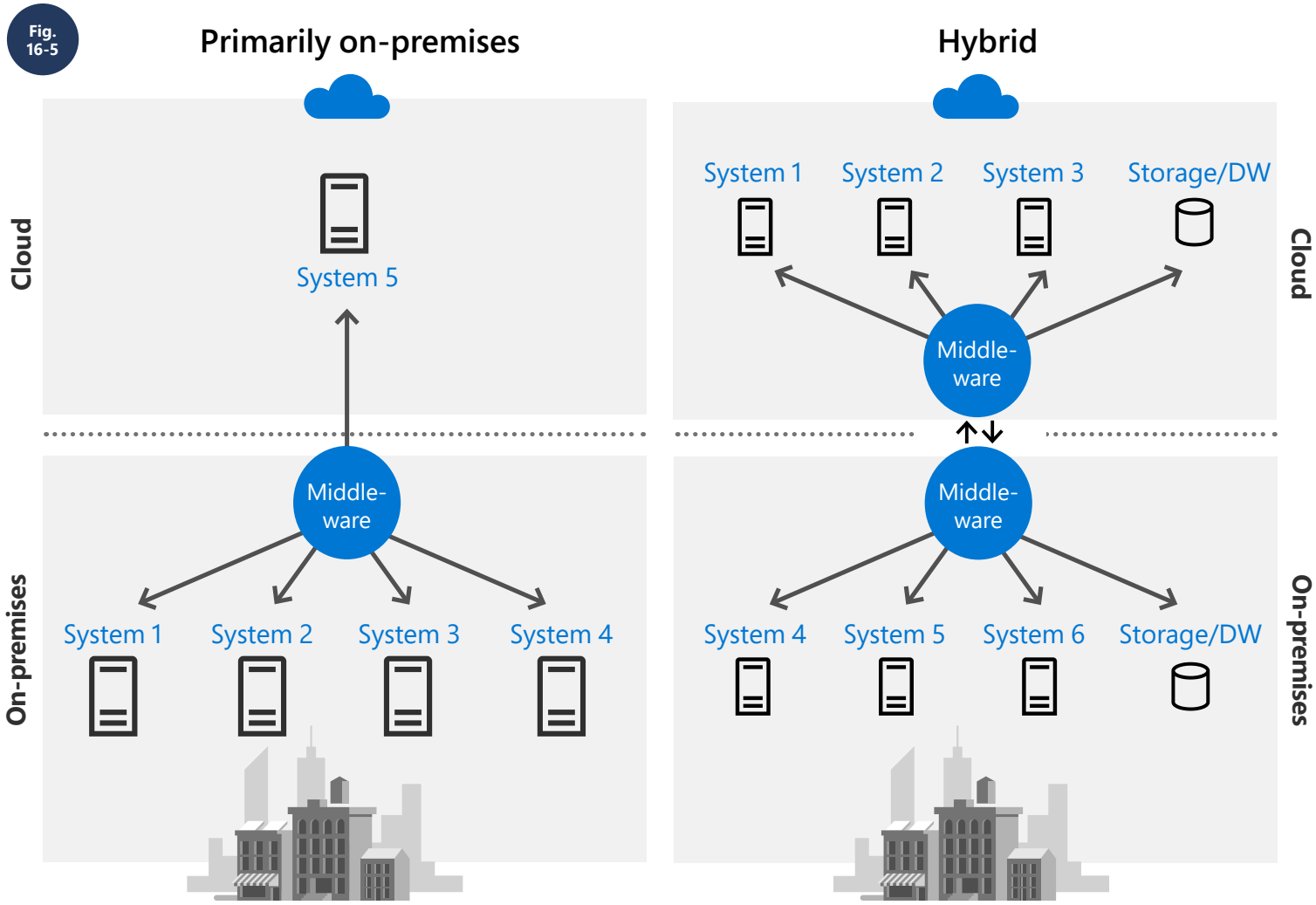
Fig. 16-4



Whether in cloud or hybrid, be sure to not fall for the temptation of building tightly coupled point-to-point, custom synchronous service integrations. They will likely not scale well, and future expansion will likely make the system more fragile. It's good practice to lay a solid foundation early. Organizations that choose not to do that often find themselves having to make tough and expensive adjustments to make the architecture scalable as the business grows.

- **Primarily on-premises** In the scenario depicted on the left side of **Figure 16-5**, the organization has in place a rather complex on-premises solution with multiple line-of-business systems. The organization is using a middleware platform that's based on-premises to connect systems and might be integrating its first SaaS cloud application.
- **Hybrid** The hybrid example on the right side of **Figure 16-5** represents large, diversified enterprises or organizations that are halfway through their cloud journey. The organization is using connected cloud and on-premises middleware platforms that can integrate in hybrid scenarios, and it might add an additional cloud application to the solution architecture.

Fig. 16-5



When planning an incremental migration to the cloud, consider the best options for where and how to achieve a reliable hybrid middleware-to-cloud platform. Consider whether the current middleware platforms and on-premises components are scalable and provide opportunities for leveraging modern technologies, such as low-code/no-code development options, or whether it would make sense to leverage the possibilities within Azure and Power Platform to migrate the current line-of-business systems to the cloud to achieve scalability, elasticity, and extendibility.

Be sure to not silo data assets in on-premises line-of-business systems and disjointed data stores that are difficult for citizen developers, makers, and modern frameworks to access.

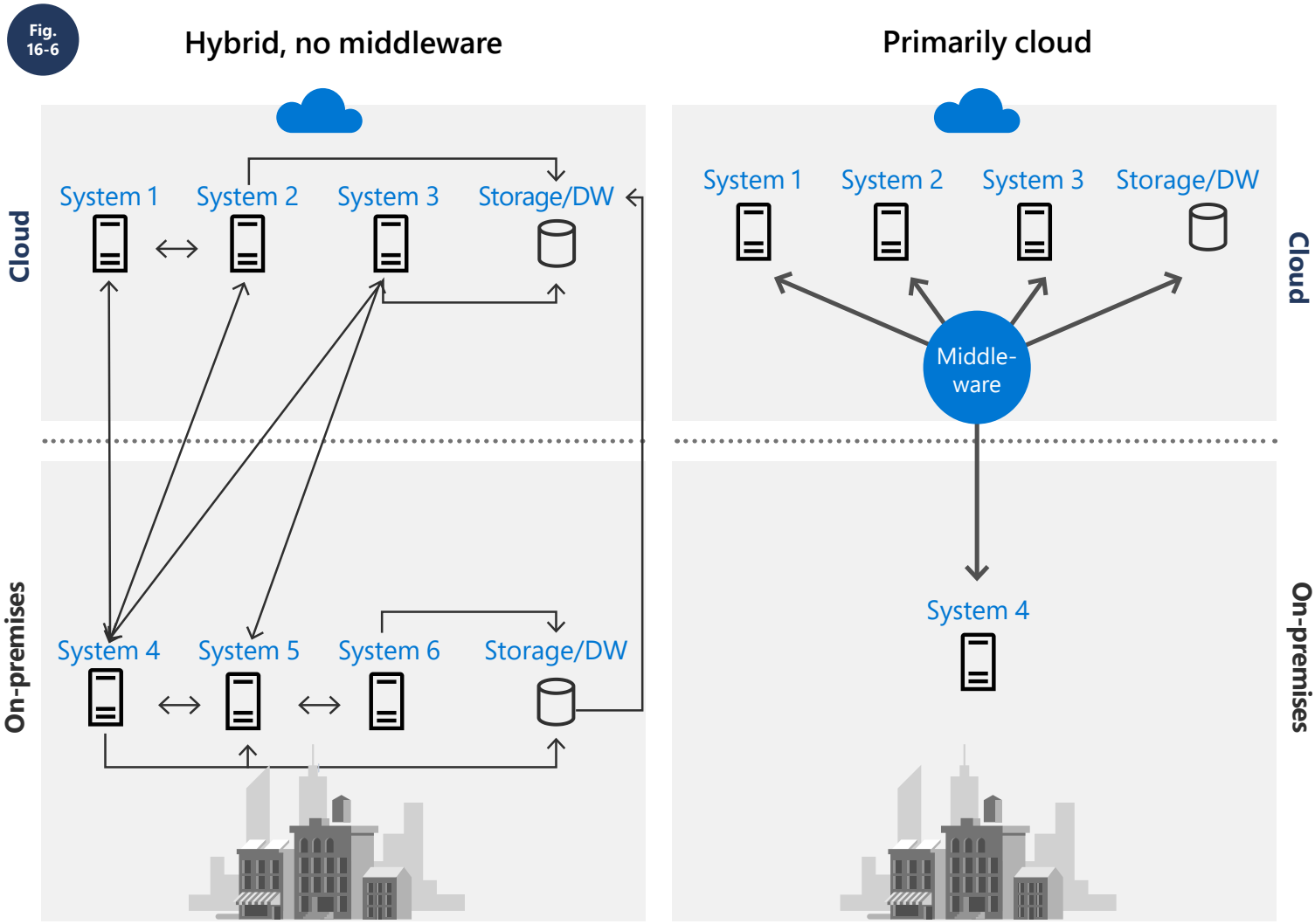
- **Hybrid without middleware** In the scenario shown on the left side of **Figure 16-6**, the organization has likely evolved the integration architecture over time. Multiple systems are integrated

When implementing a system that's primarily cloud, compare the cost of implementing a clear strategy and using a platform and middleware such as Azure and Power Platform with the cost of operating and maintaining the existing integrations.

using different technologies, and it seems System 4 is acting as a hub. No middleware platform is applied, and strategy is likely somewhat loosely defined. The perceived cost of applying an integration platform and middleware is making it necessary to use point-to-point integration when new systems are introduced.

- **Primarily cloud** In the scenario shown on the right side of **Figure 16-6**, the organization has a higher level of cloud maturity, and platform and middleware are platform as a service (PaaS) cloud services. The organization has integrated a few on-premises components that are necessary, and additional line-of-business systems are all in the cloud.

Fig. 16-6



It's important to not continue using resources on maintaining an integration architecture that spans multiple technologies and doesn't have a centralized mechanism for error handling, notification, high availability (HA), and disaster recovery (DR).

Be sure to continue using the cloud-based strategy and the toolset provided by Azure and Power Platform to integrate to on-premises components when necessary.

Middleware provides specialized capabilities to enable communication, transformation, connectivity, orchestration, and other messaging-related functionality. Dynamics 365 Finance and Dynamics 365 Supply Chain Management are business applications that provide integration capabilities to support interfaces, but they are not designed to replace middleware.

Middleware

Integration middleware is software or services that enable communication and data management for distributed applications. Middleware often provides messaging services on technologies such as SOAP, REST, and JSON. Some middleware offers queues, transaction management, monitoring, and error logging and handling. Different middleware platforms can support on-premises, cloud-based, or hybrid scenarios. The following are characteristics you should consider when choosing a middleware platform.

Key characteristics

When deciding whether to integrate to an existing system, you should consider several important characteristics of the solution, the system, and its context.

Business Applications and Power Platform allow users to design, configure, and customize an application to meet a customer's business needs. In doing so, it's important to consider performance and scalability. Ensuring a performant system is a shared responsibility among customers, partners, ISV providers, and Microsoft.

- **Scalability and performance** The planned platform, middleware, and supporting architecture must be able to handle your organization's expected persistent and peak transaction volumes in the present, the short term, and the long term.
- **Security** Authentication defines how each system confirms a user's identity, and you should consider how that will work across systems and with middleware. Authorization specifies how each system grants or denies access to endpoints, business logic, and data. It's important to ensure that an integration platform and middleware are compatible with system security and fit into the solution architecture landscape.
- **Reliable messaging** Middleware typically provides messaging services. It's important that the messaging platform supports the architecture of the integrated systems and provides a reliable mechanism or technology to ensure that messaging across systems is accurately sent, acknowledged, received, and confirmed. This is especially important in situations in which a system or part of the supporting architecture is unavailable. Error-handling concepts

related to messaging, such as idempotency and transient versus persistent errors, are discussed in the "Mind the errors" section later in the chapter.

- **HA and DR** The middleware platform supports the expected level of stability and availability across the interface in line with the cross-system requirements, for example, HA for mission-critical scenarios. Requirements for DR is another consideration for the platform and middleware. For example, if the integrated systems or the supporting infrastructure experiences an outage, it's important to consider what might happen to shared or synchronized data when the outage is resolved or if either system has to roll back to the latest backup.
- **Monitoring and alerting** The platform, middleware, and supporting infrastructure must support the requirements for monitoring activity and alerting users and administrators if there's a problem. In addition to ensuring that the platform can manage adequate levels and venues of alerts and notifications, it's equally important to consider who in your organization is responsible for responding to those alerts and notifications and whether the necessary workload is feasible.
- **Diagnostics and audit logs** It's important to also consider the requirements for monitoring cross-system activity and for audit trails and logging. You should consider whether the planned platform and middleware support that.
- **Extensibility and maintenance** It's important to consider the impact of additional business requirements in the long term. Considerations include the following: the level of effort required to extend the integration beyond the first implementation; whether the integration requires expertise in multiple technologies and changes to both systems or whether there is a single low-code/no-code point of extensibility; how and how often updates are applied; and what level of technical and functional support is available.
- **ALM** To keep costs as low as possible, consider how you can ensure effective lifecycle management, version control, and documentation of the integrated solution. You should verify that all components and parts of the solution can use ALM tools such as Azure DevOps. For more information about ALM, refer to Chapter 11, "Application lifecycle management."

Data consistency across multiple systems is important, and for that to occur you need to ensure data quality. Keep in mind the saying that "if garbage comes in, garbage comes out"—it's important to verify that your data flows correctly and in a timely manner across all the integrated systems.

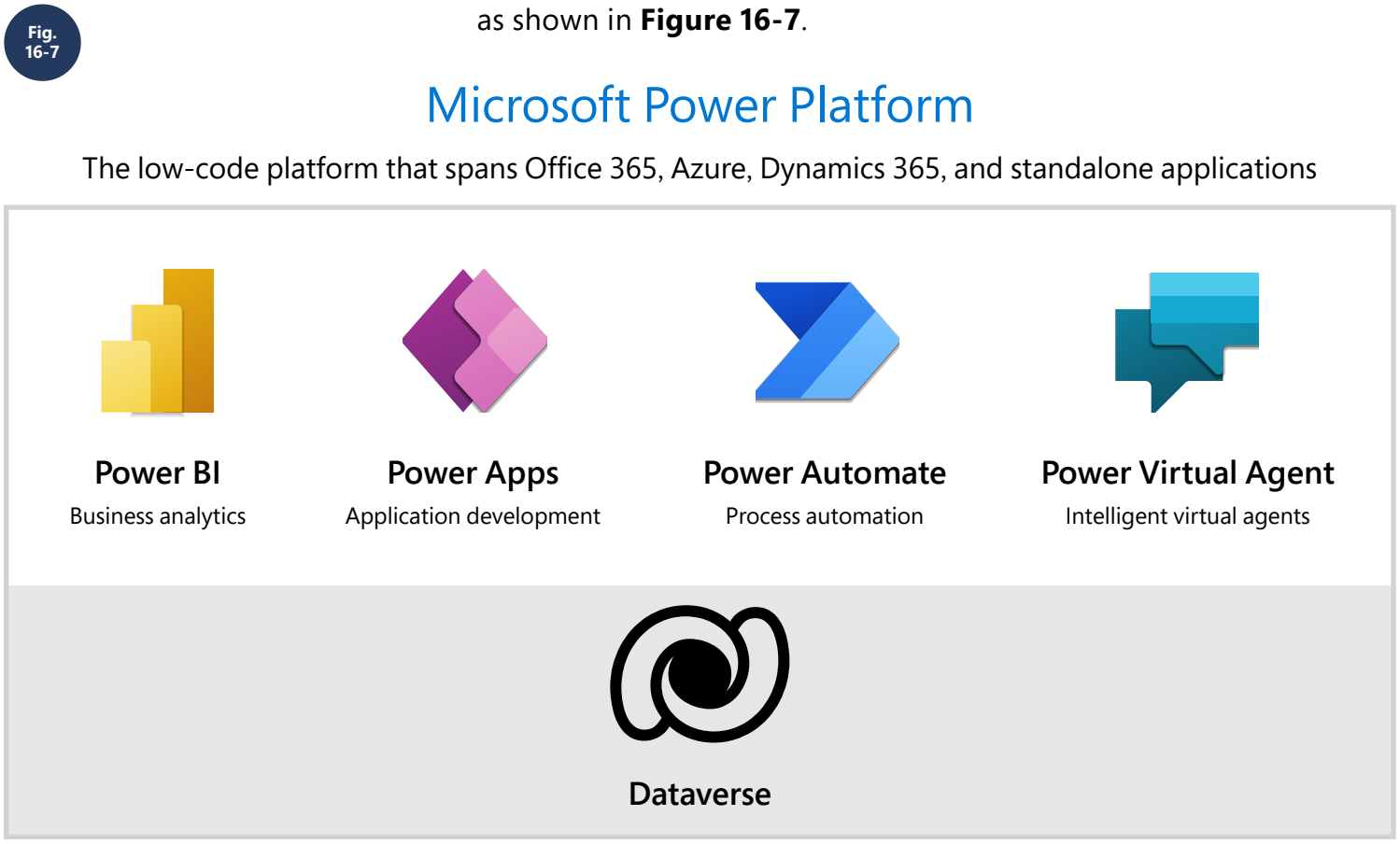
The following section discusses the preferred platform to use to integrate with Business Applications on Azure.

Power Platform and Dataverse

Power Platform enables integration between your organization's systems. It also provides opportunities for your organization to use low-code/no-code development models as well as more complex developer platforms to integrate solutions. In Business Applications, this avenue for integration and much more becomes available when you use Microsoft Dataverse.

Dataverse provides a data foundation for applications in Dynamics 365 and Power Platform. When your solution components live in Dataverse, you can natively connect and extend the solution using Power BI, Power Apps, Power Automate, and Power Virtual Agents, as shown in **Figure 16-7**.

Fig. 16-7



The primary benefits of using an integration pattern based on Dataverse are as follows.

Works with any type of app

- Dataverse works with a broad set of data sources through connectors or Microsoft Power Query.
- Dataverse has built-in connectors through Power Automate that offer configurable, deep integration to not only Microsoft cloud services such as SaaS, Business Applications, and Azure, but also to popular applications such as Adobe, Mailchimp, and Twitter.
- Dataverse provides a modern REST-based API, a vast developer toolkit, and a growing list of samples that link two or more connectors, which means that you often don't need to start your integration project from a blank slate.

Works with any kind of data

- Dataverse is integrated into data-centric tools and services such as Microsoft Excel, Outlook, Power BI, Power Query, and AI Builder that are traditionally used by knowledge workers and integration engineers.
- The Dataverse fabric has built-in analytics, reporting, and AI that you can use to provide insights into your organization and support decision making. You can obtain analytics and reporting data using low-code/no-code options or by using the full capability set of Azure Data Factory, Power BI, and Azure Databricks. For more information about analytics and reporting, refer to Chapter 13, "Business intelligence, reporting, and analytics."

Let's take a look now at how you can use Power Platform and Dataverse with Business Applications.

Dynamics 365 Customer Engagement apps

Customer Engagement apps data is securely stored and managed in Dataverse. This means that the Customer Engagement apps can natively use Dataverse capabilities to integrate to other applications.

Integrate Finance and Supply Chain Management apps and Dataverse

Finance and Supply Chain Management apps are built on a specialized cloud SQL schema, optimized for online transactional processing (OLTP). However, you can integrate into Dataverse through several design patterns, some of which are seamless and synchronous or near real time. When you connect Dataverse and Finance and Supply Chain Management apps, you're using a rich set of predefined denormalized endpoints in Finance and Supply Chain Management called data entities. For more information about how to connect Finance and Supply Chain Management apps to Dataverse, refer to the "Integrate business applications" section later in this chapter.

Choosing a design

Many factors influence the choice of patterns for your integration. Integration scenarios can be grouped roughly into three kinds: UI, data, and process integration. While there is some degree of gray area and overlap between them, there are also some distinct characteristics and expected behaviors.

UI integration

In UI integration, the primary point of integration is centered around an action that's performed on the UI. The integration might or might not trigger business logic or cause anything to be written to the system. UI integration creates a seamless user experience even though the data and process might exist in separate systems, as shown in the example in **Figure 16-8**.

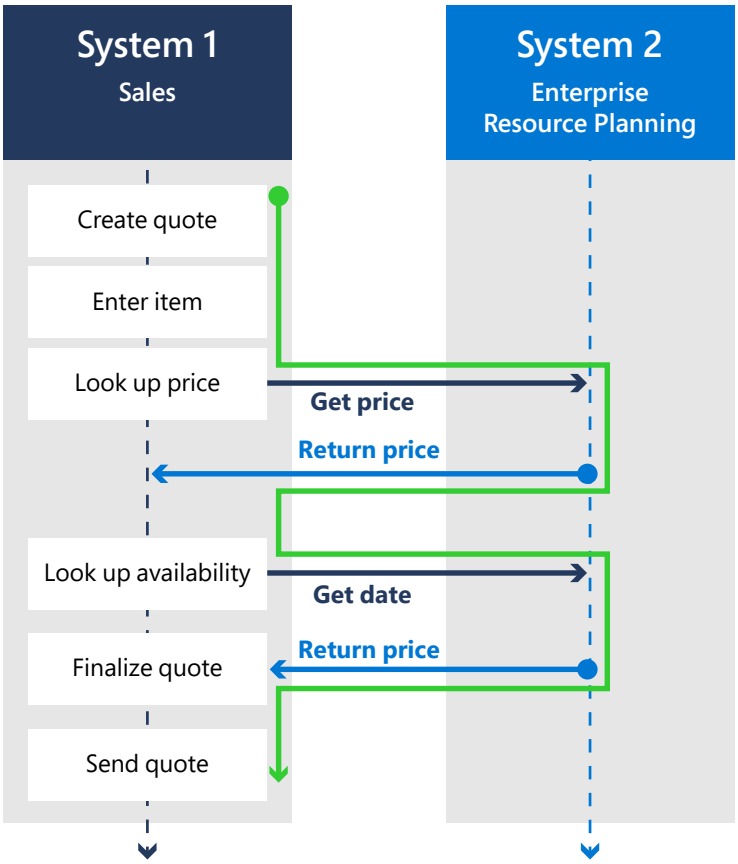
In this figure, an order taker at a company works in the Sales system to take an order from a customer, and in that process the order taker looks at real-time information from an ERP system.

The UI integration allows the order taker to answer the customer's inquiries accurately and immediately without switching systems. The information might or might not be used for the order-taking process.

A data warehouse provides capabilities to store, process, aggregate, analyze, data mine, and report on both current and historical data sets typically with data aggregated from a varied set of cloud and line-of-business (LOB) applications. A data warehouse has specific architecture optimized for analytical functions. An Enterprise Resource Planning (ERP) system such as Dynamics 365 for Finance and Supply Chain Management maintains a highly normalized data set optimized for transactional processing and business functionality. It's important to select the appropriate data warehouse platform to meet your organization's data warehouse and analytical needs.

For more information on reporting and analytics solutions, refer to Chapter 13, "Business intelligence, reporting, and analytics."

Fig. 16-8



The simplest form of UI integration is when a widget, canvas app, or view from one system is embedded into the UI of another system without ever touching the business logic or data of that system. This enables users to focus on their tasks—for example, scheduling interviews with job applicants or providing services to customers—without the need to switch between several LOB systems. Sometimes values can be passed from the hosting system to the hosted system to provide session context from one system to the other.

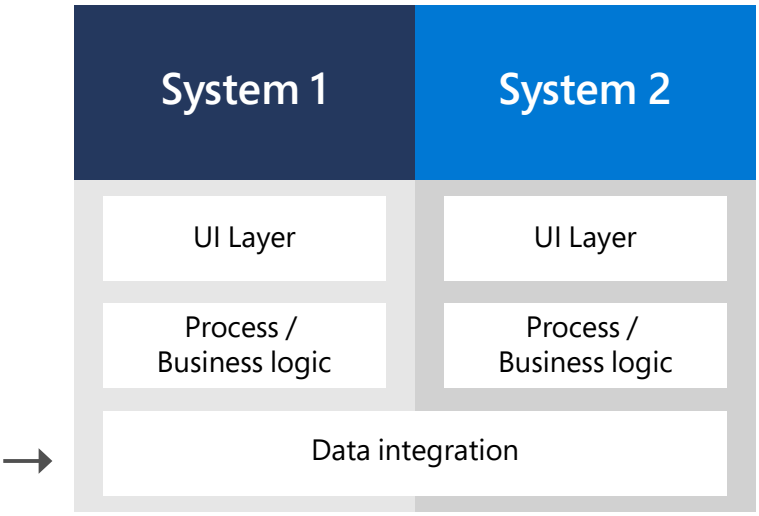
The following are additional examples of UI integration:

- Commodity price ticker; an email view
- On-hand inventory
- Embedded Power BI dashboard showing the current day's production quality issues in a manufacturing plant

Data integration

Data integration, shown in **Figure 16-9**, is integration between systems that takes place at the data layer, and data is exchanged or shared between systems. Data integration is different from process integration in that both systems work with a representation of the same data, whereas in process integration the process starts in one system and continues in the other system.

Fig. 16-9



We use data integration patterns in the following scenarios.

- Master or transactional data that isn't the central part of a single, continuous business process is synchronized between a process in one system to a process in another system.
- Data is shared or exchanged between systems when needed for calculations, classifications, or references.
- Data is shared or exchanged between systems

A key characteristic of UI integration design is that it's embedded.

The benefit of UI integration is that a user can retrieve and provide real-time information from multiple sources without switching between systems, thereby saving on training, user licenses, and more importantly, time when interacting with customers.

so that actions that happen in one system are reflected in the other system.

- Aggregate data from a system with a detailed level of data is exchanged to a system with a higher level representation of data.

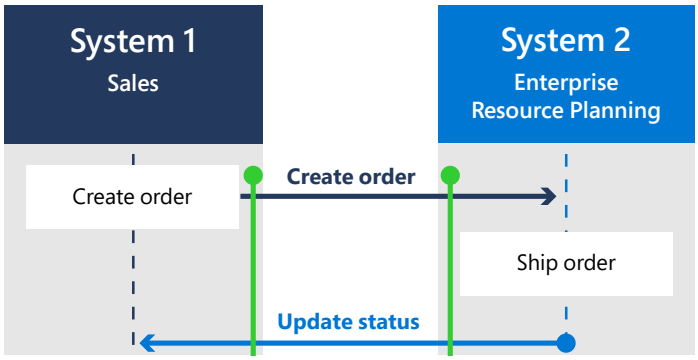
The following are examples of data integration:

- Aggregate marketing data assets
- Bank account reconciliation
- Order status updates
- Exchange rate updates
- Postal code and city list synchronization
- Accounts and customer data synchronization

Keep in mind that often data doesn't originate from a system within your own organization; it can come from an external source to upload. Conversely, data might be extracted from your system to be sent to an auditor, a regulatory body, or an industry data exchange.

In the example in **Figure 16-10**, orders are synchronized between Sales and an ERP system. When the order is created in Sales, the ERP system sends regular updates of the order data to Sales, which enables users to find accurate order status information.

Fig. 16-10



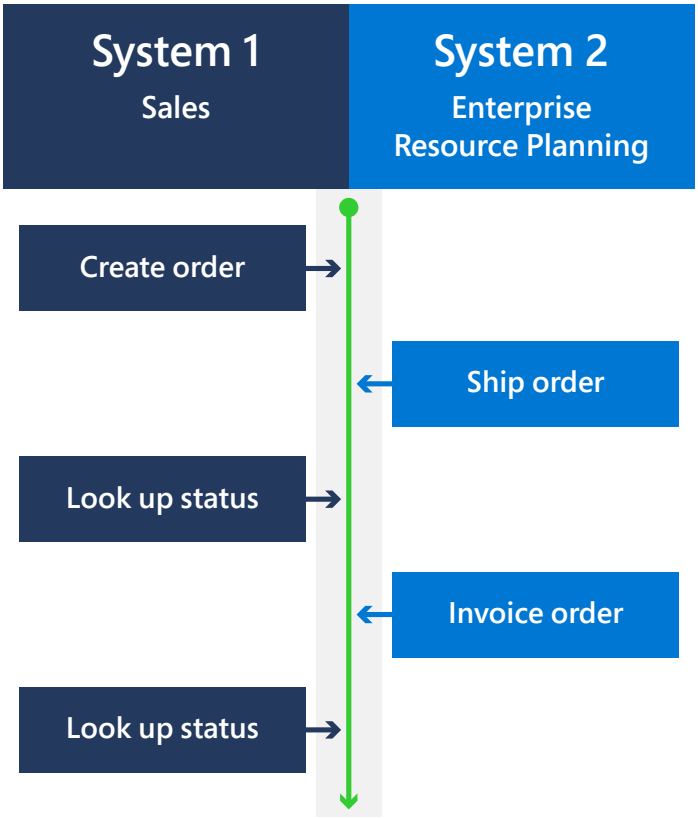
Data integration is also a useful integration type because multiple features within Business Applications enable administrators or users to configure many of these integration points out of the box, for example, Microsoft Teams, journal uploads for expenses or bank account statements, exchange rate synchronization, and features for regulatory reporting and extracts.

When designing data integration, you should consider which system will be the system of record, or owner, of the information. There are scenarios in which this is clear cut, for example, uploading worker data from Dynamics 365 Human Resources into Finance and Supply Chain Management apps (the former is the system of record). But there are also scenarios in which each system owns a separate part of the overall entity, for example, the integration of Accounts to Customers between Sales

and Finance and Supply Chain Management. In this scenario, Sales is the owner of the basic information, but that information is just a small subset of the fields in the customer entity in Finance and Supply Chain Management.

Another type of data integration, shown in **Figure 16-11**, is when two systems share the data layer, so updates to one system are instantly reflected in the other system. This is possible with Business Applications in Dataverse.

Fig. 16-11



First-party model-driven applications such as Sales, Dynamics 365 Field Service, and custom Power Apps all share the same data layer, even if applications are built and deployed separately.

With Dataverse it's also possible to integrate Customer Engagement and Finance and Supply Chain Management Apps so that a similar seamless experience can exist between Sales and the ERP system.

Systems may have separate processes and even different business logic, but if the underlying data layer is the same, the need for transfers of data, synchronization, and programming to transform the data is completely eliminated. This kind of integration is now possible because the development of shared data stores such as Dataverse set standards for how a certain kind of data is defined in the data layer.

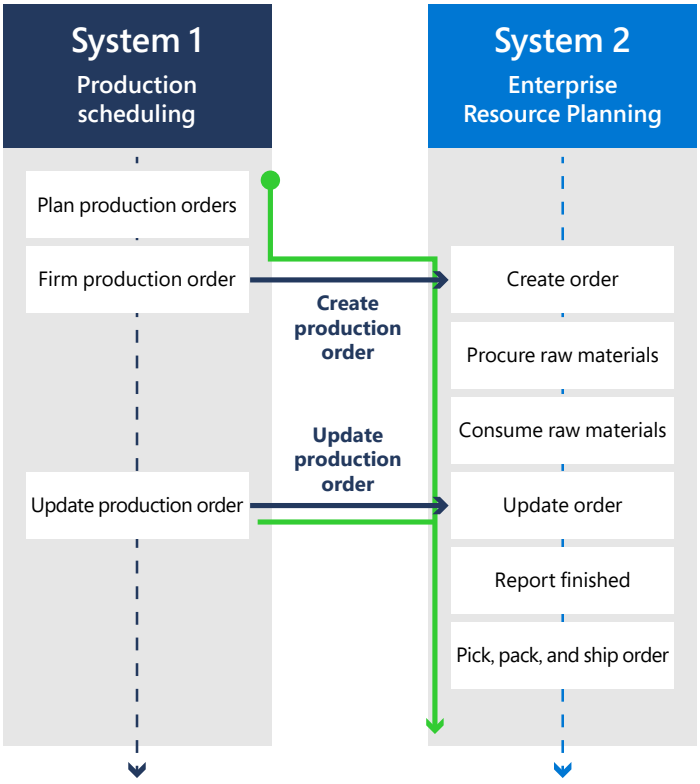
A key characteristic of process integration design is that it's event driven.

The benefits of process integration are accuracy, efficient timing of information and activities in the organization, and reduction of manual errors.

Process integration

Process integration refers to when a business process is designed to span multiple systems. There are many varieties of process integrations, such as from a plan-to-produce workflow in which production forecasting or block scheduling occurs in a Production Scheduling System and the rest of the manufacturing and supply chain management process (production, order management, and fulfillment) and the billing process occur in an ERP system. **Figure 16-12** shows this type of integration.

Fig. 16-12



In this scenario, the business process spans two systems. The user in the Production Scheduling System creates and maintains the production schedule, thereby initiating the business logic to procure, produce, and reserve the order in the ERP system for later picking and shipping. User interaction might be required in the ERP system, but the point is that the event of creating the order in the Production Scheduling System triggers certain business logic in the ERP system, and the rest of the business process steps are handled either automatically or by the appropriate users in the ERP system.

Process integration can span multiple systems and often requires batched, scheduled, and sometimes near real-time integration.

In the example shown in Figure 16-12, without integration, orders would have to be manually created twice, increasing time and risking errors such as typos and missed updates.

Design options on Power Platform

Let's now discuss some of the common services available across Power Platform and Azure that can help in designing a dynamic, scalable solution.

With Dataverse, you can leverage Power Platform to integrate apps. You should consider several components and features of Azure and Power Platform when designing an overall solution as well as individual integration.

Power Automate

Power Automate provides low-code/no-code solutions to help you automate workflows and build integration between various applications. Power Automate automates repetitive and manual tasks and seamlessly integrates business applications inside and outside Power Platform.

In fact, Power Automate has connectors to more than 300 specific applications and technologies and a vast selection of integration samples to start from, all with the goal of helping users to be more productive. Power Automate is available to Office 365 users who have the appropriate license and access. For more information, read about Power Automate and "What is Microsoft Dataverse?"

Azure integration frameworks

Azure provides designers with a rich, scalable feature set and user-friendly design elements as well as the option to build more complex integration solutions. Developers can use tools including Visual Studio, Visual Studio Code, and Azure DevOps.

Power Platform and Azure together can handle integration requirements from simple scenarios to enterprise architecture scenarios.

For more information, visit "Azure Logic Apps" and "Azure Integration Services."

Choosing a pattern

Choosing the right pattern is a critical part of successfully implementing integration between systems. When choosing an integration pattern, you should consider factors such as what its main functionality is and how it's built, including platform, language, user interface, and the connectivity type it handles. We recommend that you also consider what type of actions you need the integration to perform, such as the following:

- **Data types and formats** What types of data are you sending—transactional, text, HTML?
- **Data availability** When do you want the data to be ready, from source to target? Is it needed in real time, or do you just need to collect all the data at the end of the day and send it in a scheduled batch to its target?
- **Service protection and throttling** When you use certain integration patterns, service protection might be built in so that there's a maximum number of records allowed because the performance decreases with quantity. Sometimes the provider

places this limitation on you, but in other cases, if you're offering integration points to other systems, you want to impose a limitation or throttling to ensure that the general performance of your system is not impacted by traffic from external requests.

- **Transformation of data** Do you want to convert or aggregate transactional data into analytical data? Are you changing .txt to HTML?
- **Triggers** What action will trigger sending data from the source to the target?
- **Trigger actions** Do you want specific actions to be automated after the data arrives at the target?
- **Process error handling** What type of monitoring will you put in place to detect any issues with the interfaces? What type of alerts do you want to handle the patterns that you'll be using?
- **Flow direction** Is the flow bidirectional or unidirectional?
- **UI** The UI determines how the data is going to be displayed once it arrives at the target system. The UI determines the user experience—how the user will interact with and handle the information.
- **Scalability** How will the interface pattern handle the expected transaction volumes in the present, short term, and long term? What would happen if you exceeded those expectations?

Pattern directions

Let's take a closer look at some common patterns for individual integrations and the pros and cons for each. This table is generalized; for more information, refer to the "Product-specific guidance" section in this chapter.

Pattern	Mechanism	Trigger	Considerations	Use when
Real time or synchronous	Data is exchanged synchronously, invoking actions via services.	User action or system event.	**Pros**: Fast request and response round trip. Real-time values and information. **Cons:** Small payloads. Action-reaction can cause tight coupling of systems. Risk of ripple effect from minor outages. Sensitive to latency.	Generally, not a best practice to use because of the risk of processes getting stuck and solutions that are fragile. Use when real-time information is critical.

Pattern	Mechanism	Trigger	Considerations	Use when
Asynchronous	Data is exchanged unattended on a periodic schedule or as a trickle feed using messaging patterns.	Scheduled and user initiated. Can wait for off hours or idle time.	**Pros:** Loose coupling of systems makes the solution robust. Load balancing over time and resources. Can be very close to real time. Timely error handling. **Cons:** Delay in response and visibility to changes across systems.	Most recommended integration patterns and technologies supported are asynchronous, although they can be near real time.
Push	One system puts (pushes) data into another. Information flows from the originator to the receiver.	Originating system user or system event.	**Pros:** If technical expertise lies within the pushing system, the custom effort lies here. Good for reactive scenarios. **Cons:** Pushing system might not have visibility into availability and load and idle times in the receiving system.	For reactive scenarios. Receiving system provides a turnkey API and organization's developer skillset is with the originating system.
Pull	Receiving system requests data from the originator—a subtle but significant difference from the Push pattern.	Receiving system request based on schedule.	**Pros:** If technical expertise lies within the pulling system, the custom effort lies here. Good for proactive scenarios. Clear visibility into availability and load and idle times in the receiving system. **Cons:** Originating system might not have the APIs needed to pull from.	For proactive scenarios. We might not have the option to add triggers or events in the originating system. Originating system provides a turnkey API and organization's developer skillset is with the receiving system.
One-way sync	Data from one system is synchronized to another by one or more trigger events.	Data state, system, or user event.	**Pros:** Establishes a clear system of record. Simple conflict resolution. **Cons:** Sometimes the receiving system or non–system of record doesn't have built-in capability to make the data read only, which can confuse users.	One system is owner or system of record and other systems consume that data. Consider scenarios in which the originating table is a subset of a similar table in the target.
Bidirectional sync	Data from two or more systems are synchronized.	Data state, system, or user event.	**Pros**: Data is kept in sync across applications. Acquired divisions on multiple platforms can continue to use their existing systems. Users can use their system to make changes. **Cons:** Complex conflict resolution. Redundant data is replicated for each system. Synchronized data might be a subset of data in systems. The rest must be automatically given values or manually updated later.	For dual-write integration patterns. Scenarios in which there isn't a clear system of record. Data from best-of-breed systems should be available in Dataverse for Power Apps and other tools and services.

Pattern	Mechanism	Trigger	Considerations	Use when
Aggregation	Data from a specialized system is integrated to another system on an aggregated level for processing or reporting.	Any.	**Pros:** Detailed data is kept in the system where it's used. Aggregation can derive a small dataset from a large one, thus limiting traffic across platforms. **Cons:** Users often expect to be able to drill down to the detailed level. While this could be done with embedding, it does require additional integration complexity or users operating in two systems.	Aggregates are needed for calculating or processing, for example, on-hand inventory by warehouse, revenue by invoice header, or revenue by customer by day, or operational data for posting in a finance system.
Embedding	Information from one system is seamlessly integrated into the UI of another system.	User event.	**Pros:** Simple because the data remains in the originating system. **Cons:** Difficult to use the data for calculations for processing.	A mix of information from first-party applications (for example, Bing, Power BI, and Exchange), third-party components, canvas apps, or other information embedded in the UI of an application. This is common practice in Customer Engagement.
Batching	Batching is the practice of gathering and transporting a set of messages or records in a batch to limit chatter and overhead.	Any.	**Pros:** Great for use with messaging services and other asynchronous integration patterns. Fewer individual packages and less message traffic. **Cons:** Data freshness is lower. Load in the receiving system can be affected if business logic is executed on message arrival.	Whenever it isn't necessary to transmit individual records.

Let's take a moment to discuss synchronous integration patterns and near real-time patterns because these are often mischaracterized and mistakenly used interchangeably.

An example of a synchronous integration pattern is a process in which a user performs a web service–based lookup of current on-hand values from one system to another. In this example, the user waits for a response before continuing the process, thus making the process both latency and availability dependent. This is known as coupling and it should generally be avoided.

A near real-time pattern for the same design scenario provides a user's system with fresh on-hand inventory information every one to three minutes, thereby enabling the user to continue the process even if there's a transient outage in the providing system or the supporting architecture. This pattern combined with using a messaging-based integration architecture results in a robust solution.

Additional pattern considerations

When choosing a specific pattern, it's important to take a broader look at the integration context.

Messaging patterns, middleware, and service bus

A service bus refers to asynchronous messaging on a well-equipped messaging platform that includes a service-based channel and endpoint connectors. This is a common, modern way of managing complex integration architecture across distributed systems. The messaging platform is often referred to as the middleware, and it often offers message routing and message orchestration, transformation, and even in some cases, load balancing message error handling, monitoring, logging, and notification services.

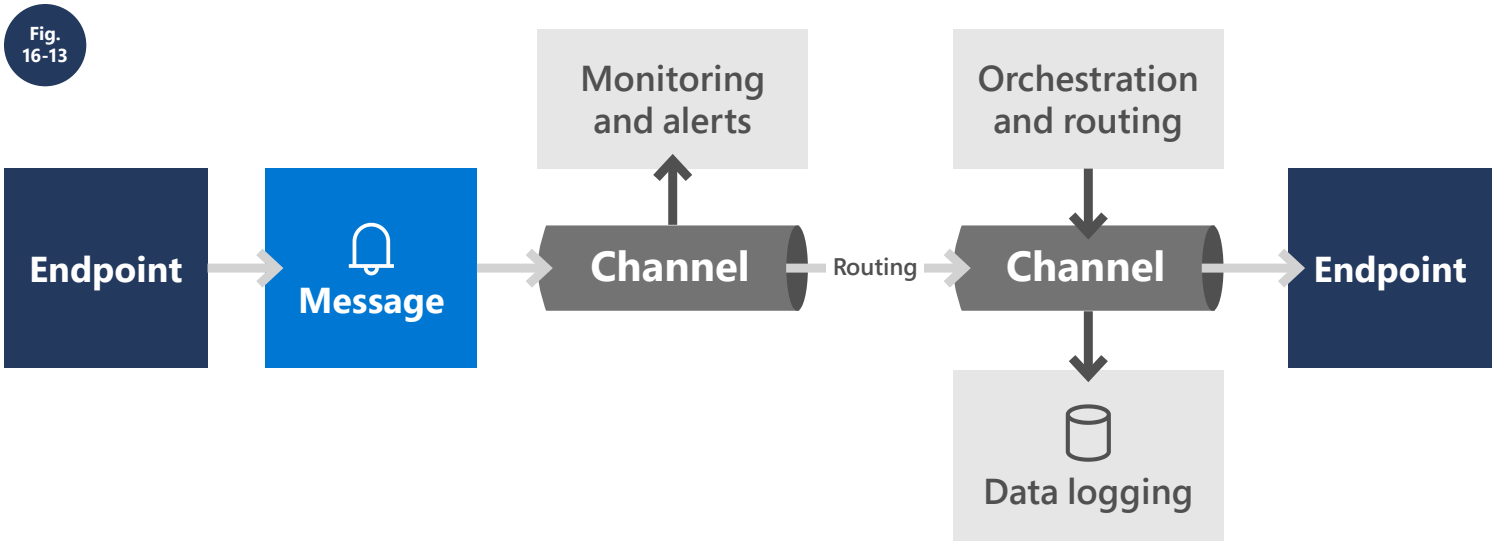
Messages are pieces of information or datasets that can be transferred between systems. Each message can contain instructions such as an event fired or a delete request, or it can contain a payload like a record or dataset. Some messaging platforms allow for prioritizing messages based on type, origin, destination, or payload.

Modern messaging platforms are based on service endpoints, REST, and JSON, but the platforms can offer many additional features. Connections between endpoints are called channels and the message paths are called routing.

The benefit of messaging patterns is that the messaging platform handles the flow of information between systems and detects individual systems or SaaS service endpoint transient outages and handles the queueing of messages to the affected systems until they're back online. **Figure 16-13** shows the typical components of a messaging platform.

The asynchronous nature of the messaging patterns allows for a robust, decoupled integration architecture with less manual intervention and lower maintenance costs.

Fig. 16-13



When implementing a messaging pattern, consider the following:

- **Frequency and volume** It's critical to have a clear picture of the frequency and transactional volume of data that will go through the interface, how that load is distributed over time, and what the requirements for these are in the longer term.
- **Idempotency and out-of-order messages** Mechanisms are available to ensure that the messages are idempotent, for example, using a message ID to check whether the message is a duplicate or an out-of-order message. Services such as Azure Service Bus support this capability and allow you to implement it without the need for additional code.
- **Service-level agreement (SLA) or end-to-end cycle latency** Consider whether there are minimum requirements for how fresh the data in the receiving system must be. For example, orders taken on an e-commerce site should reach the warehouse in less than five minutes.
- **Triggers and actions end-to-end process** It's also important to look at the cross-system business process in the bigger picture:
  - What happens before the integration steps?
  - Is there potential risk of introducing semantic errors earlier in the process?
  - What event triggers the integration?
  - Does the integration trigger action in the receiving system?
  - What business process steps happen after the integration steps are done?
  - If an error is identified, logged, and notified, how is the problem corrected? How often might errors occur, how long

might it take to correct them, and who is responsible for making the corrections?

- **Batching** Batching messages or datasets enables less frequent communication, or "chatter." But it also typically makes messages and payloads bigger.
  - Do the requirements support a batched approach in which datasets are consolidated into each message, or does the integration require individual records or rows?
  - If a batched approach is suitable, how many rows and records are in each message?
  - Are there service protection volume limits to consider?
- **Topology and architecture** Topology and architecture are important considerations for the short and long term as your organization grows internationally or as you migrate systems to the cloud and need to address new challenges and considerations.
  - Are there integration components that will span the cloud and on-premises boundary? Does the messaging service handle that?
  - Where in the world is the data located, and does the inherent latency between geographic regions support the requirements?
  - Does any data exist in countries or regions of the world that have local privacy policies and regulations?
  - Are there service protection limitations in technology—protocol or platform dependencies in any of the integrated systems?
  - Does the data exchanged require transformation, reformatting, or remapping as part of the integration?

For more information, read about "Messaging services on Azure."

We highly recommend that when you design integration, you don't assume that everything will always work as intended. In fact, when working with integration, there are many external factors that we often cannot completely control. Therefore, we need to make sure that our integrations are robust by design, which we discuss in the next section.

Mind the errors

Designing for failure is a crucial component of architecting any solution, including integration components. As part of the integration architecture,

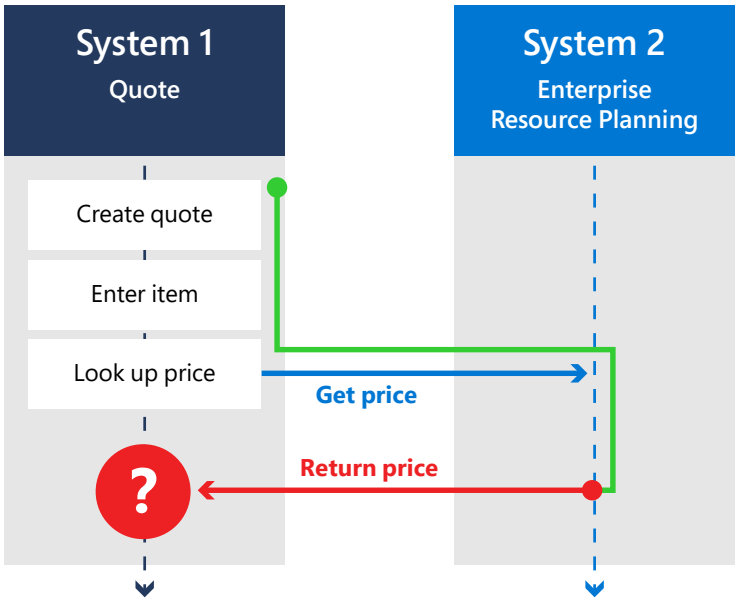
it's important to incorporate the following:

- **Error logging** Select the platform (such as a log file or database) to log the errors.
- **Error monitoring and notifications**
  - Define the process and technical approach to monitor errors.
  - Select an error notification approach, which is the process to notify administrators and other stakeholders if there's an error. Business-critical errors might need a different notification approach than non-critical errors.
- **Business continuity** It's important to plan for business continuity with minimal disruption to business in the event of errors. For more information, see Chapter 20, "Service the solution."

The more integration touchpoints there are, the greater the potential for errors. Therefore, error handling also needs to be planned in accordance with the integration scenario. For example, in the scenario of a synchronous integration pattern, an error might require a complete rollback of the entire process, whereas in an asynchronous data integration scenario, it might be acceptable to fix a data issue just by notifying the administrator. (See **Figure 16-14**.)

Let's now discuss error management for two key patterns of synchronous and asynchronous integration.

Fig. 16-14



As a reminder, in a synchronous integration pattern, each step is dependent on the completion of a preceding step. In the event of an error, consider a retrial or rollback—which option to use is dependent on whether the error is transient or persistent. In the scenario of a transient error, the normal behavior of the application will resume after a few retries. Note that retrial limits should be predefined to avoid a situation in which all resources are blocked. Once the retrial limit has been crossed, the entire process will need to be rolled back, and appropriate error messages should be logged.

For an asynchronous integration, the error

management strategy might differ depending on your business requirements. Consider creating a queue that can continue to validate the message being exchanged and retry until the retrial limit is reached. The message that fails from being exchanged can also be stored in another queue for later intervention.

Transient errors such as network timeouts get fixed after a few retries. However, persistent errors require intervention to be fixed.

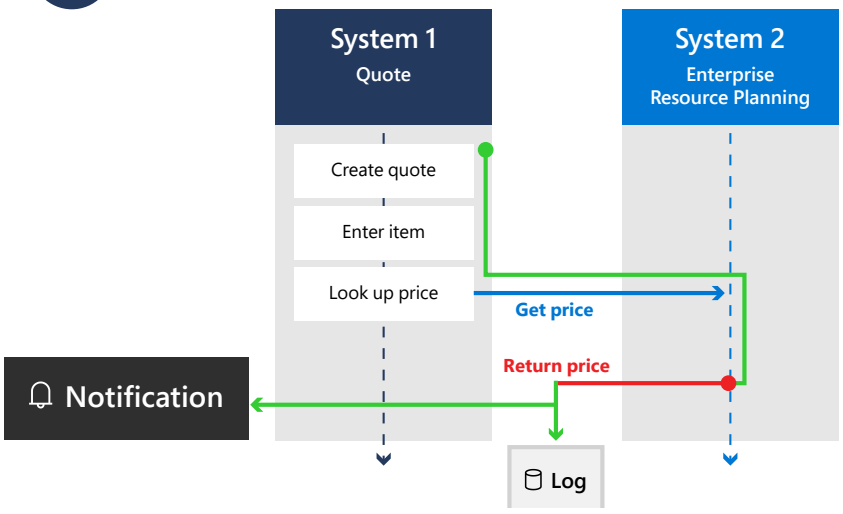
An important consideration as you build your retrial mechanisms is to ensure that your requests are idempotent. This will prevent data duplication issues due to retries.

The following are some of the most common error scenarios in any integration between Business Applications and other applications:

- System becomes unavailable
- Authorization and authentication errors
- Errors caused by platform limits
- Errors caused by service protection limits applied to ensure service levels
  - API limits (limits allowed based on the user license type)
  - Throttling
  - Process errors
- Runtime exceptions

It's critical to plan error logging, monitoring, and communication as part of an overall error-handling approach. Logging errors with the exact timestamp and message details helps the implementation team and application owners troubleshoot the errors. Error tracking also enables the application owners to identify potential areas of improvement in their process by reviewing frequent errors, if any. **Figure 16-15** shows a persistent blocking error. The platform supports logging and notifications so that an administrator can be notified and resolve the issue.

Fig. 16-15



It's also important to identify owners who should receive error details through email as part of regular monitoring. The monitoring and notification approach might differ for business-critical errors versus non-critical errors. In the case of a mission-critical application integration, some organizations might choose to send text notifications to the administrators to speed up the process.

Error messages displayed to end users through the UI should be clear, precise, and short. They should also include information users can act on, such as point of contact information or a link to an FAQ. This makes the user experience a more positive one and should be considered a best practice as part of error handling.

Consider using tools such as Azure Monitor that help maximize your applications' availability to collect, analyze, and act on telemetry.

For more information, visit "Azure Monitor overview."

To summarize, error logging and monitoring have a significant impact on the following key factors and therefore are considered essential:

- Application health including performance and stability
- Ease of troubleshooting and resolving errors
- Minimize business impacts when errors occur
- Increase user satisfaction

For more information about integration patterns and technologies, refer to the "Product-specific guidance" section later in this chapter.

Challenges in integration

The diverse applications used in organizations make up the IT backbone of those companies. Organizations use applications that might be a combination of on-premises, cloud, and third parties. These applications need to communicate with each other for different business needs. Integration challenges during a project implementation can cause delays or cost increases. A key focus of any IT department is to ensure that these

integrations enable business productivity and not become a blocker to business growth.

Integration challenges can be categorized into the following areas.

Business

Each integration scenario has a direct impact on the application you're integrating with and on its users. Any downstream applications might also be indirectly impacted. For integration to be implemented in a successful manner, you should address the following during the Initiate stage:

- **Identification of application owners and stakeholders** Application owners need to identify downstream applications that might be impacted in an integration. However, a common scenario is to bring in these owners after the planning is complete. This often results in misaligned timelines and scope and in turn creates project delays and poor quality. Integration design needs to take every impacted application into consideration.
- **Alignment between business owners** Business stakeholders have different needs for their organization's various applications. Unless there is a collective understanding of integration scenarios and approaches, requirements might be mismatched among the various owners. This in turn often results in delayed timelines, cost overruns, and a lack of accountability. System integrators should consider the following:
  - Identify the key owners and bring them together to walk through the scenarios.
  - Differentiate between process, data, and UI integration to simplify and streamline the integration scope.
  - Outline the impact on business groups affected by the integration.
  - Highlight issues and risks in the absence of following a consistent approach.

  A transparent conversation enables business stakeholders to understand the underlying risks and benefits and thus build a common perspective.
- **Ambiguous and unrealistic expectations** Integration requirements can sometimes be ambiguous or incorrectly

perceived as mandatory. Common examples of the latter are unnecessary synchronous integrations and replicating high transactional data volumes into your business application to make it a reporting application. Such architecture decisions can result in solutions that are neither scalable nor performant. Encouraging and facilitating conversations to help business stakeholders define their requirements clearly and understand the necessary tradeoffs as part of the architecture decisions is a critical step in ensuring a smooth implementation.

- **Lack of business continuity planning** This is an aspect of implementing any IT project that most often is ignored. However, no matter the type of integration—cloud to cloud, on-premises to cloud —every business group must define operating procedures that will be enforced as a part of any availability and disaster recovery situation. As an example, consider a scenario in which the on-premises application of a Dynamics 365 business application is temporarily unavailable. This can impact integration components, resulting in data or process issues.
- **IT-driven requirements and design** Implementation teams sometimes consider integration between systems as a primarily technical exercise and overlook the critical role played by business teams. However, successful integrations between business applications often depend completely on a well-defined cross-system business process, and if the specific process details and broader requirements are not properly defined, the integration project might take much longer or go through several iterations of trial and error. For more information, refer to Chapter 7, “Process-focused solution.”

Technology

Most enterprises have legacy applications with traditional, on-premises architecture. The move to cloud applications requires consideration of the patterns they support and the best practices when planning for integration. A low-code/no-code pattern should now be at the forefront of any integration architecture. Engaging in conversations about not just the current setup but also about future planning for performance, extensibility, and maintenance plays a key role in choosing the right

technology. When choosing the appropriate technology, consider the following.

- **Does one size truly fit all?** Many enterprises have an enterprise architecture approach that might or might not be aligned with the modern approaches for cloud applications. Prior to investing in a specific approach, evaluate whether the existing architecture aligns with cloud patterns. Sometimes, a generic approach is taken—this can result in inefficiencies in integration, unscalable architecture, and poor user experience and adoption. Therefore, it's crucial to consider design paradigms such as the following:
  - Definition of the integration approach based on multiple parameters
  - Benefit of a proof of concept to determine the pros and cons of one approach over another
  - Synchronous versus asynchronous integration
  - Process, UI, and data integration
  - Single record or batch
  - Frequency and direction of the integration
  - Message reliability and speed
  - Data volume
  - Time expectations (some scenarios require a batch integration to be completed during a specific time window)
  - Error management and retrials
- **Will sensitive data be exposed?** System integrators must understand IT and customer concerns around security, especially in the context of integrating on-premises applications with Business Applications. Categorizing security concerns as follows aids in identifying who and what is required to help address them:
  - Access control
  - Data protection
  - Compliance and regulatory requirements
  - Transparency

  For more information, refer to Chapter 12, "Security."
- **Storage costs and platform limits** To ensure service quality and availability, Business Applications and Power Platform enforces entitlement limits. These limits help protect service quality and performance from interference by noisy behavior that can create disruptions. System integrators must incorporate

these limits as part of the overall architecture. If these aren't planned for, the service will be throttled, resulting in failure and errors within the integration layer. Storage costs are also often ignored. Although this might not have an impact initially, over time, it can result in increased storage costs and therefore should be planned for appropriately.

- **Connectivity** Network latency can become a constraint, especially in heavy data-load scenarios. System integrators must ensure that they design payloads accordingly for the efficient use of network resources without compromising performance.
- **Anti-patterns** Implementation architects should follow the best practices for Business Applications. Sometimes these architects don't take cloud patterns sufficiently into account in integration scenarios with on-premises applications, resulting in poor performance. The behaviors leading to such situations are referred to as anti-patterns. Consider the following common anti-patterns:
  - Are there repeated connections between on-premises components and Business Applications that impact performance? If so, consider sending data in batches.
  - Is there latency between a customer's on-premises applications and a Dynamics 365 datacenter? If so, consider using a cloud service such as Power Automate, Azure Functions, or Azure SQL to reduce the latency impact.
  - Is a lot of data being synchronized with Business Applications for reporting purposes? Keep in mind that the database under Business Applications isn't intended to be a data warehouse for all of the organization's data assets. Consider using a dedicated datastore for reporting purposes.
- **Proprietary technology** Customers might be using third-party technology within their IT landscape that doesn't provide interface details or adequate support to enable integration easily. Often such issues are identified either toward the end of design or during the development stage. This causes delays in the project timeline, burdening the customer with time constraints to mitigate such risks. System integrators must highlight such dependencies in the planning stage to ensure adequate support or an alternate approach.
- **Readiness** With the increasing pace of transformations in the

technology world, architects sometime choose an approach due more to its familiarity than its applicability. Customers and system integrators must evaluate whether to request additional resources specialized in the specific technology who will be a better fit for their current and future needs.

Project governance

The initial stage of a project should include a defined project governance model. Integration between on-premises and Business Applications can range from simple to complex, and the lack of well-defined project governance areas results in gaps and issues in the smooth implementation of a project. Following are common project governance concerns specifically for the integration components:

- Has the impact of the integrations been identified for the end user, process, and reporting? This might require planning for change management activities, including communication and training.
- Making a solution performant should be at the forefront of any design decisions made by the implementation team. This applies equally to the integration layer and the integration layer. Is performance testing planned and does it cover integration components? Performance testing is another activity that tends to be considered optional. However, architects and project managers must consider embedding this in their Business Applications implementations. This will help identify any performance bottlenecks prior to deployment for end users.
- Are development and test environments available for all applications for thorough system integration testing? Is a plan for stub-based testing during the unit testing phase required?

Asking these questions during the initial stages of the project enables both the implementation partner and customer to proactively identify and plan for any dependencies and risks.

For more information, refer to Chapter 8, “Project governance.”

Product-specific guidance

The following sections discuss specific integration-related considerations for Finance and Supply Chain Management; specific technologies and platforms for integration from Customer Engagement; and how to integrate between individual business applications.

Finance and Supply Chain Management

Many ways of integrating into and from Finance and Supply Chain Management use an abstraction layer with integration and data migration capabilities that shield users, administrators, and developers from the heavily normalized database. The abstraction layer consists of data entities.

Data entities

In Finance and Supply Chain Management, a data entity encapsulates a business concept, for example, a customer or sales order line, in a format that makes development and integration easier. It's a denormalized view in which each row contains all the data from a main table and its related tables instead of the complex view of the normalized data model behind it.

Said another way, data entities provide conceptual abstraction and encapsulation, a denormalized view of what's in the underlying table schema to represent key data concepts and functionality. There are more than 2,000 out-of-the-box data entities.

Data entities help encompass all the table-related information and make the integration, import, and export of data possible without the need to pay attention to the complex normalization or business logic behind the scenes. For more information, see the "Data entities overview."

Finance and Supply Chain Management integration patterns

The following table is a list of integration patterns with pattern descriptions, pros and cons, and use cases.

The out-of-the-box data entities are general purpose entities built to support a wide variety of features surrounding a business entity. For implementations in which a high-volume, low-latency interface is required, it's recommended to build custom data entities with the necessary targeted and specific features.

You can replicate an out-of-the-box data entity and remove any fields and logic around features not used by your organization to create a highly performant interface.

Pattern	Mechanism	Trigger	Considerations	Use when
OData	Data is exchanged synchronously, invoking actions using RESTful web services and the HTTP protocol stack. All data entities are marked public, and CRUD operations are mapped.	User actions and system events.	**Pros:** It allows developers and service platforms to interact with data by using RESTful web services. It provides a simple and uniform way to share data in a discoverable manner. It enables broad integration across products. **Cons:** The pattern is synchronous and subject to Service Protection limitations and throttling. Cold start might be slow.	Use with messaging services and point-to-point integration. Inbound (Push) or outbound (Pull) of datasets from entities when the volume and payload are relatively low.
Custom Services—SOAP, REST, and JSON	A developer can create external web services by extending the application with X++. Endpoints are deployed for SOAP, REST, and JSON.	User actions and system events.	**Pros:** Easy for developers to add and expose service endpoints to use with integration platforms. **Cons:** Requires ALM and SDLC for coding and deployment of extensions into Finance and Supply Chain Management. The payload is low compared to other patterns.	Used when invoking an action or update, for example, invoicing a sales order or returning a specific value. We recommend using REST Custom Services in general because it's optimized for the web. REST is preferred for high-volume integrations because there's reduced overhead compared to other stacks such as OData.
Consuming web services	A developer can consume external web services by adding a reference in X++.	Scheduled and user initiated. Can wait for off hours or idle time.	**Pros:** Easy for developers to add and expose service endpoints to use with integration platforms. **Cons:** Requires ALM and SDLC for coding and deployment of references into Finance and Supply Chain Management. The payload is low. Risk of hardcoding connection strings and credentials in related code. Maintenance workloads and risks when the service is updated.	Use when consuming services from other SaaS platforms or products, for example, commodity tickers and lookups of real-time values. The recommended pattern is to use Power Automate instead when possible.
Data Management Framework REST Package API (asynchronous, batched, cloud, or on-premises)	The REST API helps integrate by using data packages.	Originating system users or system events.	**Pros:** On-premises and large-volume support. The only interface that supports change tracking. **Cons:** Supports only data packages.	Large-volume integrations. Scheduling and transformations happen outside Finance and Supply Chain Management.

Pattern	Mechanism	Trigger	Considerations	Use when
Recurring Data Management integration REST API (asynchronous, cloud, or on-premises)	With the Data Management REST API, you can schedule the recurring integration of files and packages. Supports SOAP and REST.	Receiving system requests based on the schedule.	**Pros:** On-premises and large-volume support. Supports files and packages. Supports recurrence scheduled in Finance and transformations (XSLT) if the file is in XML. **Cons:** None.	Large-volume integrations of files and packages. Scheduling and transformations happen inside Finance and Supply Chain Management.
Electronic Reporting	A tool that configures formats for incoming and outgoing electronic documents in accordance with the legal requirements of countries or regions.	Data state, system, or user events.	**Pros:** Data extracts and imports are configurable in Finance and Supply Chain Management. It supports several local government formats out of the box. It can be scheduled for recurrence. **Cons:** Comprehensive configuration when used for messaging purposes.	Electronic reporting to regulatory authorities and similar entities.
Excel and Office integration	Microsoft Office integration capabilities enable user productivity.	Data state, system, or user events.	**Pros:** Out-of-the-box integration (export and edit) on almost every screen in the product. **Cons:** Performance decreases with the size of the dataset.	Extracts for ad hoc reporting or calculations. Fast editing of column values and entry of data from manual sources. Import of journal records such as general journals, expenses, and bank statements and similar transactional data.
Business events	Business events provide a mechanism that lets external systems, Power Automate, and Azure messaging services receive notifications from Finance and Supply Chain Management.	User or system events.	**Pros:** Provides events that can be captured by Power Automate, Logic Apps, and Azure Event Grid. **Cons:** Extensions are needed to add custom events.	To integrate with Azure Event Grid, Power Automate, or Logic Apps. To notify of events inherently driven by a single data entity, for example, an update of a document or a pass or fail of a quality order.

Pattern	Mechanism	Trigger	Considerations	Use when
Embedded Power Apps (UI)	Finance and Supply Chain Management support integration with Power Apps. Canvas apps can be embedded into the Finance and Supply Chain Management UI to augment the product's functionality seamlessly with Dataverse.	Users.	**Pros:** Seamlessly integrates information from Dataverse without integrating the backend. Opens opportunities for low-code/no-code options directly into the Finance and Supply Chain Management UI without the need for updates and compatibility. **Cons:** Power Apps and related artifacts aren't deployed with the build and must be configured in the environment directly. Separate ALM stories for Power Apps.	Whenever the required data exists in Dataverse and is loosely coupled with Finance and Supply Chain Management. Using an app embedded in the UI provides a seamless experience for users.
Embedded Power BI (UI)	Seamlessly integrates Power BI reports, dashboards, and visuals with information from Dataverse or any other source, without integrating the backend.	Users.	**Pros:** By using graphics and visuals supported by Power BI to present data from any source, workspaces can provide highly visual and interactive experiences for users without leaving the Finance and Supply Chain Management UI. **Cons:** Power BI artifacts are not deployed with the build and must be configured within the environment directly. Separate ALM stories for Power BI components and Finance and Supply Chain Management Apps.	Whenever reports, dashboards, or visuals exist in Power BI.
IFrame (UI)	The Website Host control enables developers to embed third-party apps directly into Finance and Supply Chain Management inside an IFrame.	Users.	**Pros:** Seamlessly integrates UI from other systems or apps without integrating the backend. Information goes directly into the Finance and Supply Chain Management UI without the need for updates or compatibility. **Cons:** External systems might have separate lifecycles, and updates to those might affect user experience with little notice.	When information from loosely coupled systems can be displayed within the Finance and Supply Chain Management UI. The experience is enhanced if the external system supports single sign-on (SSO) and deep linking.

Priority-based throttling

Service protection is important for ensuring system responsiveness, availability, and performance. In Finance and Supply Chain Management apps, service protection is enforced by throttling. Throttling affects OData and custom service pattern integrations only. Administrators can configure priorities for external services (registered applications) directly in the application so that lower priority integrations are throttled before high-priority integrations.

For more information, read about "Priority-based throttling."

Requests that are throttled receive a response containing a retry-after value, indicating when the integrating application can attempt a retry.

Microsoft Dynamics Lifecycle Services (LCS) monitors throttling activity.

Customer Engagement

In this section we talk about frameworks and platforms to use when integrating from Customer Engagement.

IFrames

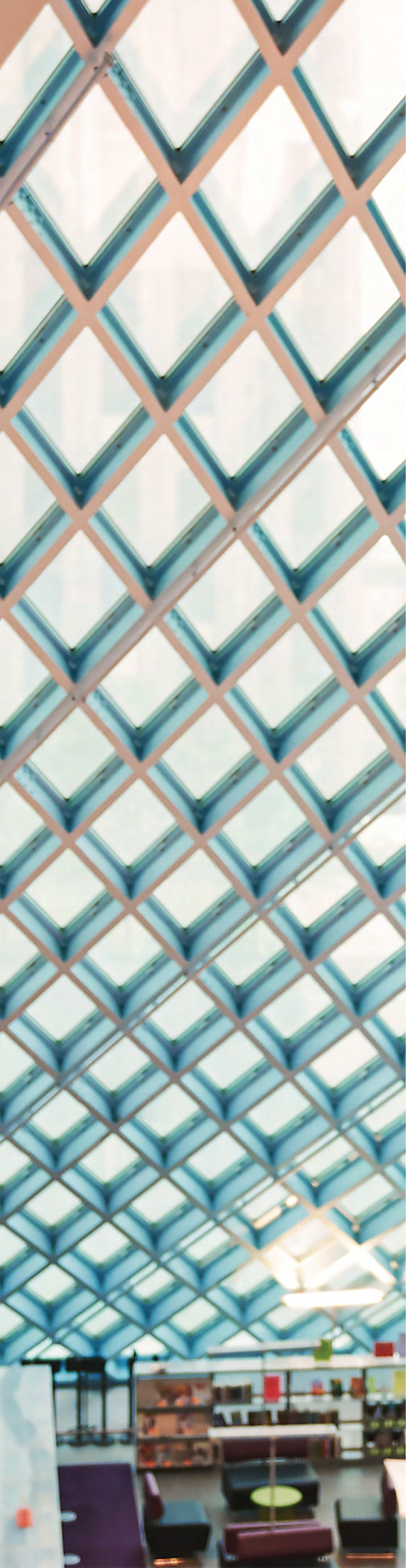
IFrame is a popular approach commonly used for hosting external URL-based applications. Consider using the Restrict cross-frame scripting options to ensure security.

Power Apps component framework

Power Apps component framework is a modern approach for creating rich visual components that allow enhancement of the user interface of model-driven apps. It leverages client frameworks such as React, and enhancements can be reused across Business Applications. It can also pull data from an external application and display the data in a visual form.

Canvas apps

Canvas apps is a cloud service that enables citizen developers to easily build business apps without the need to write any code. These apps can use connectors from other cloud services and be



embedded in model-driven apps to present data from other applications on the user interface.

HTML web resources

A precursor to Power Apps component framework, HTML web resources were commonly used to render data in a visual form, providing designers with more flexibility. They can be used to pull data into external applications using the available endpoints.

Dynamics 365 Channel Integration Framework

Dynamics 365 Channel Integration Framework hosts the widgets and triggers the events for scenarios such as telephony and chat integration. There are multiple versions of Channel Integration Framework for different scenarios, including multi-session and single-session needs. Channel Integration Framework also integrates channels such as voice, chat, and messaging.

Virtual tables

Virtual tables pull data on demand from external data sources. This approach is implemented as tables within the Dataverse layer but doesn't replicate the data because the data is pulled real time on demand. For more information, read about the limitations of virtual tables.

Azure Data Factory

Azure Data Factory provides a code-free, serverless data integration service to seamlessly orchestrate ETL (extract, transform, and load) and ELT (extract, load, and transform) operations with disparate big data sources. It can ingest data from Business Applications using the web API endpoints. We recommend that you consider platform and service protection limits as well as Azure costs.

Webhooks

Commonly used for near real-time integration scenarios, webhooks can be invoked to call an external application upon the trigger of a server event. When choosing between the webhooks model and the Azure Service Bus integration, consider the following:

- Azure Service Bus works for high-scale processing and provides a full queueing mechanism if Dataverse pushes many events.

- Webhooks enable synchronous and asynchronous steps, whereas Azure Service Bus enables only asynchronous steps.
- Both webhooks and Azure Service Bus can be invoked from Power Automate or plug-ins.

For more information, visit "Use Webhooks to create external handlers for server events."

Azure Functions

Azure Functions uses serverless architecture and can be used to extend business logic, including calling external applications. It runs in Azure and operates at scale. Azure Functions can be called through Power Automate and Azure Logic Apps. For more information, read about "Azure Functions" and "Using Azure Functions in Power Apps."

Integrate business applications

In this section we look at the integration between two or more business applications.

Some of the patterns mentioned earlier in this chapter are specific to when we want to integrate one of the business applications into a third-party or custom-built system.

Four options are available for integrating the Customer Engagement apps suite and Finance and Supply Chain Management apps. Note that we're actually integrating Finance and Supply Chain Management into Dataverse. These options have distinct characteristics, different scenarios, and different pros and cons.

Virtual tables

The virtual table option enables you to connect Dataverse to Finance and Supply Chain Management Apps entities as virtual tables that offer the same full CRUD (create, retrieve [or read], update, delete) capabilities as the entity endpoint in the app. A benefit is that we can access the data in Finance and Supply Chain Management Apps in a secure and consistent way that looks and behaves the same as any other table or construct in Dataverse. We can also use Power Automate to connect to almost anything.

Another benefit is that the data doesn't have to live in both the transactional database underneath Finance and Supply Chain Management Apps and Dataverse but is still seamlessly available as tables and rows in Dataverse.

Because Dataverse is cloud based, this direct integration option isn't available for on-premises implementations of Finance and Supply Chain Management Apps. You can use the Data Management REST API instead.

For more information, read the "Finance and Operations virtual tables FAQ."

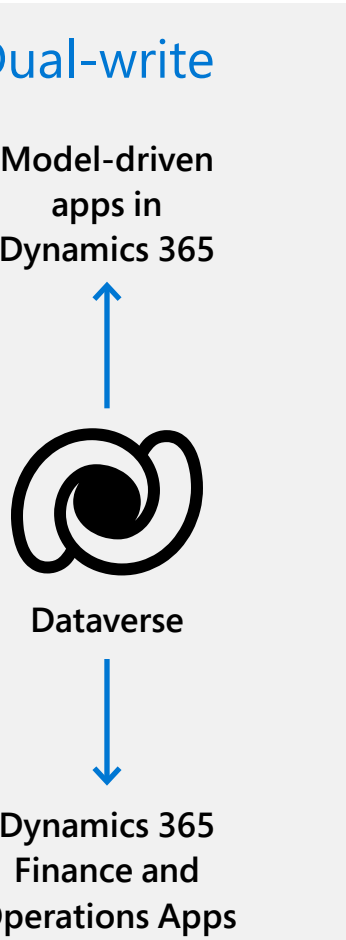
Dual-write

The second option is dual-write integration, shown in **Figure 16-16**. Dual-write also provides synchronous, real-time integration between Finance and Supply Chain Management Apps and applications in Dataverse. Dual-write even offers offline capabilities.

A common scenario for the dual-write option is one in which both Customer Engagement and Finance and Supply Chain Management Apps are working on data that is fundamentally the same, for example, customer and account, product and item, and projects.

The data is shared to provide seamless integrated experiences, for example, for product mastering, prospect to cash, and project to cash across systems.

Fig. 16-16



The benefit of using the dual-write approach is that the two applications share the same dataset, so changes in one system are seamlessly and instantly reflected in the other one.

Another benefit is that in some cases the functionality is out of the box and configurable with minimal effort, but it's also extendable by a developer. Additionally, the full set of capabilities of Power Platform is available for the shared datasets.

For more information, read about dual-write.

Dynamics 365 configurable integrations

The third option is configurable integrations. These provide a starting point for integration between specific Customer Engagement Apps and Finance and Supply Chain Management Apps through integration artifacts that enable integration between Customer Engagement Apps in Dataverse and Finance and Supply Chain Management Apps. Configurable integrations are built on data entity integration to

Dataverse and enable an administrator to relatively quickly set up the supported predefined integration scenarios.

The supported scenarios are as follows:

- **Prospect to cash integration** The Prospect to cash integration is a process and data integration between Sales and Finance and Supply Chain Management Apps. The process integration enables users to perform sales and marketing activities in Sales, and other users handle order fulfillment and billing in Finance and Supply Chain Management. For more information, read "Prospect to cash."
- **Field Service integration** Field Service integration adds integration points between the Field Service App and Finance and Supply Chain Management Apps to enable process integration for work orders and projects. For more information, read the "Integration with Microsoft Dynamics 365 Field Service overview."
- **Project Service Automation** Similarly, the configurable integrations provide integration between Dynamics 365 Project Operations and Dynamics 365 Project Service Automation by synchronizing project essentials such as contracts, tasks, milestones, time, fees, expense forecasts, and actuals. For more information, see the "Project Service Automation overview."

Other integration patterns

The fourth option for integrating the Customer Engagement apps suite and Finance and Supply Chain Management is either Application Connector or point-to-point OData integration patterns with data entities. For more information, read about the "Application Connector."

Conclusion

Implementing Business Applications is often a building block into an organization's larger solution landscape. In that case, the organization can benefit from automation to gain streamlined and effective cross-system processes and avoid manual errors.

Integration is also key in closing the digital feedback loop and making all the organization's data available, not just for reporting and visibility

but also to make it accessible to users in the organization. With Azure, Power Platform, and Dataverse you're well on your cross-system journey toward reporting, analytics, and AI-assisted forecasts and predictive analysis so that the right information is readily available for the right people at the right time.

With Microsoft cloud technology and Power Platform, low-code/no-code options, and the citizen developer concept, once the systems are integrated and the data is available in Dataverse, that journey can start without requiring massive investments upfront.

In this chapter, we highlighted the path from business goals, high-level architecture, design, and pattern considerations—the work that should be done when integrating systems and processes—toward a successful solution. We introduced integration patterns and discussed the dos and don'ts and common challenges that could unnecessarily prolong the start of the journey if not addressed properly, specifically about business ownership, technology, and implementation project governance.

Finally, we referenced some of the technologies that people are using across the globe to successfully integrate their Business Applications into the solution landscape.

References

REST vs CRUD: Explaining REST & CRUD Operations

Messaging Integration Patterns

Open Data Protocol (OData)

Consume external web services

Custom service development

Data management package REST API

Recurring integrations

Electronic messaging

Office integration overview

Business events overview

Embed canvas apps from Power Apps

Analytical Workspaces (using Power BI Embedded)

What's new and changed in Platform update 31 for Finance and Operations apps (January 2020)

Checklist

Define business goals

- [ ] Document and define goals and expected benefits of integrations being implemented in a business-centric way.
- [ ] Align the planned integration's purpose with short- and long-term organization goals.
- [ ] Ensure the overview of the integration architecture, systems, and integration points is clear and understandable.
- [ ] Ensure that stakeholders have a shared understanding of the purpose and scope of the integrations that are being implemented.

Choose a platform

- [ ] Ensure the organization understands the concept of cloud versus on-premises platforms and the boundary between them.
- [ ] Plan to use either an integration middleware or messaging service.
- [ ] Ensure the integration architecture, platform, or middleware supports the expectations for monitoring, audit, notifications, and alerts.
- [ ] Ensure the integration architecture supports the expected level of security, availability, and disaster recovery.
- [ ] Ensure all components of the integration architecture support ALM and version control.

Choose a design

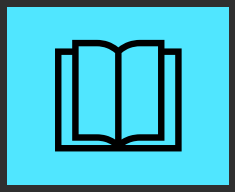
- [ ] Align the designs of each integration with the overall integration architecture.
- [ ] Avoid unnecessary replication and transmission of data into the solution in integration designs.
- [ ] Clearly state the options and benefits of each of the following: UI, data, process integration, and Dataverse.

Choose a pattern

- [ ] Design integrations to favor robust, asynchronous messaging-based patterns.
- [ ] Align patterns used for each integration with expectations for volumes, frequency, and service protection limitations.
- [ ] Set realistic estimates of the operating costs for services, platforms, and storage involved and be aware of how scaling affects them in the future.

Project governance

- [ ] Plan each integration for user and performance testing under realistic loads, as well as the end-to-end process leading up to the integration, across the system boundary, and after the point of integration.
- [ ] Plan for testing the end-to-end process patterns used for each integration in line with recommendations for volumes, frequency, and service protection limitations.
- [ ] Have change management activities related to integrations that reflect and support overall business goals.
- [ ] Complete the impact analysis on upstream and downstream processes.



Case study

Public sector infrastructure organization learns how to choose the right solution for integration

A public sector infrastructure service organization was implementing the Dynamics 365 Customer Service app and required integration with a public-facing website, several on-premises apps, and Office 365 services such as SharePoint Online. One of the implementation objectives was to gather insights from the data that was scattered across different business applications.

The organization was working with an IT vendor that had several years of experience building complex integrations using technologies such as IBM MQ and Microsoft BizTalk.

They decided to keep a uniform integration architecture and proceeded with a microservices-based architecture to build small, reusable components using service endpoints to gain the following benefits:

- Faster and independent development, deployment, and release cycles
- Simplicity of architecture to allow decoupled and reusable components
- Fault isolation

The organization chose Azure API Management to abstract their APIs and implement a secure integration layer.

As the team started building and testing the initial components, they identified some challenges due to the architecture:

- They experienced slow performance with batch-integration scenarios because they called the services as they would have in a point-to-point integration.
- They couldn't use standard functionalities that would have been available with out-of-the-box approaches such as SharePoint Online integration with Power Platform.
- For an aggregated view, they decided to replicate all data into the Dynamics 365 Customer Service app, which led to additional storage costs.
- They encountered throttling and API limits issues, which prevented successful completion of the integration.

At this stage, the organization decided to re-evaluate their standardized architecture principle and redefine their integration strategy. They divided their needs into two categories: batch and point-to-point synchronizations. They also adopted a "Configure-before-you-customize" approach.

With a reasonable mapping of different integration needs to the patterns, they were able to redefine a suitable integration architecture for these components.

For point-to-point integrations, the organization used the previously defined microservices-based architecture and incorporated the necessary changes for aligning with Power Platform API limits.

For batch synchronizations, the organization designed the architecture to manage the overall ETL processes with the following approach:

- They extracted the data from their source application by creating SQL Server Integration Services packages and storing that data in a staging database to perform all transformations.
- They hosted their staging environment in Azure SQL, within the same datacenter as the Customer Service app.

- They sent that data as batch data to the Customer Service app to reduce the number of individual connections.
- They considered API and throttling limits and built retries as part of the design.

For an aggregated view of the data, they realigned their approach to use Azure Data Lake, as Microsoft recommended. The out-of-the-box Export to Azure Data Lake feature required simple configurations to export data from the Customer Service app into Data Lake and didn't involve writing any additional code.

These changes in their overall approach provided significant benefits:

- Their batch synchronization performance improved significantly, and they cut the duration of the daily sync by more than 70 percent.
- Their "Configure-before-you-customize" approach let them choose out-of-the-box options to reduce development and maintenance costs.
- The retry mechanism, combined with monitoring, ensured that any failure scenarios were well handled and negative impacts on end users were avoided.

Even though the architecture realignment reset the organization's timeline, the benefits outweighed the risk because they were able to align their design with product features and the recommended practices. The organization already had a change-management plan in place to take care of stakeholder communication and alignment, and to plan trainings for all affected teams in a timely manner.

17

Guide

A performing solution, beyond infrastructure

Introduction

When a new Dynamics 365 implementation is delivered, users typically expect an improvement in system performance.

The following are performance factors to consider when designing solutions:

- Solution performance is critical for user adoption, customer experience, and project success and necessary to enable businesses to achieve their goals.
- Clear goals and realistic expectations are vital to developing a solution that performs well.
- Scalable solutions begin with the correct use of the right software.
- Customizations increase the risk of performance issues, but these risks can be mitigated with the right mindset and planning.
- Performance testing needs to be realistic to be meaningful.
- Performance issues are complex to resolve and therefore are better avoided than fixed.
- Ensuring performance is an iterative process that involves incremental improvements throughout the solution lifecycle.

Good performance is often assumed as a given, a default experience. The reality is that although Dynamics 365 products are scalable and powerful, various factors are involved in achieving a high-performance solution. These include defining and agreeing to performance metrics, testing throughout the phases of a project, and taking design and build considerations into account, particularly for customizations and integrations.

In this chapter, we explore various aspects of performance and how they should be addressed throughout the stages of implementation. We discuss why performance is directly related to the success of a project and why it must be prioritized early. We also discuss how to align expectations with business users to enable meaningful performance discussions so that challenges are identified and translated into design requirements.

Finally, we cover performance testing strategies to protect organizations from common performance risks and anti-patterns as well as how to approach resolving any issues that do occur.

Why performance matters

Performance can pose challenges as projects approach go live. These challenges are typically due to a failure to properly prioritize performance in earlier phases. Let's back up a bit and consider first why good performance is important.

Let's consider city planning—specifically, the importance of street design to ensure livability. Streets enable residents and visitors to get into and out of a city. A city could plan a basic two-lane road to accomplish this goal. However, what happens when 1,000 cars are on that two-lane road at the same time? What happens if an accident occurs? What happens if you add 10,000 cars? Yes, eventually drivers will get to their destinations, but it likely won't be in the desired amount of time. These are the types of questions to ask when looking at performance.

People expect superior response times and a smooth experience from the organizations they choose to provide products and services. If customers don't get this, they tend to look elsewhere. Let's go back to our street design. Because of poor street design, people might decide not to live in or visit that city. Residents might even elect different leadership because of their poor experience. The takeaway is that system performance affects people in numerous ways and can have serious consequences.

Customer experience is key

A poorly performing application solution can definitely have a negative effect on your organization. Microsoft Dynamics 365 products support a wide range of business functions related to customer interaction and are an opportunity to either delight or disappoint customers.

Customers might use integrated applications with a direct dependency on Dynamics 365 apps, for example, through a sales website. In such a scenario, poor performance can quickly turn into lost sales as customers abandon the website to shop elsewhere. No one enjoys a frustrating and slow sales experience, and customers can quickly switch to a different business to purchase goods and services.

Often, though, customers interact with systems indirectly through an agent within an organization, for example, when a customer calls a call center to discuss a faulty product or delayed order shipment. These types of interactions can be challenging for a business to resolve in a positive way and are only made more challenging if an agent cannot access information in a timely manner. This illustrates again that system performance is key to ensure a positive customer experience, regardless of whether the customer interacts with the system directly or indirectly.

System success

As businesses evolve and optimize, the ambition remains to achieve more with less. The need to sell more products or service more customers is always accompanied by the need to reduce the expenditure of money, effort, people, and time, at all levels of the organization.

This constant need to achieve more with less creates pressure on employees to work in the most efficient way possible. Time spent waiting for a system to perform an action is wasted time, and employees who rely on these systems to do their jobs quickly realize this.

Poor performance can have significant effects on a project. It's usually not well tolerated by business users and can be detrimental in terms of user adoption and system reputation.

User adoption

User adoption is a critical factor in the success of any software project. Any business case (and projected return on investment) depends on the system being used as intended. Poor performance directly drives user dissatisfaction and can make user adoption incredibly challenging.

Users are keen to adopt systems that increase their productivity, which essentially means minimizing wasted time. Business users achieve their goals when solution performance is optimal and as expected. A poorly performing system wastes users' time and therefore reduces productivity.

If the impact of poor performance on users' time is significant enough,

they'll likely find more efficient ways of working on their own. These workarounds ultimately serve the interests of specific users rather than the business and often result in data stored off-system (for example, in spreadsheets). This might eventually lead to a situation in which the system no longer serves its purpose, doesn't deliver a return on investment, and ultimately fails.

System reputation

Even before go live, performance can help or hinder user adoption. During the development phase, the application typically is presented to a set of key users in different areas of the business to collect feedback. These users then talk to colleagues about the implementation. In this way, the reputation of the application spreads throughout the business long before most users touch the system. Keep in mind that performance impressions tend to spread quickly. For example, if a demonstration goes well, a wave of excitement might flow throughout the company. This positivity can help increase user adoption because of the anticipated improvement in productivity.

Performance issues discovered in the buildup to go live or shortly afterward cause a great deal of stress for the project team, which usually works long hours to ensure a timely delivery of a functional system. The realization that the system doesn't perform well can quickly force a change in priorities and derail progress. These issues are often visible at the most senior levels and tend to escalate quickly.

But it's in the cloud

Having established that performance matters, surely you can just deploy the fastest computers available to run the solution, right? Well, not quite.

Dynamics 365 is a cloud-based software as a service (SaaS) solution. Cloud solutions entail different challenges from on-premises deployments. For example, several years ago, Dynamics 365 deployments were commonly hosted on infrastructure physically located close to or even in the same office or warehouse as users. Today, single deployments are used by users across the world on a range of devices with

different network infrastructures. Project teams therefore need to build solutions that accommodate users who have a variety of hardware, increased network latency, and a range of network quality.

The SaaS nature of an application means that the infrastructure on which the solution runs is housed and maintained by Microsoft. Other than a few predefined configuration options, the infrastructure is beyond the scope of a customer's control, which offers benefits in setup cost and ongoing maintenance responsibilities but does limit the options to change performance.

Having said that, the performance of an application extends beyond the infrastructure. Although a correlation exists between the performance of software and the quality of its underlying infrastructure, the on-premises mindset of fixing software problems with additional hardware is expensive and usually only a temporary fix.

Using our street design example again, adding a privately operated toll road or highway doesn't necessarily mean drivers will arrive at their destinations any faster.

We look at more examples later in the chapter, but for now it's important to be clear that most performance issues are best solved by correct implementation decisions and not by adding hardware. Moreover, it's crucial to acknowledge that performance is not guaranteed simply because the software is running in the cloud. It's still the responsibility of the project team to deliver a well-performing solution.

Prioritize performance

Given that performance is so important to users, customers, and ultimately the success of the overall system, let's look at how performance relates to project delivery.

Performance doesn't happen unless you make it happen

Performance issues in Dynamics 365 projects are commonly analyzed, understood, and fixed at a solution level, often against tight deadlines and high visibility. It's fair to say that the most common cause of unexpected performance issues is a failure to properly prioritize performance from the beginning of a project.

Performance in any software project is not guaranteed. It requires careful planning, expectation management, open communication, and dedicated time and resources. This is especially true for projects that feature integrations, data migrations, and extensions on top of the base product. These are all common in Dynamics 365 implementations. Anything added to the core Dynamics 365 app has the potential to negatively impact performance, so the impact needs to be understood and managed to ensure performance remains acceptable.

Performance influences decisions in many areas of the project, including the following:

Data strategy

- Is the volume of data stored likely to cause performance issues for users?

Integration strategy

- Are real-time integrations feasible given the performance expectations of the users?
- Can overnight batch integrations complete within a given timeframe?

Data modeling

- Do we need to denormalize for performance reasons?

Security modeling

- Will our security model work at scale?
- Are there bottlenecks?

Environment strategy

- Is a performance test environment available?
- Is our performance test environment representative of production?
- Have we budgeted for performance testing?

Design and implementation of customizations

- Are the developers building to meet specific performance goals?
- Do user expectations align with what's technically feasible?
- Do we need to use patterns (such as asynchronous or synchronous) or batch jobs (such as background processes)?

Testing design and approach

- What's acceptable—what's considered a pass or fail?
- Is our testing representative of user behavior?
- Are we testing based only on today's business requirements or keeping future expansions in mind?

User acceptance and adoption

- Is performance measurable?
- Are user expectations realistic?

These aren't the types of questions the delivery team should be asking when a performance issue surfaces, especially when approaching the go-live deadline. A successful project will have answers to these questions early on. At the least, the delivery team should identify risks and seek possible resolutions in the early stages of delivery. This might lead to proof-of-concept work to test performance.

Let's put this into the context of our street design scenario and consider the questions that need to be asked and answered to maximize the design. For instance, how many residents currently live in the city? How much traffic can we expect to be on the roads? What's the projected population growth and how long will the design support it? Will traffic lights be installed? If so, how many and how will that affect traffic? What's the speed limit and are there risks associated with that limit? Each of these questions helps determine the best street design before we even start the project.

Resources

Considering performance from the early stages also ensures that the correct expectations are set in terms of time, money, effort, and people. For example, for performance testing, a dedicated performance test environment is needed as well as the people to do the testing. Business stakeholders might need additional time with the delivery team to understand, agree with, and document performance requirements. It might even be necessary to allocate more development time and resources for code optimization.

The most successful projects from a performance standpoint dedicate workers to performance-related activities. Performance testing is a subject on its own, and expertise is required to design test cases and execute test runs. Furthermore, a single owner within the project should be identified to drive performance planning and remediations across the development teams during all phases of the project.

Fig. 17-1

Prevention is better than cure

A common approach with Dynamics 365 projects is to implement the system, get it to a point where it's functionally correct, and only then deal with the inevitable performance challenges (see **Figure 17-1**). Although this approach might seem logical under the pressure of delivery, the inevitable performance issues place additional stress on the team during the most difficult parts of the project and add work during the buildup to go live.

Dealing with performance issues after functional testing is completed can create additional project risk because the changes needed to resolve them are typically significant and therefore come with a high risk of introducing bugs. This essentially invalidates any prior functional testing. A regression testing cycle is then required to ensure

performance fixes haven't broken the previously tested functionality. This wastes time at a critical stage in the project.

User confidence

Performance is always linked to perception. It's important to be aware of user feedback during the implementation of the project because it can affect users' engagement during testing and ultimately help or hinder user adoption at launch. Projects that prioritize performance early on tend to present better-performing solutions during implementation. This early planning helps reassure users that they'll receive a solution that enables them to become more productive—and this leads to better engagement and adoption overall.

Why not prioritize performance?

With all this in mind, it's perhaps surprising that many Dynamics 365 implementations don't prioritize performance. There are several reasons for this. In some cases, people lack experience with the specific products. Performance issues can be complex, and it's often difficult to foresee them without extensive knowledge of the platforms and associated patterns and anti-patterns gained over years of experience.

Sometimes, however, projects intentionally don't include performance testing in an attempt to save on budget. For some projects, performance activities are deprioritized because of timing constraints. However, these are shortsighted approaches because they're an opportunity for performance issues to surface and risk the success of the project. The best way to avoid performance problems is to be proactive and seek out and address risks early to minimize their impact. It's a more productive approach than reactively responding to problems as they occur.

Establish requirements

Acceptable performance is the goal of every project, but the definition of "acceptable" is often vague, if defined at all. To successfully deliver acceptable performance, we need to be clear on what that means and then be able to track progress against our performance goals.

When we look at the requirements for our street design scenario, we know that drivers want to go from point A to point B quickly and safely. However, each driver might have a different idea of what quickly and safely means. To get somewhere safely, a driver might need to drive slower to handle unexpected situations. But then some drivers might become upset because it takes too long to reach their destination. If you create a goal for drivers to get from point A to point B within X amount of time and that time increases, it might indicate that you need to change the design.

In the next section, we discuss how to define performance goals that are achievable and acceptable to users.

Why do we need performance requirements?

If you ask a user for their performance requirements, you'll likely get a response that the system needs to be fast—and not much else. However, "fast" is an ambiguous term; people have different expectations of what that looks like, which makes understanding whether you've achieved acceptable performance nearly impossible.

A better approach is to establish a common understanding across the implementation team and stakeholders on performance, which enables us to:

- Align expectations of the definition of "acceptable."
- Understand the business impact of not achieving acceptable.
- Understand constraints during the design process.
- Provide clear instructions to developers during implementation.
- Test in a meaningful way with behavior models representative of expected usage.
- Determine whether acceptable has been achieved.
- Assess gaps between what's been achieved and what's acceptable.

This approach is the same for other system requirements. It's vital that performance be considered like other requirements gathered in the initial stages of implementation.

More specifically for performance, it's also important to know when to stop optimizing. For example, developers might apply performance tweaks to a piece of code, optimizing with no goal in mind. Any optimization of tested code comes with some risk of regression issues; therefore, this practice should be performed as infrequently as possible. However, without a clear understanding of when enough is enough, it's difficult to gauge what level of optimization is sufficient and when further optimization is unnecessary.

Work with users

Any performance-related discussion should include representatives of the system users. They often give meaningful insight into what is and isn't acceptable. Decisions made without input from users risk rejection in the later stages of the project, which is what we want to avoid.

This doesn't mean users are solely responsible for deciding performance requirements. It's important to be clear that performance comes with an associated cost, and business stakeholders need to assess requests coming from users within the context of the wider project. Aggressive performance requirements might be achievable, but they require additional development, testing effort, and people. With this in mind, it's important to understand the underlying need for each performance requirement and for business stakeholders to be prepared to consider a compromise where it makes sense to do so. Performance for the sake of performance is expensive and unnecessary. Communicate this to users and take a pragmatic approach to focus on what specific performance requirements are important.

Identify performance-critical tasks

Users will use the system under development to perform their own set of tasks. Some of these tasks will be more performance critical than others. For example, consider a call center agent attempting to access customer data at the start of a telephone call with a customer. It's critical that the agent is able to access the data, and every second counts. Time spent waiting for a form to load is a waste of time for the agent and the customer.

In contrast, a user preparing month- or year-end financial reports might expect to wait a little while for actions to complete. Performance is still important, but for activities like this that occur infrequently, the business case for investing time, effort, and money to improve performance is more difficult to justify.

For example, business users might offer the justifications shown in **Figure 17-2** when asking for a performance improvement.

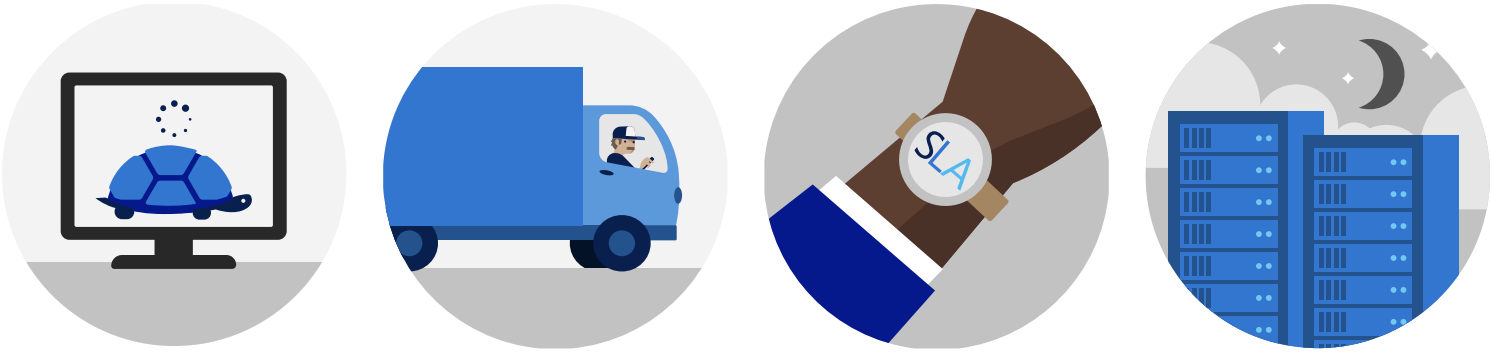
Spend time with users to understand the activities for which performance plays a critical role. Agree on these areas with project stakeholders and then focus performance-related work on these activities to maximize the value of the efforts. Consider performance testing for each system area, including the following:

- Functional processes
- Background operations (for example, batch and workflows)
- Integrations
- Data migration
- Reporting and analytics

Understand user expectations

For each critical task you identify, have discussions with users about what they deem to be reasonable performance. It's important to discuss each task separately and focus on the specific task when discussing acceptability. This leads to different requirements for different tasks, which is a more granular, actionable, and sensible outcome than having broad, systemwide requirements.

Fig. 17-2



"I need to load this record quickly for a customer on the phone; otherwise, I might lose the sale."

"I need to be able to load the products into the delivery truck to meet my shipment schedule."

"I have to be able to do this within a few minutes; we have a service-level agreement to meet."

"This overnight process needs to happen within the time window; otherwise, the users won't be able to work."

Also consider that a user with exclusive access to a system is likely to have a different experience than someone who uses the system only during peak hours, along with thousands of other concurrent users, or while an intensive background process runs. We recommend that you map out predicted system usage throughout the day, week, and month to understand the volumes of users, data, and actions in these time periods to better locate any potential performance issues.

The usual measurement of performance is time taken, so requirements should be shaped around what is an acceptable time to perform a specific action, in a way that can be measured. However, it's often easier to ask users, "What does unacceptable performance look like?" and work backwards from there. This approach tends to lead to more realistic requirements, although they might be unnecessarily aggressive. Take the time to understand the business impact and risk of unacceptable performance in these scenarios.

Anticipate growth

When discussing performance requirements, consider the roadmap of the business as well as design requirements for an increase in demand on the system. Data and user volumes play an important part in how a system performs, so it's important to anticipate any growth expected in the near future and design for that rather than focus on the current requirements. Along with future growth, also plan for seasonality in the system load, for example, during the end of the year.

Document requirements

It's crucial that performance requirements be documented (the same as for other system requirements). Documenting requirements provides visibility to all parties about the expectations of the software and provides clear goals for the implementation team. Additionally, any performance risks identified during discussions with users should be documented in the project risk register to ensure they're tracked and mitigated as much as possible.

Assess feasibility

The implementation team should review performance requirements

along with other system requirements as early as possible to assess the impact to the organization. Some performance requirements might have a significant impact on design decisions, so you want to be aware of these as soon as possible.

Occasionally there might be doubt about whether a particular requirement is even achievable with the available software. In this situation, it's advisable to run a small proof-of-concept pilot project to understand feasibility as early as possible, thereby minimizing the risk of accepting unachievable requirements. Discuss requirements deemed unachievable with the business—it's generally a more positive discussion during the early stages of the project than during user acceptance testing (UAT) or immediately before going live.

With clear performance criteria established, accepted, and documented, the project team can move confidently on to implementing a system that will perform within expectations.

Design for performance

A common cause of performance challenges in Dynamics 365 projects involves incorrect use of the applications—in other words, uses that aren't aligned with the intentions of the software or are due to poor solution design. Poor decision making during the design phases of the project can open the door to performance challenges further into the project.

Looking at this from the perspective of our street design discussion, there are different ways to tackle a situation like traffic. We could design our roads to be 10 lanes; that would handle a lot of traffic. But this creates other complications. Do we have land that will support that infrastructure? What are the relative costs associated with that design? How easy will it be for residents to cross the street? How long will a light need to be red for them to cross it? Will the street need additional lighting?

In the next section, we discuss the architectural and design considerations for the project design phases that ensure success from a performance standpoint.

Use the right tool for the job

Dynamics 365 products are flexible and can be changed to suit the needs of many businesses. However, exercise restraint when considering incorporating this flexibility into your design. Just because we can adapt the products to achieve something functionally, it doesn't guarantee they will do them well.

For example, the xRM concept, which became popular around the release of Dynamics CRM 2011, spawned many systems to manage any type of relationship in the CRM product. The ease of development for basic data access, including a built-in user interface and security model, combined with its rich extensibility features, made it a popular choice to begin system development. Although this proved successful in many situations, many projects ran into trouble because they used the product in a way unsuited to its strengths. Dynamics 365 products are designed for specific use within a business context. They're designed and optimized for users to access master and transactional business data, not for keeping high-volume historical transactions.

Additionally, keep the Dynamics 365 roadmap in mind. It's worthwhile to explore newer features because they often bring performance enhancements compared to older approaches.

Many areas of this book discuss what to consider when making system design decisions. Often the consequence of making the wrong decisions during these stages are performance issues. It's important that use of Dynamics 365 products is in line with their intended purpose.

Design for scalability

Whether you design a solution for the first time or redesign one due to issues, keep performance in mind from the beginning. The following are recommendations for building a highly scalable solution.

- Design for parallelism and multi-threading for high-volume scenarios.
- Use set-based operations—`RetrieveMultiple`, `insert_recordset`, and `update_recordset`—as appropriate.
- Build multiple lightweight processes connected to each other.
- Design shorter transactions to reduce lock escalations and increase user concurrency.

Design for responsiveness

Response time is important for the success of your solution. The following are recommendations for building a highly scalable solution:

- Consider whether to design for synchronous, asynchronous, or background (batch) processing.
- Consider data growth over time because rapid data growth requires a lot of tuning to keep the same response rate.
- Design for aggressive caching in Dynamics 365 Finance and Operations as appropriate.

It's important to note that users typically assess whether performance is in line with their expectations, not within an arbitrary period. With this in mind, we can consider performance solutions that don't actually involve speeding anything up.

For example, choosing to execute a task asynchronously doesn't reduce the time to execute the task, but it does ensure the user interface doesn't prevent the user from working. As long as the user doesn't need the result of the task in real time, this could be an acceptable performance solution.

Data migrations and integrations

Many performance issues related to data migration and integration happen due to unreasonable expectations of the platform, specifically around data volumes. A good data strategy often works around this by ensuring that Dynamics 365 products contain only the data required for users to complete their day-to-day operations.

When planning a data migration, it's important to understand the business requirements for the data being migrated. Implementation teams sometimes decide to migrate far more data than required and then run into performance issues due to the size of the data set. For example, importing the last 10 years of unconverted leads into a business's sales system will likely take a substantial amount of time. The fix here is unlikely to be to speed anything up; it's more likely to involve questioning the value of 10 years of data, much of which is probably stale and unlikely to provide value.

In the case of high-volume integrations, consider when and how often these will run. Heavy processing can place a strain on the application, and if this happens at the same time that users are interacting with it, issues are likely to arise. This is particularly important to consider for global businesses, whose systems might be in almost continuous use.

It's also important to be aware of platform limitations when designing any high–data volume or processor-intensive operations, which are common in data migration and integration scenarios. For example, Dynamics 365 Customer Engagement has documented throttling limits that need to be accounted for within designs. Working within these limits helps ensure that the performance of the underlying platform remains acceptable. As another example, the Finance and Operations OData endpoint has a page size limit of 10,000 records and isn't designed for high-volume integrations. Other approaches, such as Data Management Framework (DMF), are better suited for these types of integrations.

Chapter 16, "Integrate with other solutions," discusses how to design integrations and integration patterns and anti-patterns in Dynamics 365.

Prepare for implementation

In this section, we discuss the design decisions the team should make before proceeding with the implementation.

Environment planning

Chapter 9, "Environment strategy," discusses environment planning in detail. From a performance perspective, consider the following as the team moves towards implementation:

- Performance testing typically occupies an environment for a significant amount of time, so a separate environment is usually advisable.
- Latency adds overhead to every operation. Minimize overhead as much as possible by ensuring that applications are located as close to each other as possible.
- Choose a performance test environment that's representative of the production environment whenever possible. For example, for Finance and Operations, the implementation team should

recommend the appropriate environment tier based on expected load and volume.

Keep in mind that if performance issues arise within scaled-down development or test environments, it's inaccurate to assume performance will improve in an environment with more resources. Without understanding the root cause of a performance issue, it's impossible to determine whether additional infrastructure will help at all, so it's crucial to understand and address performance issues early in the development process rather than assume they'll go away in production.

User personas

The organization should have a sufficient number of user licenses to be able to realistically model the anticipated behavior during performance testing. Teams also need an understanding of the user personas for testing expected usage. Keep the following in mind for user personas:

- User locations
- User security configurations
- Data typically associated with each type of user
- Expected number of concurrent users divided by persona

Proof-of-concept development

Teams sometimes run an initial proof-of-concept project before implementing the main project to prove out ideas and gain understanding of product suitability. Custom functionality built during proof-of-concept projects is often carried over into the main project.

Although this approach can significantly accelerate development during the main project, it's worth bearing in mind that there's often little in the way of developmental governance for proof-of-concept projects, which are usually undertaken without any performance requirements or considerations in place. These projects can therefore become a source of performance problems unless the team takes the time to review the outcomes along with any new requirements prior to further implementation.

Customization and performance

The extensibility of Dynamics 365 applications provides the powerful ability to tailor software to meet individual business needs but also introduces risk—performance issues commonly involve product customizations and integrations, particularly those involving code. This section discusses these issues and provides guidance on how to avoid common problems.

Chapter 15, "Extend your solution," discusses how to approach customizing Dynamics 365.

Just write fast code?

Many performance issues are due to custom code, and the fix inevitably involves reworking this code to remove bottlenecks. Project teams often question why such code wasn't written correctly to begin with, but the truth is that software development is difficult. Developers regularly wrestle with complex and abstract processes based on vague and changing requirements along with competing priorities against tight deadlines.

The most productive developers tend to focus on what's necessary and disregard anything superfluous to requirements. Keep in mind that if a requirement is missing key information, the developer won't be aware of this and will proceed with implementation based on the given information. It's common for requirements to lack performance criteria, which isn't typically top of mind for the developer or part of sign-off criteria.

It's also uncommon for a developer to deeply understand the business context of the system for which they're developing code. A developer likely might not have experienced the time-pressured environment of a call center or order fulfilment center and therefore won't be able to judge the need for performance. Therefore, specific performance requirements should be mapped through to functional requirements where appropriate to ensure that developers have visibility to performance constraints on the functionality for which they're responsible.

Bad design leads to bad code

Performance issues related to poorly performing code often aren't due to the code itself but rather to a flaw in the design decisions. This can result in situations in which developers are unable to write code to perform an operation efficiently, even when carefully following best practices, because of the constraints placed on the solution by its design.

Retrofitted performance

Sometimes developers focus on getting code working correctly before working quickly. Although this can be a reasonable approach, the pressure of deadlines often means that the optimization planned for a later date doesn't get done, leading to technical debt and a need for rework. A better approach is to be clear on any performance constraints from the beginning and then implement the solution accordingly.

Connected components

Performance issues can occur even when developers write fast, efficient code. This is because in larger projects involving several developers, the isolated nature of development might result in performance issues not being discovered until components are put together as part of a larger process. This risk should be identified during design activities, and it can be mitigated using a code review process during which team members can consider the impact of the implemented pieces of code running together.

Requirement evolution

A change in functional requirements can be another reason for customization-related performance problems. A developer decides how to implement code based on the given requirement at a point in time. A change in requirements might invalidate some or all of these decisions and cause the implementation to become unsuitable.

Common mistakes

Code can become suboptimal from a performance standpoint for a number of reasons, and specific guidance is beyond the scope of this chapter. However, the following factors are often involved in performance challenges, so we recommend that you understand and avoid them during implementation.

Chatty code

One of the most common causes of code-related performance issues is excessive round trips. Whether between the client and server or between the application and database, an excessive number of requests for an operation can really slow it down. Every request carries latency and processing time overhead, and it's important to keep these to a minimum.

Round trip issues are sometimes due to poor control of variables; a process might be so complex that it's easier for a developer to retrieve the same data multiple times than structure the code to minimize the calls. We recommend that developers avoid this practice and that the implementation team identifies such problems during the code review process.

Some of the content in this section discusses technical concepts. For those less interested in the technical aspects, note that the details aren't essential to understand. Also note that free tools are available for less technical people who want to learn about these concepts, including the Dynamics 365 Customer Engagement Solution Checker and the Finance and Operations Customization Analysis Report (CAR).

Round trip issues are also sometimes due to queries being executed within a loop, or worse, a nested loop. The code structure works for the developer, but the parameters of the loop can add a significant number of calls into the process, which in turn results in a significant performance issue.

Consider the pseudocode example in **Figure 17-3** as triggered manually by a button click in the user interface.

If `ExecuteRequest()` sends a single request to the server, the total number of requests will be the product of the sizes of `collection1` and `collection2`. Assume each request takes an average of 50 milliseconds.

It's common for developers to write logic that iterates across collections of data that are dynamic in size, due to the data-driven nature of code within Dynamics 365 implementations. It's also common for developers to work with low volumes of data within development environments. However, that means that these types of issues often aren't identified until the latter project stages, which include meaningful data volumes. Low data collection can be avoided by prioritizing minimal data retrieval during code design, retrieving each piece of data only once, and identifying any deviations from these steps as part of a code review process.

Fig. 17-3

```
foreach (var a in collection1)
{
      (foreach var b in collection2)
      {
            ExecuteRequest ();
      }
}
```

Size: `collection1`	Size: `collection2`	Total requests	Total execution time
2	2	4	0.2 seconds
5	5	25	1.25 seconds
10	10	100	5 seconds
50	50	**2,500**	**2 minutes, 5 seconds**

Retrieving too much data

Another common performance-related issue is retrieving more data than necessary, often in columns. For example, the practice of selecting all the columns in Dynamics 365 apps to avoid specifying individual columns can cause performance issues. The number of joins executed on the database server to provide lookup names and option set values can be a significant overhead, particularly when querying large volumes.

The same applies for rows. Be mindful of the amount of data being retrieved because it takes time to query, download, and process the volume of records even before the code is able to interact with them.

Broadly speaking, only the data required by a process should be retrieved; anything more is simply a waste of resources.

Intercept synchronous events

Many events in Dynamics 365 products can be customized, but if the customization executes synchronously, it adds overhead directly to the user's experience. For example, if a customization placed on `RetrieveMultiple` in Customer Engagement executes, or an event insert or pre-insert in Finance and Operations executes, the user has to wait for the customization to execute while they watch a spinning icon. This approach isn't recommended, but when it's necessary, exercise extreme caution to ensure minimal performance impact.

Unintended execution

Performance issues sometimes happen because customizations are executed accidentally or multiple times in error, for example, duplicate plug-in steps or duplicate method calls. Project teams should ensure that developers are aware of exactly how their customizations are triggered and mindful of circumstances that might inadvertently trigger their code. Background processes or batch jobs recurrence should be set according to business needs.

Performance testing approach

The project team and users need to be confident that requirements identified earlier in the project are achieved when the system is implemented. A performance testing strategy ensures that system performance is measurable and provides a clear indication of whether performance is acceptable.

What makes a performance test good?

A common approach to performance testing is to try to push the system to the breaking point to learn whether the system can handle the demand. However, this approach typically provides little value, particularly in Dynamics 365 apps. The best approach is to model or simulate expected system usage including peak load and day-in-the-life business scenarios to make sure the solution is capable of achieving defined performance goals.

Realistic

Being realistic means understanding the quantity and personas of users using the system at a given time and defining day-in-the-life activity profiles for the personas to understand the actions they'll perform. If the performance test incorrectly assumes that all the users will run the most complex processes in the system concurrently, the projected demand placed on the system will be far higher than reality. Strive to model the users in an accurate way to get the most meaningful results from the test.

It's important to understand the difference between testing the functionality and testing as a user would use the system. Functional testing is often achieved by invoking specific logic and measuring the outcome, for example, an API call, whereas user testing is testing that models the actual behavior of a user. Testers often create a suite of functional tests pointed at a web service, execute them with a large number of concurrent users, and call that a performance test—but this provides misleading results. Typically, this approach pushes certain functionality much more than the usage expected once the features are live (creating false performance issues), and other functionality can be bypassed entirely (for example, the user interface). The result is that genuine performance issues can slip through testing unidentified.

Keep in mind that user interaction with the application is fairly slow. For a given process, users don't typically click buttons as quickly as they can; they take time to think between actions, and this can be incorporated into a performance test as a think-time variable. This can vary from user to user, but an average figure is sufficient to model behavior.

The key point here is to develop a performance test that represents a number of users working concurrently and place a realistic amount of demand on the system.

Developing a reliable performance test also includes realistically modeling client infrastructure. For example, there's little value in performance testing your Dynamics 365 Field Service or Warehousing mobile application from the office when users will likely use the application in areas of low mobile reception.

The configuration of the system should also be realistic and based on the expected infrastructure and configuration of the production environment, use real security configurations for the test users, and work with volumes of data anticipated in production.

Isolated

A dedicated performance testing environment is generally recommended for two main reasons.

Performance test results are meaningful only if testers are aware of the activities occurring in the system during the test. A performance test is worthless if an unaccounted-for process places additional demand on the system during test execution.

Additionally, it's common for performance testing to occur around the same time as UAT. Performance testing can take a substantial amount of time to execute and push the system to its limits, thereby disrupting any manual user testing activities. A dedicated performance testing environment ensures that performance testing can happen in parallel with other testing.

Business data

Key to good performance testing is using business data such as setups, configurations, masters, and transactions. It's recommended to use the configurations and data to be migrated that will ultimately go live in production. Additionally, all the data preparation activities must be ready in advance—for example, data from 100,000 customers or sales orders should be available via an import file.

Repeatable with low effort

Ideally, we execute a performance test once and use the results to ensure that the criteria specified in the requirements are achieved. However, performance testing is rarely a one-time activity—often, it's an iterative process. This happens for various reasons; for example, functional changes might occur during UAT that risk a change in performance. Similarly, if performance testing demonstrates an unacceptable level, a retest of optimizations will be needed once the solution is implemented.

If a low-effort performance testing approach isn't available, it can be tempting to skip performance test iterations, which makes it difficult to assess the impact of any changes made between runs. Conversely, a fully automated process, potentially including environment provisioning and data setup, can be executed frequently with minimal effort, providing a clear picture of variations in performance in response to specific changes.

Functionally correct

A performance test result is meaningful only if the system functions correctly. It's tempting to focus on the performance metrics of successful tests. However, if errors occur, they should be corrected and the test should be executed again before any analysis is performed on the results. Deviations in behavior between test runs can significantly skew a performance test result and make any comparison meaningless.

Document results

The output of performance testing activities should be documented clearly so that interpretation is straightforward. The results should be mappable to performance testing criteria and enable the team to quickly assess whether the requirements were achieved or whether there are gaps between requirements and results. For example, page load times can be captured for certain activities and compared to acceptable page load time requirements agreed to by the business. It should also be possible to identify when the performance test was executed and against which version of code if applicable.

Finally, it's important that the information in the documented report of results remains consistent with the defined template; the team might need to compare various performance test run results as changes are made.

Plan performance tests

Document your performance testing strategy and include the following:

- Performance testing scenarios including key user activities, background (batch) processes, integrations, and reports
- Normal and peak data volumes for each scenario
- Ideal target and maximum acceptable target response time for each scenario
- Day-in-the-life workload to simulate in the most realistic manner, especially at peak hour
- Data preparation activities for each scenario, for example, existing master or transaction data in the system or creation of file to import
- Environment for performance testing
- Iterations of performance testing (plan for two or more rounds)
- Performance testing tools
- Test results documentation tool
- Quantity and profile of concurrent users including security configuration
- Guidelines and workflow for investigating and fixing performance issues
- Risks such as delays and budget related to performance testing

We recommend investing the time and effort to automate this process as much as possible. For example, with relatively low effort, you can set up performance tests to run through Azure DevOps release pipelines, providing a way to execute test runs with a single button click or automatically after a deployment. You can even run them overnight.

Network conditions during testing

Be aware of network conditions during testing. Performance tests are often conducted in isolated environments with the best network conditions, so they usually show favorable results. However, when the solution is moved to real-world use, performance issues might arise due to actual network conditions and the solution being implemented at scale.

To benchmark what's feasible for the solution when all network conditions are favorable, ensure that the performance testing environment is located in the same Azure region in which you plan to deploy production for business users. As mentioned earlier, latency adds overhead to every operation, and minimizing testing latency can help identify potential location network issues if the test results vary with the location testing.

Address performance issues

Analyzing performance problems is complex and situation specific and is beyond the scope of this book. However, let's discuss some broad considerations that might help you structure an approach to resolving issues.

Deal in facts

When faced with a performance problem, implementation teams often begin to frantically point fingers at certain parts of the system and develop ideas to find the magical quick fix to solve the problem. Unfortunately, this approach often causes more problems than it solves because teams make significant changes based on instinct, often degrading performance further or causing regression bugs as a result.

It's crucial to understand the root cause of a performance issue in detail before making any changes. This can be a time-consuming process depending on the complexity of the issue, but it's necessary to ensure that any optimizations made are meaningful. Ideally, the team will understand the issue in enough detail to have a reasonably accurate expectation of the differences in performance prior to testing the change.

Expectations

Keep in mind that performance issues are rarely due to a single issue. Suboptimal performance is most commonly the result of a number of factors working together, so a single fix is typically unrealistic. Generally, performance is an iterative process of applying incremental improvements.

In addition, as discussed throughout this chapter, a performance issue in a business's Dynamics 365 solution doesn't necessarily imply

a performance issue in the Dynamics 365 application or its underlying infrastructure. Microsoft support teams are available to help with performance issues, but keep in mind that when a performance issue is isolated to a single environment, investigations typically start by examining anything unique in that environment, such as customizations, rather than the application code or infrastructure.

Knowledge is power

Performance issues can be complex and difficult to resolve, so it's vital that the implementation team has sufficient knowledge to be able to ask the right questions and analyze issues meaningfully. The implementation team is often able to assist with performance issues, but issues can surface after go live and expiration of any warranty period. It's therefore crucial to transfer knowledge to allow business as usual (BAU) teams to resolve performance issues.

Identify problematic customizations

To identify the source of a performance issue, we recommend that you first assess whether the issue occurs in a specific customization. Many Dynamics 365 customization options can be enabled and disabled, and strategically disabling customizations often helps identify whether the issue is customization related, and if so, which customization is the source of the issue.

Proceed with caution

When you identify the root cause of an issue, it's important to take a risk-averse approach to optimization. Performance tweaks can involve complex code changes and have significant risk of regression issues, so it's crucial to proceed with caution.

Low-hanging fruit

It's advisable to identify smaller opportunities for performance gains, rather than consider reworking large and complex areas of the system. For example, for a poorly performing piece of code, there are usually several options for optimizations with varying risk of causing issues and varying performance gains. In some situations, it might be advisable to make a number of low-risk tweaks; in other situations, it might be better

to make a more complex change and manage the risk. Which approach to take is entirely situation dependent, but it's important that the team agrees on the approach and that the changes are managed carefully.

One change at a time

It's crucial to assess the impact of a change as soon as it's made—if multiple optimizations are made at the same time, it's impossible to understand the impact of each change after a performance test run. For example, if two changes are made and one increases the performance by 10 percent and the other decreases it by 10 percent, the net result will be no performance change. By testing the same changes individually, the team can use the performance test results to reject one change and accept the other one.

Workshop strategy

FastTrack runs a solution performance workshop focused on solution design that covers the impact of additional configuration and customization on the overall performance and end-user experience. The workshop emphasizes the importance of performance prioritization, goals, and testing during the stages of a project.

Workshop scope

The workshop includes the following topics that address how to incorporate performance activities into the overall delivery plan and allocate sufficient performance expert resources to the project:

- **Data volumes** Projected workload and integration volumes to ensure expectations are within limits and aligned with intended product usage
- **Geolocation strategy** Physical locations of users and servers to identify any network-related challenges
- **Key business scenarios** Main areas of the business for which performance is particularly important
- **Extension performance** Planned customizations to understand how implementation is aligned with recommended practices
- **User experience performance** Modifications to the user experience in conjunction with best practices

- **Performance testing** Performance-related goals and testing strategy to ensure performance is measured

Timing

We recommend that you conduct the performance testing strategy workshop before solution design or as soon after as the team is able to provide detailed information about performance requirements and the performance testing strategy. Scheduling a workshop later in the implementation is risky because any findings and recommendations from the workshop could cause significant rework.

Product-specific guidance

Operations

Following are recommendations for achieving optimal performance in your Finance and Operations solutions:

- Use Tier-2 or higher environments based on business objectives. Don't use a Tier-1 environment.
- Keep the solution up to date with hotfixes, platform updates, and quality updates.
- Identify and maintain a log of performance-related risks.
- Use DMF to import and export large volumes. Don't use OData for large volumes because it isn't natively built to handle large payloads.
- Use set-based data entities and parallelism to import and export large volumes.
- Build your own data entities to avoid potential standard entity performance issues. Standard entities contain fields and tables that you might not need for your implementation.
- Configure a batch framework including batch groups, priority, and threads.
- Define batch groups and assign a batch server to each batch group to balance batch load across AOS servers.
- Design heavy batch processes to run in parallel processing.

- Use number sequence preallocation.
- Perform cleanup of routines and jobs regularly.
- Avoid record-by-record operations; use set-based operations such as `insert_recordset` and `update_recordset` where applicable.
- Define business user roles and assign the appropriate level of access needed, which makes for better performance. The administrator role, for instance, will have better performance than a user with limited access.
- Use the Optimization advisor workspace to identify business processes to be optimized.

Performance tools

- Trace Parser
  - Diagnoses performance issues and various errors
  - Visualizes execution of X++ methods as well as the execution call tree
- Lifecycle Services Environment Monitoring
  - Monitors server health metrics
  - Monitors performance by using the SQL insights dashboard
- Query Store
  - Reviews expensive SQL queries during defined intervals
  - Analyzes the index used in queries
- PerfSDK and Visual Studio load testing
  - Simulates single-user and multi-user loads
  - Performs comprehensive performance benchmarking
- Performance timer
  - Helps determine why a system is slow
  - https://yoursite.cloudax.dynamics.com/?cmp=USMF&debug=develop
- Optimization advisor
  - Suggests best practices for module configuration
  - Identifies obsolete or incorrect business data

Customer Engagement

The following are guidelines taken from the Fast Track Performance Optimization Tech Talk for optimizing solution performance:

- Configure before customizing. Achieve a goal by configuring standard application components if possible. Customizations such as plug-ins and JavaScript are useful, but they can carry a performance overhead and can also make future upgrades more challenging.
- Avoid `OptionSet` attributes in a Quick Find search.
- Stay current on deprecation announcements to align with the product roadmap.
- Reduce form load JavaScript for a better form load experience.
- Display the minimum required number of fields required in the forms.
- Design forms and pages to display the most important information at the top.
- Minimize the number of controls in the command bar or ribbon.
- Use collapsed tabs to defer loading content.
- Avoid unsupported customizations such as direct Document Object Model (DOM) manipulation.

Use the Solution Checker to detect potential performance issues in code and configuration.

Conclusion

This chapter discussed why performance is expected and critical for user adoption, the customer experience, and project success. We noted that although Dynamics 365 projects are built to perform well at scale, their flexibility means it's crucial that implementation teams consider performance as an iterative process throughout the solution lifecycle.

Performance often involves tradeoffs, so project teams need to work closely with users and business stakeholders to determine and align performance expectations.

Good solution performance starts with good design. This means using Dynamics 365 products for their intended purposes and fully considering the performance implications of design decisions early on in the process. Many performance issues occur due to poor customization,

configuration, and design choices, so it's important to proactively work to avoid this by clarifying requirements and implementing best practices.

Performance testing is crucial to ensure performance expectations are achieved during implementation. However, performance tests are valuable only if they're meaningful and provide a clear indication of how the system will perform in the real world.

Performance issues are difficult to resolve and place unnecessary strain on the implementation team during the most critical points of the project. With the right prioritization and planning, performance can be baked into the implementation strategy, leading to a much more confident launch with greater project success.

References

Product-specific guidance (Operations)

Monitoring and diagnostics tools in Lifecycle Services (LCS)
Performance troubleshooting using tools in Lifecycle Services (LCS)
Work with performance and monitoring tools in Finance and Operations apps
Query cookbook
Take traces by using Trace parser
Diagnose issues and analyze performance by using Trace parser
Performance timer
Query Store Usage Scenarios
Monitoring performance by using the Query Store
Optimization advisor overview

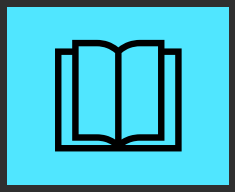
Product-specific guidance (Customer Engagement)

Performance tuning and optimization
Optimize model-driven app form performance
Tips and best practices to improve canvas app performance
Introducing Monitor to debug apps and improve performance
Performance considerations with PowerApps

Checklist

Performance focus

- [ ] Establish that the responsibility to deliver a performant solution on the SaaS platform is shared by the cloud service provider and the implementor who is customizing and extending the out-of-the-box application.
- [ ] Design, develop, and test for performance in accordance with the service boundaries.
- [ ] Include performance as part of the requirement conversation and ensure it is reflected in the design and the development of the solution, with specific documented goals that are tracked through future change requests.
- [ ] Establish development, user experience, and Information Architecture guidelines that create a positive perception of performance, prioritizing application responsiveness and user experience.
- [ ] Understand that the end-user performance is accumulative of several factors, including the network latency, device performance, browser add-ins used, firewalls, and proxies. Establish the performance baseline in the context of the actual user environment and location.
- [ ] Ensure the performance testing considers realistic loads and actual usage scenarios.



Case study

Corporate travel company learns how testing is critical to measure and optimize performance

A global corporate travel company was implementing Dynamics 365 Customer Service to drive a call center transformation. Several process flows required integration with legacy systems—many with high transaction volumes—and there was a business-mandated customization that the project team agreed couldn't be achieved using out-of-the-box functionality. This complex solution was projected to support 4,000 users at scale and needed careful planning to ensure optimal performance.

Despite the complex customizations and integrations, the customer had decided performance testing was out of scope due to budget constraints, assuming that because Dynamics 365 is a cloud service, its performance is owned by Microsoft.

While still in the design stage, the project team decided to use the best practices from the Success by Design framework and their own experiences to highlight the risks of leaving performance testing out of scope. The project team pointed out potential negative outcomes if performance wasn't considered:

- Users wouldn't adopt the system if its performance affected user productivity.
- Costs would be higher if reactive measures were required to address performance issues later.

The project architect worked with the company's stakeholders to explain that while Dynamics 365 is a cloud service, factors such as additional customizations, data volumes, and network latencies each play a role in driving the performance of the application. Their recommendations were summarized as follows:

- Review the application design and code for scalability.
- Establish baselines and review the areas that need to be optimized, including network configurations.
  - Formulate a baseline for form load times to understand performance and align a new solution.
  - Ensure performance with integrated systems meets the company's expectations.
  - Agree on realistic key performance indicators (KPIs) for the business.
  - Assess current end-to-end connectivity to determine network performance and latency.

Based on the project team's observations, the project steering committee was advised to approve adding solution performance testing to the test cycle, and agreed to do so.

The customer's architect worked with the implementation partner's architect to plan for performance testing by:

- Defining the key criteria for testing, including integration scenarios.
- Assessing the concurrent user loads and peak usage.
- Examining the data volumes.
- Establishing the tools for testing.

Performance testing helped the team identify several gaps before deploying the solution to production. In addition, the team:

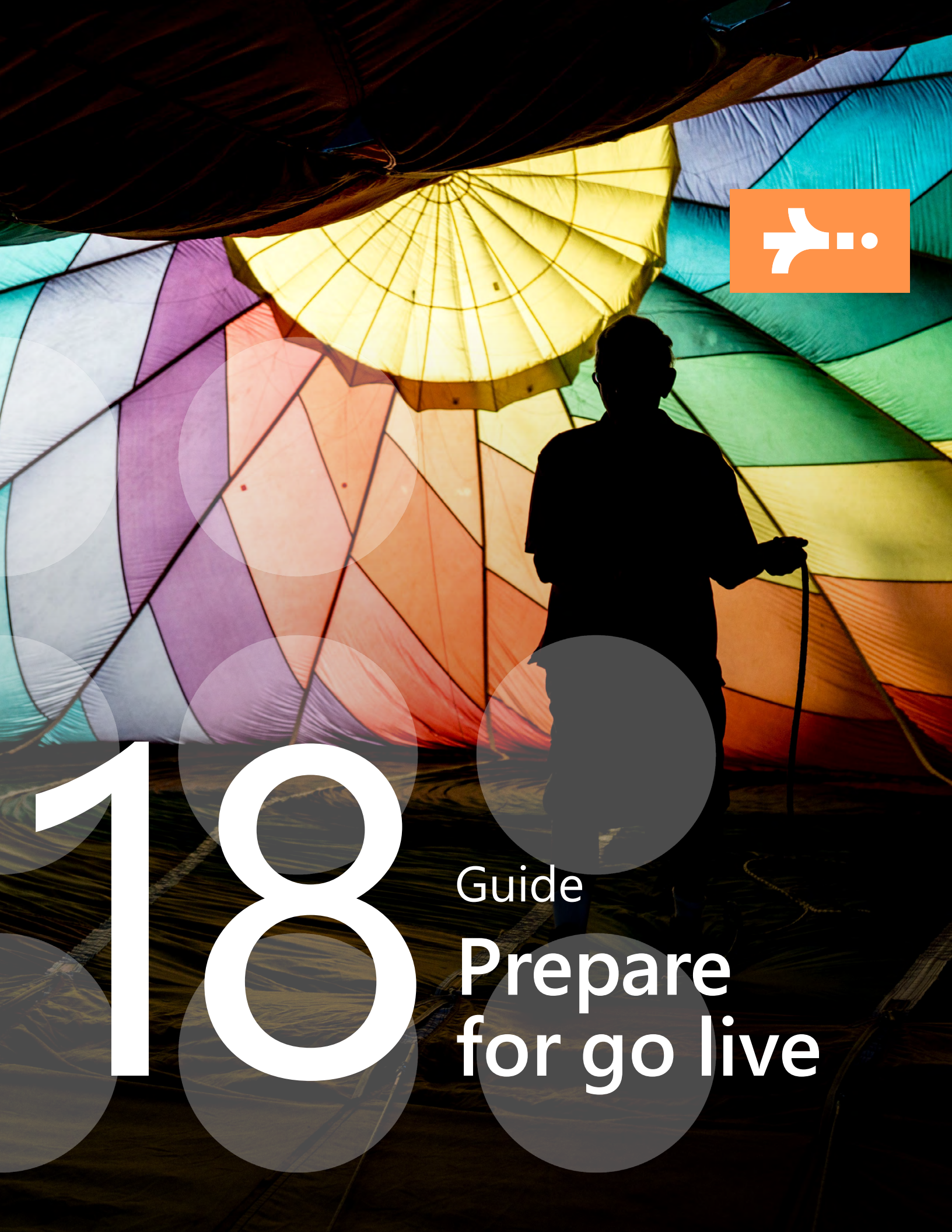
- Optimized their network configuration.
- Identified sync calls that were blocking resources, and made the necessary design changes.

- Reinforced and incorporated basic patterns (such as retrieving only the required attributes) across their code components.

As mentioned earlier in this chapter, “Prevention is better than cure.” Performance testing ensured optimal performance and stakeholder alignment before the system was made available for end users. For overall success and higher levels of adoption, a performance plan is fundamental throughout an application’s lifecycle.

Section

Prepare



18

Guide

Prepare for go live

Introduction

Go live is the process through which a new solution becomes operational. It's a critical milestone during the deployment of a new business solution.

This is the stage in which all the parts of the project come together and are tested and validated to ensure everything works as expected, not just on their own, but as a whole.

A good plan strives to ensure a successful go live by minimizing unforeseen issues and project risks not identified during previous phases. A good plan also has processes in place to deal with times when the unexpected does happen, which makes for a more resilient project.

The go-live process goes beyond getting the production environment ready and deploying the solution. It includes a variety of activities such as executing data migration and communication plans, completing training, finalizing testing, signing off on the solution, and onboarding users.

For scenarios in which there are phased rollouts, the plan should also ensure that additional deployments are considered in advance. It's important to keep refining the plan and include the learnings obtained during each rollout to improve the plan and build resilience.

The go-live readiness plan addresses all the activities to minimize unforeseen issues and project risks and build processes to help address possible unexpected issues.

Preparing the solution for a successful go live focuses on activities such as training, preparing the production environment, conducting user acceptance testing (UAT), and all the other activities described in previous chapters. Activities should be planned well in advance and completed and signed off on before the cutover starts.

And finally, the cutover, the last sprint!

Go-live readiness includes all the key activities identified in the initial stages of the project and refined throughout the project to ensure a smooth go live. These activities encompass getting all required resources to get the solution ready for production and ensuring that end users are trained, engaged, and part of the process to drive adoption and stickiness to the new solution.

When going live with a new solution, there's often a transition period, also known as cutover. The cutover process involves several steps that need to be planned, executed, and monitored to ensure the completion of all the essential activities critical to the transition to the new solution.

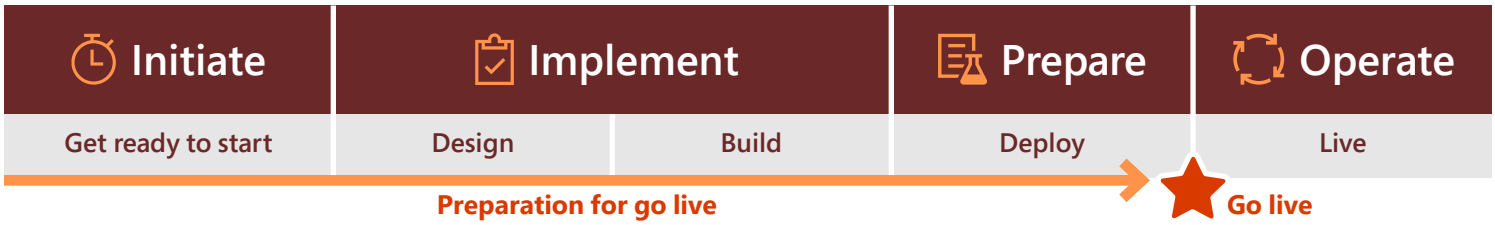
Go-live readiness

All the tasks and efforts undertaken during an implementation project are preparation for the biggest milestone of the project: go live. In the Success by Design framework phases, completion of these tasks is when the project reaches the Prepare phase.

At this point, if you followed all the previous guidance and recommended practices, the solution should have sufficient maturity for going live. You should perform an evaluation on the preparedness of the people, processes, and systems for the solution to ensure no critical details have been overlooked for go live. While you'll never be in a position of zero risk for go live, the go-live readiness review is a qualitative method to determine how fully prepared the new solution is to run your business. When the go-live strategy is aligned with best practices, requirements for going live have been successfully completed (including testing, code, and configurations), and there's a concise and agreed-to plan for the next actions required for the transition to the new system (cutover), then the project is ready to go live. **Figure 18-1** shows the Success by Design implementation phases and when go live occurs.

To get ready to go live, you need to verify completion of the most important project tasks. Once confirmed, you can perform the cutover activities. The next sections discuss these topics in detail.

Fig. 18-1



Assess go-live readiness

The implementation team should assess go-live readiness early and frequently. This provides clarity on the status, readiness, completeness, and maturity of each activity and helps project stakeholders and the steering committee make a go/no-go decision.

The assessment can be performed and reviewed multiple times in distinct phases of the project. When the project moves to the Prepare phase, the implementation team evaluates the preparedness of the solution, which serves as a quality gate to deploy the new solution. This evaluation is done with the go-live readiness review.

Go-live readiness review

To assess the go-live readiness of a project, Success by Design promotes the go-live readiness review, a deep-dive review that can be performed as a workshop. The review validates whether a solution meets an organization's needs and expectations, both functionally and nonfunctionally. It also validates the plan that has been established to ensure a smooth transition in the available cutover window and to avoid any surprises during and after go live. The go-live readiness review summarizes all the tasks that have been completed and covered from previous reviews.

The review uncovers potential risks and issues that could imperil go live and provides a set of recommendations and actions to mitigate them.

Start early to prepare for the go-live review. Schedule time to complete the go-live checklist and conduct the review, accounting for time to mitigate possible risks and issues, especially go-live blockers. Keep in

mind that time is needed to prepare the production environment. Starting the review just a few days away from the go-live date risks a delay. However, starting the review too early can also be detrimental to the process because there might not be sufficient information about the project to complete the go-live review with reliable results to properly identify potential risks of the solution.

To execute the go-live readiness review (shown in **Figure 18-2**), you should complete all required activities for going live and have the business sign off on them. The main requirements for going live are as follows:

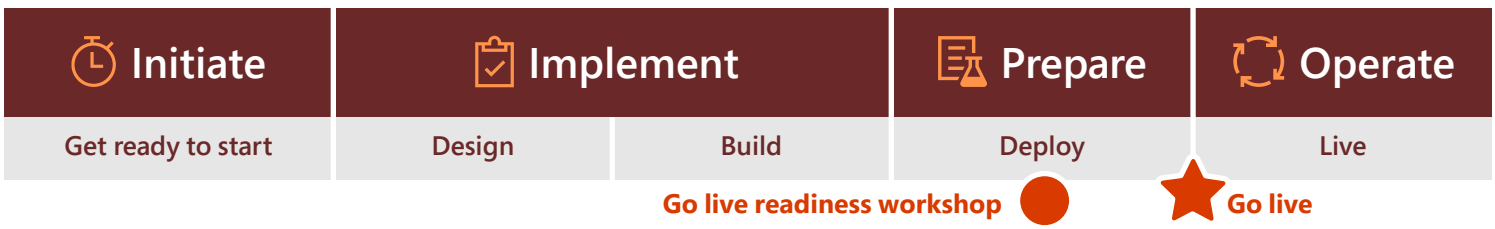
- System integration testing (SIT), UAT, and performance testing are completed and signed off on.
- Code is complete and deployable packages are prepared to move to production.
- A data migration plan is prepared.
- A cutover plan is ready and signed off on.
- End user training is complete with the correct security roles.
- The correct number of licenses are available and assigned.
- A production support plan is established and signed off on.
- Any critical open issues from the Solution Blueprint Review or other previous implementation reviews are mitigated.

The go-live readiness review is a holistic view of the implementation project: looking back on the completed and signed-off on activities required for going live, looking at solution preparations, and looking forward to the plan to roll out and maintain the new solution.

Check your readiness for go live

Once the solution is considered ready and the go-live prerequisites are fulfilled, it's time to execute the go-live readiness review and evaluate how well prepared the solution and the organization are for the next step.

Fig. 18-2



Use the go-live checklist to perform the review to help ensure a smooth go live. Create a mitigation plan to address identified issues and risks and identify an owner for each one.

Keep the review objectives in mind

When executing the review, make sure you understand the responses on the checklist. If something isn't clear, ask questions and validate the responses.

Identify all the possible risks and issues and get the recommendations on each key area. All issues and risks must have a mitigation plan. Identify workarounds for any go-live blockers. Follow up on mitigations of issues identified for critical activities.

Stakeholders must be kept informed about the results of the go-live review and resulting mitigation actions.

After the review is complete, the implementation team determines whether the go-live date is feasible.

The go-live checklist

The go-live checklist is a list of requirements for go live. Use it to assess and validate the preparedness of your solution. The checklist includes mandatory activities as well as recommended practices.

In the next section we discuss in depth the main topics in the checklist. Refer to the "Product-specific guidance" section for specific product requirements.

The Solution Scope to be released

What you check

The Solution Scope conceptualizes the final solution that will be ready for go live. It includes all the requirements designed and developed

during the Implement phase, ensuring their alignment with the business needs to be able to operate once live.

Consider the following during the Solution Scope review:

- Business processes supported by the solution
- Different solutions and applications used
- Whether the solution contains customizations and independent software vendor (ISV) solutions
- Type and volume of integrations
- Number of go lives or rollouts
- Number of users at go live and in future rollouts
- Solution Scope signed off on by the business

Why you check it

It helps the stakeholders understand the complexity and magnitude of the solution. In addition, even though the Solution Scope was defined at the beginning of the project and you're running the project according to it, reviewing the Solution Scope at this time in the project can help determine additional actions and identify potential risks.

Consider the scenario of a big-bang go live containing a heavily customized solution in which multiple integrations are delivered. The more functionalities going live on the first day, the greater the risk that issues will occur. If the risk is higher, be sure to assemble a good support team and the right escalation path. The Solution Scope must be representative of the scope of your go-live event. The recommendations you follow for a big-bang go live might be different from those for a smaller implementation.

Expected outcome

Solution Scope is aligned with the solution that's going to be live. It has been communicated, shared with stakeholders, and signed off on by the business, agreeing that the expected scope is covered.

Risks of not reaching expected outcome

Unless a Solution Scope is clearly defined and shared, stakeholders won't have an understanding of what will be included in the solution. In addition, the solution might not include all requirements or might

include developments that don't meet business expectations, which might cause delays. A well-defined Solution Scope, on the other hand, significantly increases the probability of project success.

Mitigation plan

Compare the Solution Scope with the solution delivered for go live, share the Solution Scope and comparison results with the key stakeholders, and determine whether the solution is complete or if functionalities are missing. If any are missing, prioritize them and assign level of risks, owners, and expected completion date.

Go-live checklist

- [x] Solution Scope to be released
- [x] Acceptance of the solution
- [ ] SIT completion
- [ ] UAT completion
- [ ] Performance testing completion
- [ ] Data migration readiness and validation
- [ ] Confirm external dependencies
- [ ] Change management for initial operations readiness
- [ ] Operational support readiness

Acceptance of the solution

What you check

Verify that the complete solution has been fully tested with all required types of testing and accepted by the business stakeholders.

Why you check it

The outcome, quality, and coverage of testing are reliable indicators of the readiness level for go live, ensuring that unexpected scenarios don't come up after end users get their hands on the new system.

As described in Chapter 14, "Testing strategy," testing accurately gauges the readiness of the solution. Testing measures a solution's quality and effectiveness because testing simulates how the solution will operate in real life. As a result, testing builds a solid foundation on which to determine whether the solution is acceptable, enabling the business to make a go/no-go decision. The business is required to sign off on the solution when there's objective and rigorous testing to prove that the solution fulfills the business vision.

Expected outcome

Unit testing through end-to-end testing must be completed successfully and the results signed off on by the business. By doing so, the business states that the solution as built meets their end-to-end process needs and that those processes are ready to be executed on the new solution.

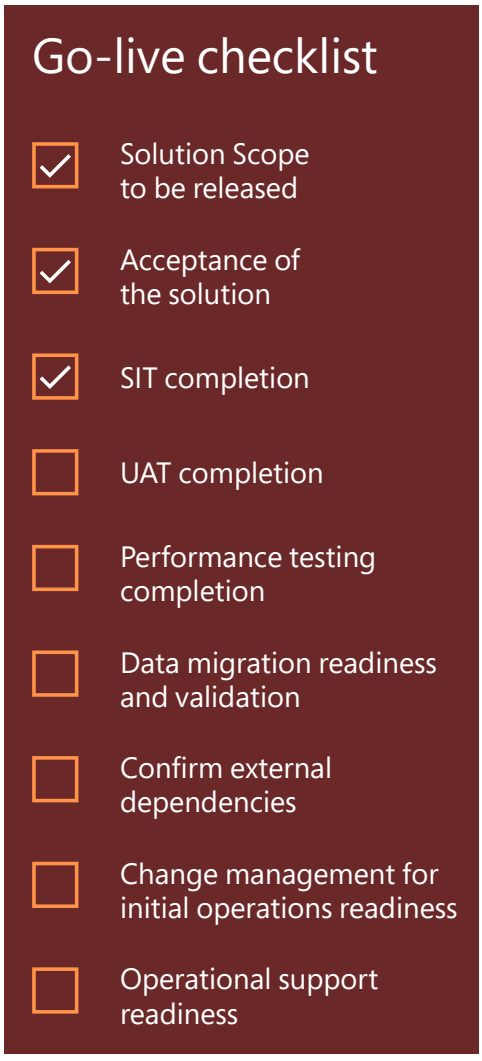
Risks of not reaching expected outcome

Going live with incomplete testing or without having sign-off from the business means that you cannot validate you have a reliable solution and therefore cannot be certain that it will work properly in production. There's a high risk of issues such as rollbacks, slow performance, unstable integrations, security holes, last-minute developments, and platform churns.

Mitigation plan

Following the best practices in Chapter 14, "Testing strategy," you'll ensure the integrity and effectiveness of the solution, minimize any go-live blockers, and avoid rushing into fixing unexpected and critical bugs close to go live. You'll complete the testing and ensure the integrity and effectiveness of the solution.

Several types of testing are covered in the chapter about testing. Some elements from these testing types are included in the go-live checklist and need to be validated. The main testing types are SIT, UAT, and performance testing.



SIT completion

What you check

Validate that SIT is successfully completed. It's important to test on expected peak volumes and have the business sign off on it.

Why you check it

SIT validates that the behavior of the solution, including integrations between different applications and with external systems, works properly. During SIT it's important not only to test and verify the interactions between all parts of the solution but also to plan and prepare for the unexpected.

Expected outcome

The testing should include simulations of external systems that are down, how those scenarios are communicated, and the impact on the user experience. Chapter 14, "Testing strategy," describes how to perform SIT.

Risks of not reaching expected outcome

Don't wait until UAT to perform SIT because this could cause significant failures during UAT. If you execute SIT during or after UAT and discover that the incorrect integration pattern was used and you need to redesign your integrations, this will put your go live at risk.

Going live without fully testing the integrations might result in unsteady and ineffective system connectivity, performance issues, flawed and weak interfaces, and data flow issues such as data inconsistency and data not being available in real time.

Mitigation plan

Complete SIT with peak volumes that are close to actual peak volumes and get sign-off from the business that the integration testing strategy works for the go-live. For guidance on which integration pattern to choose, see Chapter 16, "Integrate with other solutions."

Go-live checklist

- [x] Solution Scope to be released
- [x] Acceptance of the solution
- [x] SIT completion
- [x] UAT completion
- [ ] Performance testing completion
- [ ] Data migration readiness and validation
- [ ] Confirm external dependencies
- [ ] Change management for initial operations readiness
- [ ] Operational support readiness

UAT completion

What you check

Verify that UAT is successfully completed (or almost completed if there aren't any outstanding significant cases) and signed off on by the business.

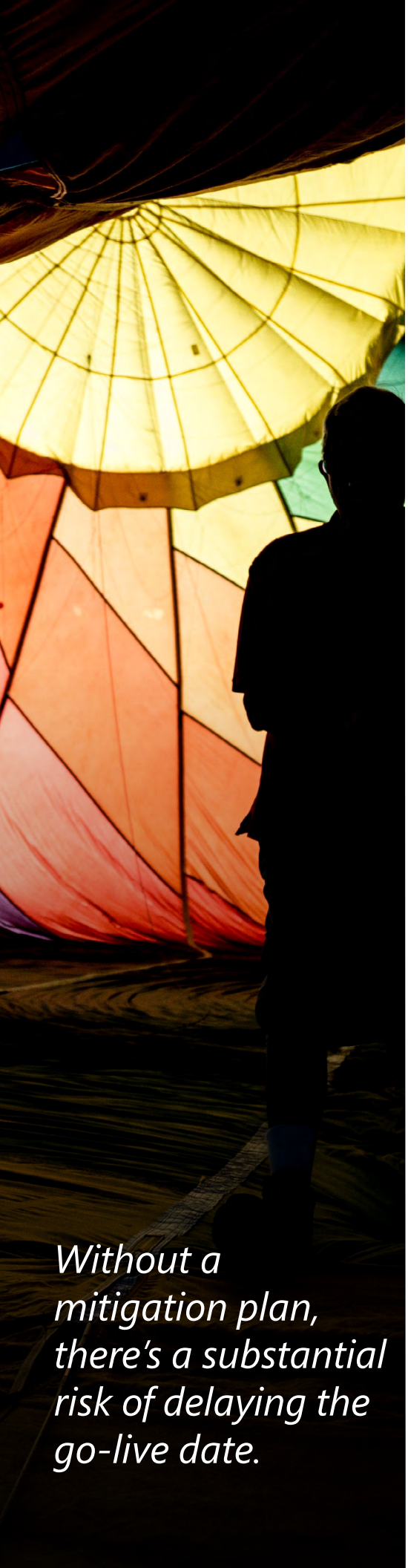
Why you check it

UAT helps validate the inclusion of all the expected functionalities. The solution might still have some minor bugs and need some fixes, but you can ensure that no critical part of the solution is missing.

Often during UAT, edge cases that were overlooked might be discovered, resulting in continuous fixes and improvements to the solution during this period.

UAT enables users to use the system and understand how to perform activities. It also provides an opportunity for users to bring up scenarios that might have been missed.

UAT should include different types of personas, a variety of geographic locations, and different environments (such as virtualized, physical, and remote) and devices (such as laptops, mobile devices, and tablets).



Without a mitigation plan, there's a substantial risk of delaying the go-live date.

Expected outcome

UAT should include the following requirements:

- Test cases should cover the entire scope of the requirements—happy path and edge scenarios.
- Use migrated data for testing because it validates how real data will behave in real life.
- Perform UAT with the correct security roles assigned to users; don't use a general security role or assign the System Admin role to all users.
- The solution must comply with any company-specific, country- or region-specific, and industry-specific regulatory requirements. Document all features and testing results.
- UAT must be conducted in an environment in a Microsoft apps subscription to ensure that the environment's properties approximate your production environment as much as possible.
- Obtain approval and sign-off from the customer.

Risks of not reaching expected outcome

If UAT isn't completed close to go live, the live operations might be delayed due to various issues including unsuitable processes, performance issues, and poor training.

If the decision is made to go live with an incomplete UAT, the project team might discover during go live that the solution doesn't work properly, and recovery would be difficult, costly, and take a lot of time and effort.

Mitigation plan

Typically, UAT is considered complete when all the cases are covered and there are no blocking issues and any high-impact bugs are identified. Establish a mitigation plan to address open items identified during UAT, with an owner and an estimated time of completion. Without a mitigation plan, there's a substantial risk of delaying the go-live date.

Performance testing completion

What you check

Validate that performance testing is successfully completed and signed off on by the business.

Go-live checklist

- [x] Solution Scope to be released
- [x] Acceptance of the solution
- [x] SIT completion
- [x] UAT completion
- [x] Performance testing completion
- [ ] Data migration readiness and validation
- [ ] Confirm external dependencies
- [ ] Change management for initial operations readiness
- [ ] Operational support readiness

Why you check it

Overlooking performance testing might result in performance issues post–go live. Implementations often fail because performance testing is conducted late in the project or not until the production environment is ready, when it's used as a testing environment. A production environment shouldn't be used as a testing environment. It's more difficult to fix performance issues in a live environment than in a test environment and might also result in production downtime.

Any open issues critical for production need to be addressed before going live.

Performance testing covers the following:

- Load times with peak volumes of transactions
- Data search performance
- End-to-end processes and specific business processes
- Performance from different devices, browsers, and virtualized environments
- Integrations
- Environment for performance meets the standards to ensure it's reliable for testing

Expected outcome

The solution delivered is performant and meets the business performance criteria: it supports the load of expected transactions and user concurrency and usage, and the speed, system response time, stability, and scalability are acceptable.

Risks of not reaching expected outcome

A common misconception is that a production environment is more performant than a test environment, and this belief is sometimes used to justify skipping performance testing. However, performance is not only about infrastructure, as discussed in Chapter 17, "A performing solution, beyond infrastructure." Therefore, we highly recommend that you conduct performance testing before going live in a test environment. A production environment cannot be used to conduct core testing; it's used only for mock cutover, cutover, final validations, and live operations.

It's important to identify the root cause of issues so as not to replicate them in the production environment.

If performance checks are performed just during UAT, that might not be a good representation of the actual usage post–go live. For example, if during UAT there's good coverage of roles and personas, regional access, devices, and environments but those are for a small percentage of users, UAT can validate the solution readiness in terms of functional and even end-to-end performance of each business process, but it doesn't represent the full load and concurrency of actual usage post–go live. Therefore, it's important to have a separate performance testing strategy that can simulate stress on the solution and concurrency usage.

Mitigation plan

Execute performance testing in parallel with UAT. Establish a mitigation plan to address issues and assign owners and expected completion dates. It's important to identify the root cause of issues so as not to replicate them in the production environment.

Go-live checklist

- [x] Solution Scope to be released
- [x] Acceptance of the solution
- [x] SIT completion
- [x] UAT completion
- [x] Performance testing completion
- [x] Data migration readiness and validation
- [ ] Confirm external dependencies
- [ ] Change management for initial operations readiness
- [ ] Operational support readiness

Data migration readiness and validation

What you check

It's crucial to validate that there's a well-designed data migration plan for go live.

Why you check it

Data migration is a key activity during the cutover. You need accurate data on day one of your live operations.

Data quality is an important aspect of data migration; you don't want to import bad data from a system into a new system. It's important to have a plan to conduct data validation.

Expected outcome

Data migration is tested several times prior to execution during the cutover to validate the strategy and processes, identify data corruption or duplicated data issues and address them, and make sure performance is measured, validated, and fits within the cutover time window.

All scripts and processes planned for the cutover migration are tested and signed off on by the business.

Any migration issues and risks are logged and a mitigation plan established. This information helps inform the go/no-go decision.

Risks of not reaching expected outcome

It's important to carry out all the steps of a data migration plan. Otherwise, there'll be a lack of confidence in the information because it might not reflect accurate data. In addition, concerns will arise about the ability to complete the go-live migration in a timely manner, especially if record quantities are exceptionally large.

Data quality is a key factor in the user experience. For example, if a user's leads are imported but a critical field is missing or has corrupted information (for example, truncated, not completed, or without the correct currency or value), this severely impacts how the user perceives the new solution, reduces the trust level, and might even eventually lower solution adoption.

Mitigation plan

Execute dry runs of how to migrate the data and prepare the migration plan. Follow more recommended practices to prepare your data migration by reviewing Chapter 10, "Data management."

Confirm external dependencies

What you check

Verify that external dependencies such as ISVs and third-party systems and services are aligned with the timelines and scope for go live.

Why you check it

External dependencies are outside the direct control of the project team. This means it's even more important for these dependencies to be accounted for when building the project plan and managing the schedule.

Expected outcome

If the scope of the solution includes external dependencies, the implementation team should coordinate with them to ensure that expectations are aligned, requirements are met, and their solutions are aligned with the roadmap timelines.

Go-live checklist

- [x] Solution Scope to be released
- [x] Acceptance of the solution
- [x] SIT completion
- [x] UAT completion
- [x] Performance testing completion
- [x] Data migration readiness and validation
- [x] Confirm external dependencies
- [ ] Change management for initial operations readiness
- [ ] Operational support readiness

All external dependencies should be documented and monitored.

Risks of not reaching expected outcome

Lack of coordination and communication with external dependencies might result in a delay in go live.

Mitigation plan

It's good practice to have regular meetings to review dependencies status because problems can cause delays and other project issues.

Go-live checklist

- [x] Solution Scope to be released
- [x] Acceptance of the solution
- [x] SIT completion
- [x] UAT completion
- [x] Performance testing completion
- [x] Data migration readiness and validation
- [x] Confirm external dependencies
- [x] Change management for initial operations readiness
- [ ] Operational support readiness

Change management for initial operations readiness

What you check

It's important to include change management at the beginning of the project and assess the level of change readiness.

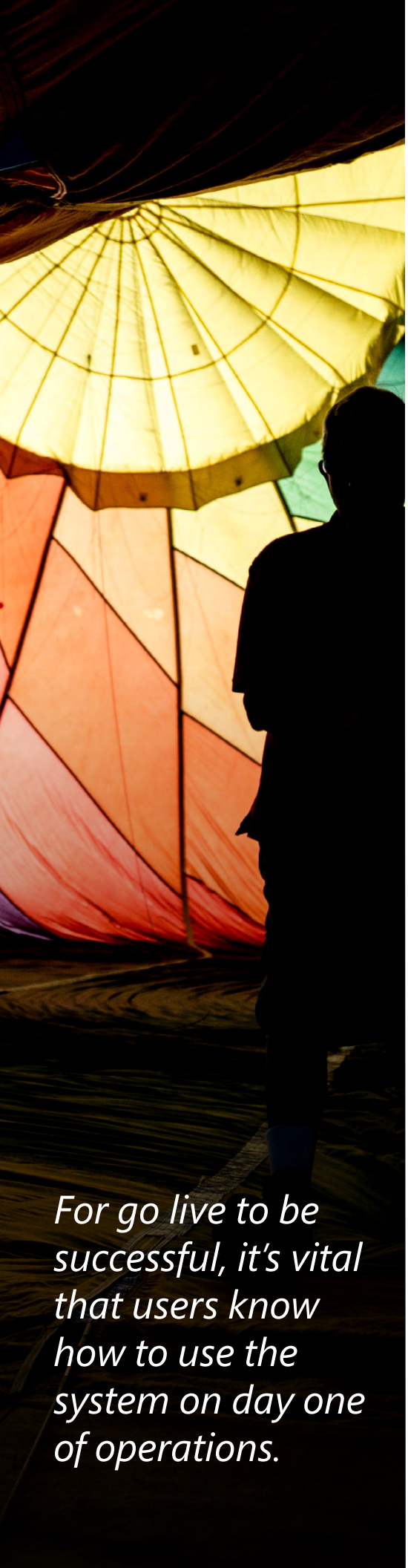
Why you check it

When delivering a business solution, the success of the technical implementation and the alignment with the business needs and goals are paramount but not sufficient to deliver the return of investment. The true value of the business solution is realized when end users actively adopt and engage with the processes implemented.

It's natural to be resistant to change even if it's positive and can significantly improve one's experience. It's important to have a change management plan that prepares users for changes. There are several resources and strategies available to achieve successful change and adoption that include ways to communicate, reduce resistance, and allow feedback.

Expected outcome

Our goal here is not to explore the plethora of change and adoption strategies but to highlight the key principles and activities that are critical before the rollout of a new business solution.



For go live to be successful, it's vital that users know how to use the system on day one of operations.

It's important to assess early in the project—and continually reassess—the changes in an organization to ensure there'll be a smooth adoption of the system.

The following are key elements to drive change management close to go live.

Training For go live to be successful, it's vital that users know how to use the system on day one of operations. Therefore, successful user training is key to go live.

Ensuring that users are trained helps them achieve the desired results from the new solution and leads to higher productivity and adoption. Failure to establish a training strategy can negatively impact usage and adoption.

The most common training strategy is a formal set of sessions delivered in a classroom setting or online. Refer to Chapter 19, "Training strategy," for additional training strategies.

When possible, end-user training should be delivered before the go-live date because this prevents delayed access to the new solution. It can also contribute to higher adoption because the users feel better prepared and a part of the program and had the opportunity to share feedback before the new solution went into production.

User engagement strategy A user-centric mindset helps address not just the functional requirements but also the unspoken needs of the user. Providing a story of empowerment to users—how the solution will help them be more efficient in their jobs, for example—is a message that strongly resonates with end users.

Your program must have an effective way to engage the end users of the solution to help drive adoption and also eliminate the risk of building a solution that doesn't necessarily meet user requirements.

A key activity to perform early in the project is the identification of the super-users, or influencers, within the community who are willing to

actively engage with the project team. Provide opportunities throughout the project for these people to provide input and feedback.

Following are sample project milestones for which super-user feedback is most beneficial:

- Early minimum viable product (MVP) or proof-of-concept demonstration
- Sprint demonstrations during implementation to highlight business functionality
- User experience and information architecture design
- Data mapping and validation that data migration activities are involved in the project
- Baseline performance metrics for frequently accessed components
- Expectations for end-user devices and accessibility needs
- Review of the timing of key milestones such as go live and cutover
- Backlog prioritization and setting expectations for gaps, open issues, and future functionality

There needs to be a well-structured nurture program for the business users that focuses on effective communication to keep them up to date and engaged in the program. Although most of this work happens throughout the implementation project, the culmination of all this planning at go live should be validated and signed off on as ready to go.

Business sponsorship The business sponsor, or executive sponsor, is a trusted and influential leader who enables cultural changes and plays an essential role in championing transformation throughout the organization by communicating the value of the new solution.

Creating an awareness of the key executive sponsors and bringing them in front of the user community to articulate the value of the new solution can positively influence adoption and usage.

As part of the go-live activities, the executive sponsor's communication strategy needs to be well planned. This includes the types of communication, the stakeholders, the timing, and recognizing the contribution from end users.

To help overcome adoption challenges, it's crucial to get leadership support to encourage innovative technology use. End users are less resistant to adopt novel solutions when business sponsors serve as a role model.

User support Creating a community that involves the super-users and providing resources such as videos, tutorials, knowledge base articles, and business processes will help with user adoption post–go live and encourage ongoing user engagement.

A well-defined channel to communicate to users about upcoming changes, open issues, and service notifications in case of planned and unplanned maintenance and outages should be established.

Risks of not reaching expected outcome

A poor change management plan might cause ineffective execution of processes on the first day of go live. Users would therefore likely have a lack of confidence in the new solution, which would be counterproductive for a successful rollout.

Mitigation plan

Reassess the change management plan throughout the implementation project.

Operational support readiness

What you check

Verify that there's a monitoring and maintenance plan for the solution in the production environment as well as for transitioning the plan to support teams.

Why you check it

Verification ensures the health and availability of the solution once it's live. Before go live, it's important to plan the transition to the solution's support teams.

Expected outcome

Support teams can be from the partner or customer organization, and

Go-live checklist

- ☑ Solution Scope to be released
- ☑ Acceptance of the solution
- ☑ SIT completion
- ☑ UAT completion
- ☑ Performance testing completion
- ☑ Data migration readiness and validation
- ☑ Confirm external dependencies
- ☑ Change management for initial operations readiness
- ☑ Operational support readiness

they need to have the necessary resources, tools, and access to subject matter experts, business analysts, process specialists, and change management champions. Support teams should also be trained on the solution; this ensures that tickets that come through the help desk are properly addressed.

Notify stakeholders when all parties agree that the system is ready to go into production and send them a list of end users who will use the new system.

Users should be notified of any unresolved issues or required workarounds and provided with a support contact.

Include the following in the operational support and maintenance plan.

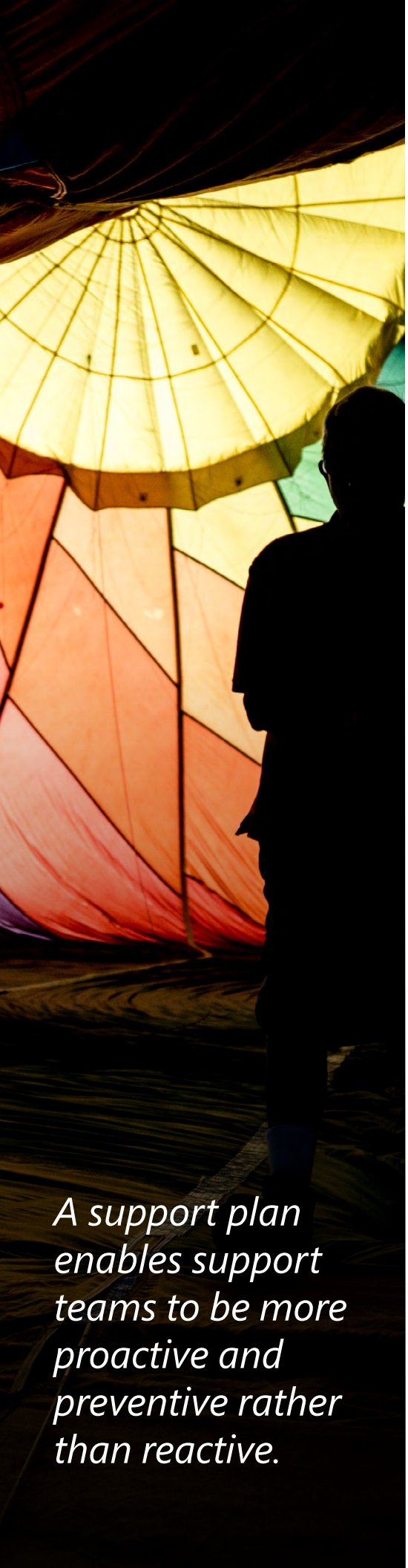
Continuous deployment plan It's important to go live in a serviceable system version in order to take advantage of latest updates, have a better quality of the solution, and have the support available if a hotfix is required.

It's also important to understand the update cadence of your new solution so that you can incorporate updates and new features and replace deprecated ones quickly. This should be part of your maintenance plan; it's vital for the continuity of your operations in production.

For discussions of continuous deployment plans, refer to Chapter 20, "Service the solution," and Chapter 21, "Transition to support."

Monitoring Establish a monitoring strategy for performance issues, errors, feature usage, and adoption. This makes the post–go live process easier to manage and makes it easier to provide updates to stakeholders about the new solution.

Hypercare Consider including Hypercare after go live because it supports and expedites the mitigation of issues, reducing time for stabilization of your operations.



A support plan enables support teams to be more proactive and preventive rather than reactive.

For additional details about monitoring and Hypercare, visit Chapter 21, “Transition to support.”

Risks of not reaching expected outcome

If there isn’t a well-defined plan to support production or if the plan isn’t properly communicated before go live, when issues start surfacing during the operations there won’t be a way to communicate them and assign resources and severity levels.

There’s also a risk that the new solution could go live in a system version that’s out of service. In such a scenario, if an issue is uncovered in production, you’ll need to update the solution to the latest version to be able to apply the hotfix. In addition, automatic updates that install unexpectedly might affect the deployed solution and cause outages, unavailability, and blockings.

Mitigation plan

As discussed earlier in this section, it’s important before go live to plan the transition to the teams who will support the solution. A support plan enables support teams to be more proactive and preventive rather than reactive.

Production environment readiness

It’s critical to prepare the production environment before go live. There are different approaches to achieve this, depending on the applications in the scope of the deployment.

For some applications, the production environment can be created from an initial phase without any dependencies. For other applications, a formal process needs to be followed to create or deploy the production environment. Additionally, several milestones must be completed prior to deploying the production environment to help with the sizing of the environment and impact on factors such as performance and scalability of the environment.

The “Product-specific guidance” section later in the chapter details the steps for each application. For a discussion of production environments, see Chapter 9, “Environment strategy.”

Establish a plan to prepare the environments for production and execute the following activities before the cutover. This prevents any last-minute issues and provides more time to ensure that tasks are completed successfully.

- Deploy applications and update them to the latest version.
- Ensure that users were created ahead of time and assigned appropriate security roles.
- Assign security groups to environments and add users to the groups to ensure that users are created only in the environments for which they should have access.
- Ensure sign-off on the security and compliance strategy including penetration testing when applicable, compliance requirements, and data sovereignty requirements.
- Tune the environment when applicable.
- Create, import, and validate configurations.
- Run and validate integrations.
- Ensure that any specific configurations for different applications are completed ahead of time, especially configurations that take time to complete. For example, setting an integration with an email engine such as Outlook might depend on the size of a user mailbox, and the bigger the mailbox, the longer the process takes to complete.

Finally, keep in mind that production environment readiness is also a big dependency for other activities during the cutover. For example, the final data cannot be migrated and validated until there’s a production environment to start those activities. And before data migration can occur, other activities need to be completed, such as creating the users and assigning the proper security roles.

Cutover, the last sprint

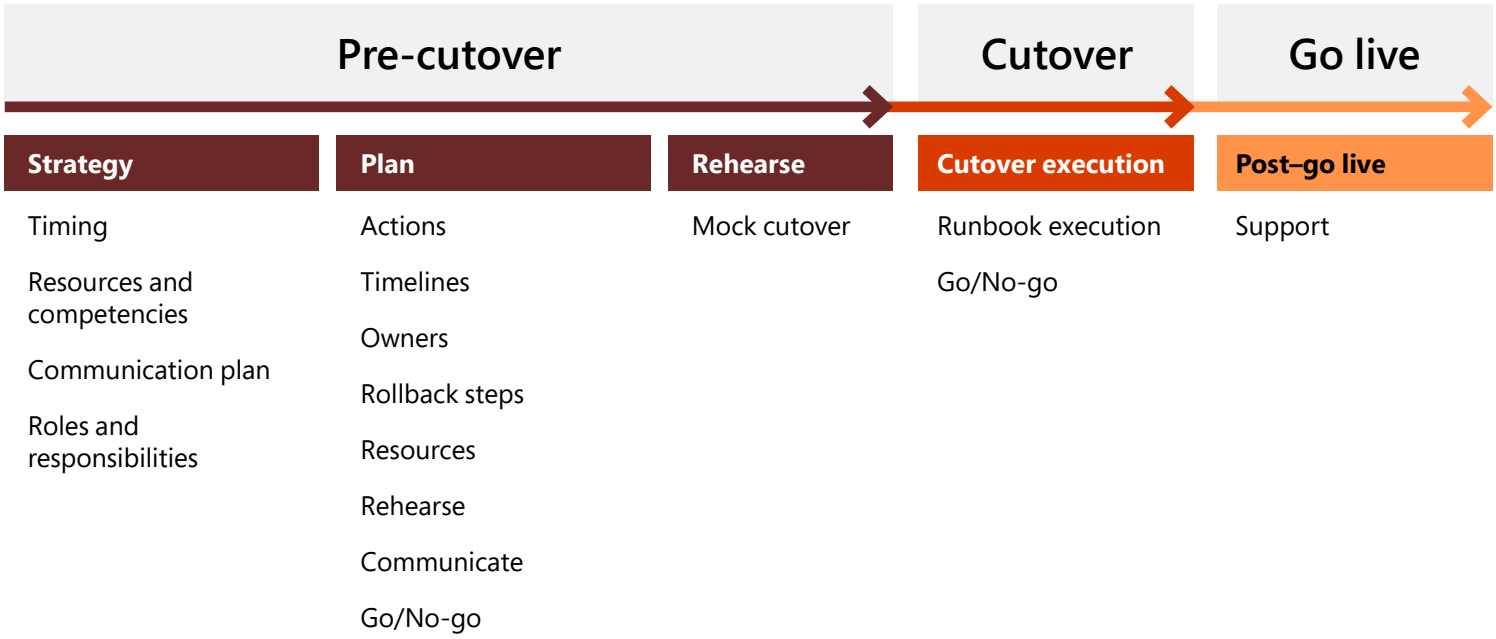
The last mile before going live is the cutover, which marks the transition to the new solution. A project cutover entails the execution of a set of

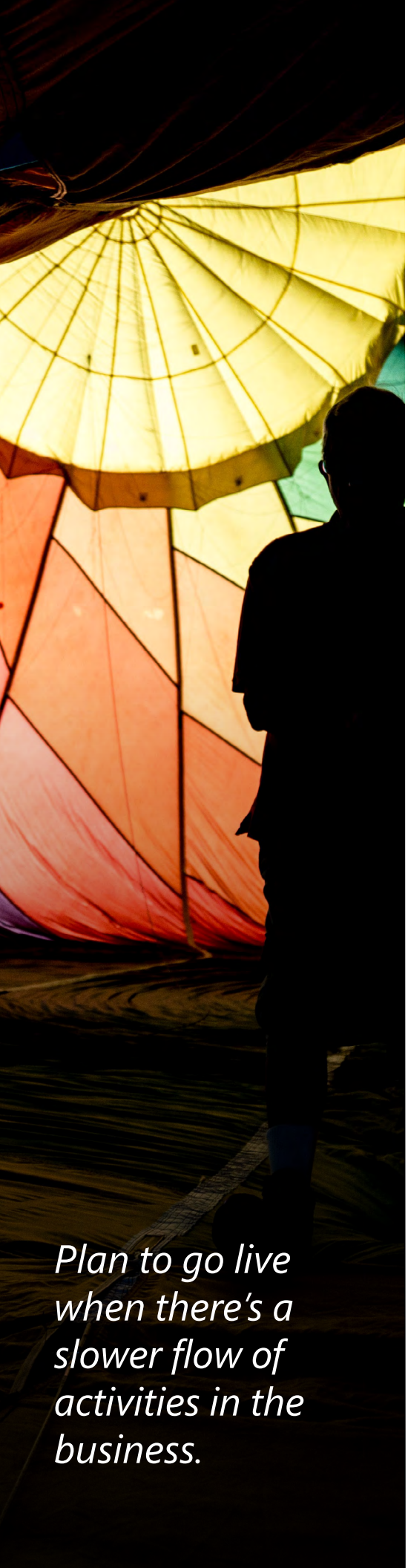
tasks required to successfully go live with a well-defined scope on a production environment. In the past, it was common to do a big-bang go live after a multi-year solution implementation period. Now, however, businesses frequently follow a more agile approach. For example, a company might plan an initial go live in a small country or region or just for the core financial processes and complete the rollout to other countries or more complex functionality at a later point. A cutover process needs to be defined and executed for each rollout.

In most businesses, only a brief period is available to complete the cutover activities because during this time, users cannot use the old business system any longer, and the new business system is not ready yet. Often the cutover period is referred to as "the cutover weekend" because it's usually no longer than 48 hours. For businesses operating all the time, this time window might be even shorter.

The cutover process, shown in **Figure 18-3**, involves several steps that need to be planned, executed, and monitored to ensure that all activities necessary to transition to the new solution are successfully completed. The cutover process includes strategy, planning, execution, monitoring, and control.

Fig. 18-3





Plan to go live when there's a slower flow of activities in the business.

Cutover strategy

The cutover strategy begins in an early stage of project planning, is refined throughout the project, and is completed before the cutover plan is created.

The cutover strategy ensures alignment of the cutover plan with organizational objectives and includes the following aspects:

- Timing, market conditions, and other environmental aspects necessary to go live with the new business solution
- Organizational readiness requirements in terms of available competencies and resources
- Setting up the communication plan
- Defining roles and responsibilities

Time your go live

It's challenging to establish a realistic and achievable go-live date because this date is set early in the implementation. It's always important to have some buffer time when planning the milestones dates, but this is especially true for the go-live date.

To set the go-live date, consider the time required to complete testing and resolve any issues that might arise, in addition to time for the preparation of the production environment and the cutover activities.

Preferably, plan to go live when there's a slower flow of activities in the business. For example, in the case of a seasonal business, like for some retailers, choose the season when sales are slow to lessen any negative impact on the business. Also consider avoiding going live during any busier times, for example, month or quarter end. In addition, take holidays into account—they might mean low activity for some businesses and for others might be the busiest times of the year. It's also important to make sure that all team members and users are fully available during that time.

In addition to those busy times, consider system release dates. Microsoft continually makes improvements in the system to deliver a better product and has established release dates that usually entail a mandatory

To determine a realistic go-live date, consider the amount of time required for each pre–go-live activity, including both the internal and external factors impacting your project. Then add some buffer time, bearing in mind that adjusting dates and resources along the project is expected because there might very well be delays, unforeseen events, loss and replacement of resources, and external or internal dependencies that don't complete on time.

update. You'll need to decide whether to set the go-live date before or after a release. This consideration also applies for any other external applications you might be integrating with your new solution.

External factors might also affect the timelines in a project, especially in long projects during which unforeseen events such as a natural disaster, political changes in the country of implementation, company fraud, or a pandemic might occur. It's important to create a list of possible mitigation plans for unforeseen events when planning your project. External factors are part of the risk management of your project.

Resources and competencies

During the process to define the cutover strategy, program stakeholders should identify which roles and competencies are required for the activities in the cutover plan and assign ownership for each activity.

The stakeholders should verify that all necessary resources are available not only for the requested duration of the cutover plan activities but also to support post–go live.

These resources might need to be requested from external companies if they're not internally available for the project team.

Effective communication helps avoid uncertainty and provides visibility to stakeholders about project status and the results of each activity, which is important for a successful cutover.

Communications plan

The communications plan is an essential part of the cutover plan. This plan identifies all the stakeholders involved in the cutover. The plan should include at least one communication channel and appropriate distribution lists, depending on the type of communication required. It's important to identify and document the different communications required for go live, who is responsible for triggering the communications, and the recipients list. Having a communication plan as part of the cutover plan enables better visibility into who the stakeholders are, at what point in time they should receive communications, and who the points of contact are.

Roles and responsibility

Effective delegation of tasks is key for a successful cutover. Therefore, it's important to define which roles are needed, based on the required

Team members working together effectively, including during the cutover as we've been discussing, is the foundation of a successful project. All those roles functioning together, doing the right thing at the right time, will result in a successful cutover.

activities, skills, and responsibilities, and assign them to members involved in the cutover. Also assign an owner to each activity and ensure that all owners understand what's required of them during the execution of the cutover.

The necessary skills and responsibilities will help better define which roles need to exist and to whom they'll be assigned. It's important that those with a role assigned know what to do and how to do it—it's expected that each person contributes with their tasks, delivering them on time and well executed.

Cutover plan

The cutover process is a critical and complex step that must be planned and practiced in advance. The center of the cutover process is the cutover plan, which lists in detail every step that needs to be executed and monitored to prepare the production environment for business processes to be executed once the cutover is complete. Cutover activities include system configuration, data migration, data validation, and decommissioning of legacy systems when applicable. These steps must be precisely orchestrated to minimize disruption and ensure a successful deployment.

When creating the cutover plan, it's important to consider and document all dependencies, the timing of each task down to the minute, who's responsible and accountable for each task, instructions, verification steps, and any additional documentation associated with each task. To arrive at the correct timings for the cutover plan, activities must be practiced and tested multiple times. It's recommended to perform a "mock cutover," or dress rehearsal, simulating the activities of the real cutover in sandbox environments. Depending on the complexity of the solution, it might be necessary to conduct multiple rounds of mock cutover. The goal is that no matter how complex the cutover procedures are, the project team has high confidence that final execution will work flawlessly.

A successful cutover plan is when there are zero or few disruptions during the execution of the cutover and a plan is in place to mitigate any breakdowns.

Components of a cutover plan

A successful cutover activity always begins with a detailed and well-defined cutover plan. The cutover activity itself is performed as part of go live, but planning for the cutover should start at an early phase of the project, and the business should sign off on it. The cutover plan is a living document that is updated throughout the project as the solution is implemented and evolves.

All project stakeholders should be involved in each activity. Project stakeholders should sign off on activities, with a commitment that all necessary resources are available for the requested duration to support the go-live activities.

A clearly defined plan including detailed actions, timelines, and owners needs to be established. The plan should also include a rollback scenario, which adds to the value of the cutover plan. Planning a rollback is especially important to ensure that the business is not affected by the outcome of the cutover plan if it isn't successful and that there's a backup plan to retry the cutover at a later point.

The implementation team should rehearse all the cutover plan steps to ensure that the cutover strategy and communication plan are ready.

Fig. 18-4

Cutover plan example

Go-live date: 8/1/2021

Today: 6/28/2021

Cutover step		Area	Environment	System	Countdown	Status	Duration	Estimated start	Estimated end	Owner	Validated by	Actual start	Actual end	% complete
Finalize configuration environment														
10	Customization installation	IMP	Config	Dynamics	3 D	Planned	1 D	7/1/2021	7/1/2021	Team 1	PM			0%
20	Configurations validation	IMP	Config	Dynamics	5 D	Planned	2 D	7/2/2021	7/3/2021	Team 2	PM			0%
30	Run data sharing policies	IMP	Config	Dynamics	6 D	Planned	1 D	7/4/2021	7/4/2021	Member 1	PM			0%
40	Take backup	IMP	Config	Dynamics	7 D	Planned	1 D	7/5/2021	7/5/2021	Member 2	Tech lead			0%
50	Move to sandbox (UAT)	IMP	Sandbox	Dynamics	7 D	Planned	1 D	7/5/2021	7/5/2021	Team 3	Tech lead			0%
60	Backup applied to UAT	IMP	Check TIME	Dynamics	7 D	Planned	1 D	7/5/2021	7/5/2021	Member 1	PM			0%
70														
80														

By doing so, the predictability of the outcome increases, backup plans can be created to avoid dependency issues, the duration of each activity is validated, and any identified issues with the plan can be corrected. **Figure 18-4** is an example of a cutover plan and shows the list of activities, the different elements for good traceability, and execution of the cutover.

All key stakeholders should attend go/no-go meetings to review any open issues, risks, and unresolved defects to collectively decide whether the application is ready for production.

The decision should be made according to the success criteria defined during the cutover plan. For example, one of the go/no-go decision KPIs might be the number of blockers or high-impacting bugs identified from UAT.

The cutover plan should also specify whether there are workarounds or additional plans that might prevent any issues from delaying go live, for example, if a blocking issue was identified during UAT but there's a workaround that can be implemented until a permanent fix can be applied. The workarounds should take into consideration the impact to end users. After all, users judge a solution by how well they can use it, regardless of whether there's a workaround. This is another reason why all key stakeholders need to participate not only to define the success criteria but also to make the final go/no-go decision.

Cutover execution

The cutover execution is the series of activities carried out in adherence to the cutover plan. Cutover execution includes the following steps:

- Communicate activities, milestones, and results to stakeholders in accordance with the communication plan.
- Ensure that all activities are executed and completed and that any unforeseen issues are addressed or can be addressed after go live, have been communicated, acknowledged by stakeholders, and documented.
- Execute the activities.
- Monitor the execution of the cutover activities.

Cutover entrance criteria

Complete the following activities before initiating the cutover execution:

- Complete test cycles (performance, SIT, and UAT) with exit criteria satisfied.
- Build and test data migration strategy including configuration data.
- Identify all mock cutover and cutover resources.
- Complete internal and external communication plans.
- Ensure that the production environment is ready.
- Complete the final cutover plan.
- Ensure the final go-live support plan is in place.
- Ensure end-user training plan will be completed by go live.
- Complete data validation plans.

Cutover exit criteria

Execute the following activities to complete the cutover execution:

- Execute all cutover steps in sequence.
- Ensure the steps fit in the time window established for the cutover.
- Document all failures and issues and resolve or identify solutions to resolve them.
- Execute data validation plans successfully.
- Obtain sign-off at cutover go/no-go checkpoint.

If you execute a mock cutover, it's crucial to validate the duration time of the different tasks and their sequences so that you can achieve a realistic plan for the actual cutover. This validation also helps you identify the need for mitigation plans and whether you need to run additional mock cutovers.

Successful go live and transition to support

Once the cutover execution is completed, the new solution is released to end users. At this point, the solution becomes operational, and it's crucial to ensure a smooth handover to the operations teams, following the operational support plans discussed in the "Operational support readiness" section in this chapter. For additional information on support, visit Chapter 21, "Transition to support."

Product-specific guidance

Operations

Operations must follow the go-live readiness review process to go live. This review process acts as the quality gate for releasing the production environment, which will be deployed only if the go-live readiness review has been completed.

For Operations, consider the following:

- **Lifecycle Services (LCS)** LCS is the workspace for managing all aspects of your operations project, including project milestones, environments, updates, deployments, support, and project administrators. As part of the go-live readiness review, the LCS workspace needs to be prepared with the correct configuration and information such as milestone dates and project users and accounts. In addition, create the Subscription estimator for sizing the production environment. Familiarity with the LCS functionalities for go live, including deploying a package, requesting a database refresh, and requesting your production environment, will help you efficiently run all the preparations for the go live.
  - LCS also contains tools such as LCS environment to monitor, diagnose, and analyze the health of the environment and troubleshoot issues if they occur.
- **Continuous updates** Ensure that your environments comply with the One Version policy. This policy specifies that an environment must have a version no older than four monthly service updates from the latest available one. Set your Update Settings in LCS for continuous updates.
  - Review the targeted release schedule to verify that the version of your environments hasn't reached the end of service. It won't be allowed to deploy production in an older version.
  - Regression testing is needed when updating or deploying fixes or making changes to the solution. The Microsoft Regression Suite Automation Tool tests regressions due to continuous updates.
- **Upgrades** For Operations, upgrades from Dynamics AX 2012 are common. It's important to perform an assessment for go live with

the specific requirements when upgrading your solution, such as the following:

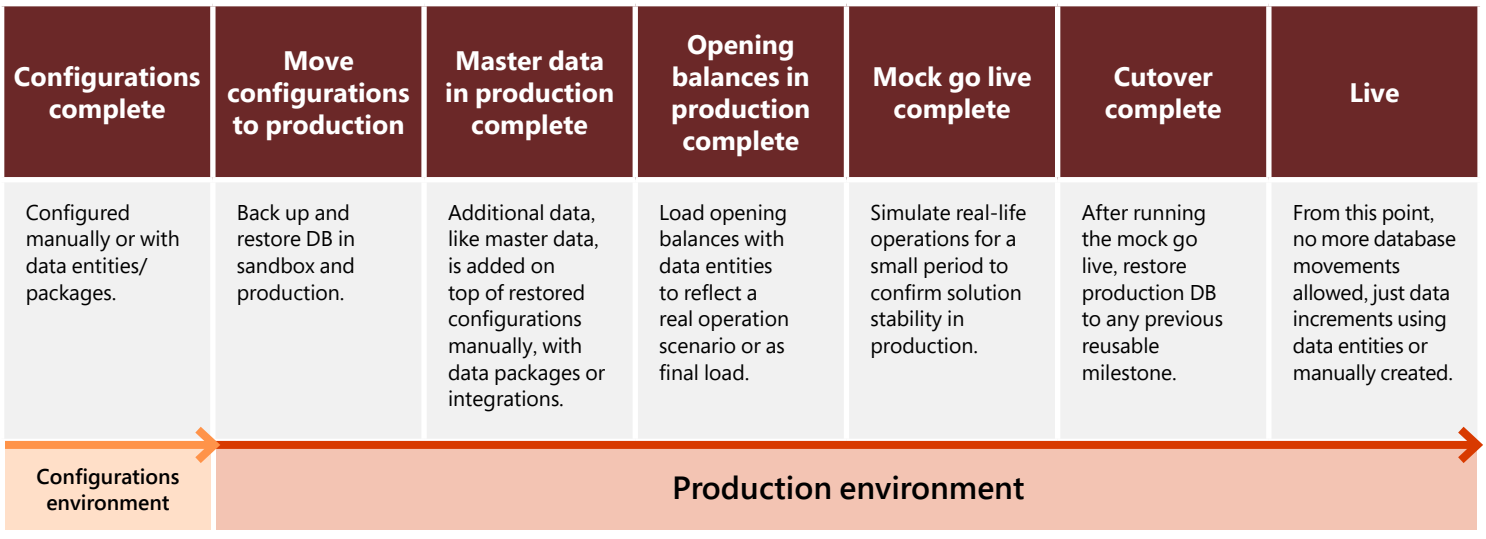
- Use the collation (SQL_Latin_General_CP1_CI_AS) because it's the one supported by the Finance and Operations apps kernel/runtime.
- Run and execute all pre-upgrade checklist steps on the source system before beginning any database upgrade activity.
- Store documents and attachments in Microsoft Azure Blob Storage.
- Freeze all code and application configuration changes in the AX 2012 environment.
- If you're using a mobile solution in the warehouse to interact with Dynamics 365, whether you're using the standard Warehouse Mobile Devices Portal platform in AX 2012, you should move to the Dynamics 365 warehouse mobile application.

- **Cutover** **Figure 18-5** shows the ideal cutover sequence for bringing data into the production environment and validating the data. This works for Operations.

Dynamics 365 Customer Engagement

To go live with Customer Engagement apps, key stakeholders should perform a go-live readiness review to understand whether the plan covers all key activities, activities that can be executed ahead of time, and the plan to transition to operations.

Fig. 18-5



Post–go live, management of your solution is important when migration activities are complete and users are actively using the system.

Go-live readiness consists of understanding the plan that helps ensure all recommendations and best practices are being followed to anticipate any unforeseen issues during the last sprint.

When engaging with the FastTrack team, this review is conducted in a workshop format and includes all parties.

This section provides key information that enables customers and partners to perform their own go-live readiness review.

Test and acceptance

In the Prepare phase, the functionalities should be completed (code complete), integrations should be tested with all external systems (SIT), and a high-level deployment execution plan should be in place. This will trigger the solution test and acceptance activities. In this phase, the nonfunctional requirements (NFR) of the solution that were previously agreed to and signed off on—for example, form load times, search performance, and integration performance under the realistic production load—should also be validated.

Several tools are available to aid with building testing automation. One such tool is Easy Repro, an open-source library intended to facilitate automated UI testing.

Training and adoption

To help with in-app contextual training, guided help functionalities are available to give users a custom, in-app help experience tailored to specific usage and business processes.

The Dynamics 365 adoption guide is an easy-to-follow how-to guide on the best way to roll out Dynamics 365 to your organization.

Capacity management

Management of your solution is important, especially post–go live, when migration activities are complete and users are actively using the system. Storage usage and growth need to be validated; using storage above the limits has an impact on some features, such as on demand and restoring backups.

Monitor API limits because after go live, there might be integrations or bulk operations that cause peaks in the usage, resulting in API limit errors.

Data migration

You can use several tools—out of the box, ISVs, and custom built—to migrate data. It's important to follow best practices and include them as part of your migration strategy.

Various factors can impact data migration activities, for example, the service protection API limits that ensure consistency, availability, and performance for everyone. Keep these in mind when estimating the throughput and performing testing and the final data migration activities.

Requests limits and allocations can also have an impact and should be taken into consideration.

Production environment readiness

When using Customer Engagement apps, the production environments can be initiated at any point in time. It's important that apps in production are the same version as those in the development and test environments; otherwise, issues might occur that impact the release process.

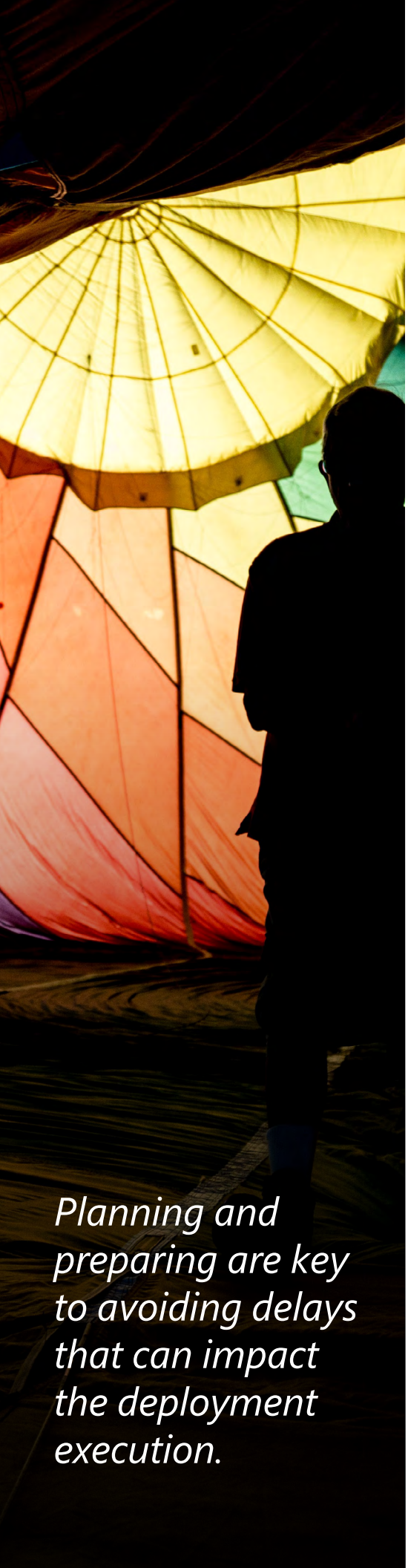
Several tools help you prepare for go live, including the following:

- Power Apps checker helps validate model-driven apps as well as canvas apps and should be used throughout the implementation.
- Microsoft Dataverse analytics enables access to metrics that help you monitor the solution, especially during the operations phase.

Continuous updates

In the words of Mo Osborne, Corporate Vice President and Chief Operating Officer of the Business Applications Group at Microsoft, "To enable businesses everywhere to accelerate their digital transformation, we are continuously enhancing Dynamics 365 with new capabilities."

Service updates are delivered in two major releases per year, offering new capabilities and functionalities. These updates are backward compatible so that apps and customizations continue to work post-update.



Planning and preparing are key to avoiding delays that can impact the deployment execution.

Upcoming features and capabilities can be validated months in advance with early access, before they're enabled in production environments.

To support customers and partners throughout the release schedule, the release plans and early access are documented and available well in advance.

Conclusion

As mentioned at the beginning of this chapter, go live is a critical milestone during a deployment—every plan and test is validated at this stage.

This chapter isn't intended to explain how go-live readiness and the cutover plan should be built; there are other resources that cover these topics. The aim of this chapter was to provide a summarized explanation of the key activities for go-live readiness and their purpose and the important takeaways, best practices, tips and tricks, and common pitfalls that can happen during this phase and their impact on how the new solution is received and adopted by end users.

We discussed the importance of training, adoption, and change management plans and how these plans contribute to the user experience and adoption of the new solution.

Planning and preparing are key to avoiding delays that can impact the deployment execution. All resources must be available in time for activities to be executed and completed. Environments must be created, configured, and operational on time.

Data migration can be complex. Realistic goals and good throughput are important to ensure that there are no delays that might impact the go-live date and that all tasks are executed according to the data migration plan. Ensure that the recommended practices to improve throughput are in place. Validate that throttling limits aren't being reached.

There should be coordination between the implementation team and external dependencies to make sure that expectations are aligned and

requirements met. Any external solutions must work as expected with the current version of the solution and platform and be aligned to the roadmap.

Support during cutover, go live, and post–go live is essential for all activities to run in the expected time. It's crucial to have resources available to mitigate any issues so that tasks can complete on time or with as slight a delay as possible. A poor support strategy for a successful rollout will impact the success of the rollout over time. It's important to create a list of responsibilities for first, second, and third lines of support as well as an escalation path and to transition knowledge from the implementation team to the support team. Ensure that there's coverage outside traditional working hours and on weekends for critical situations.

Ensure that application and platform monitoring is in place, and set up appropriate alerts so that the operations team can take necessary actions and notify users. A strong monitoring and communication plan has major impact on adoption and confidence in the solution. All critical application components should have a logging mechanism with a well-defined process to monitor the system and alert administrators.

A process needs to be in place post-deployment and post–go live to manage changes (such as upcoming updates or new versions of ISV solutions) and prepare users for changes and the adoption of new features and capabilities.

References

GitHub - Microsoft/EasyRepro: Automated UI testing API for Dynamics 365

Use solution checker to validate your model-driven apps in Power Apps

Microsoft Dataverse analytics

Microsoft Dynamics 365

Create guided help for your Unified Interface app

Dynamics 365 Adoption guide

Accelerate Dynamics 365 Adoption checklist

Build a champion program

Dynamics 365 release schedule and early access

Prepare for go live

Report production outage

Targeted release schedule (dates subject to change)

One Version service updates FAQ

Monitoring and diagnostics tools in Lifecycle Services (LCS)

Evaluate go-live readiness for Dynamics 365 Customer Engagement apps

Checklist

Go-live readiness

- [ ] Assess go-live readiness early and often through quality gate reviews to provide clarity on the status, readiness, completeness, and maturity of each activity and to help project stakeholders and the steering committee make go/no go decisions.
- [ ] Ensure that the solution scope is aligned with the solution that's going to be live and that it has been communicated, shared with stakeholders, and signed off on by the business, agreeing that the expected scope is covered.
- [ ] Ensure that the solution has completed all test cycles, which include system integration, performance, and UAT, with exit criteria satisfied and signed off on by the business stakeholders.
- [ ] Ensure all scripts and processes planned for the cutover migration are tested and signed off by the business stakeholders.
- [ ] Ensure external dependencies such as ISVs and third-party systems and services are aligned with the timelines and scope for go live.
- [ ] Plan all change management activities such as training, user support, user engagement, and communication, and implement all pre-go-live change management activities from Initiate through Prepare phases.
- [ ] Have a plan for monitoring and maintenance routine for production as well as for transitioning to support teams.
- [ ] Ensure the production environment is prepared and administrators and IT are familiar with monitoring, troubleshooting procedures, and avenues of support.
- [ ] Create a cutover plan that considers all dependencies, the timing of each activity, roles and responsibilities, instructions, verification steps, and any additional documentation associated with each activity.

Cutover

- [ ] Implement the cutover plan by completing all activities in sequence, satisfying the exit criteria, and including the sign-off at the cutover go/no-go checkpoint.

Case study

The gift: go live in a cloud solution

A toy company recently embarked on an implementation project and migrated all their processes from on-premises to Dynamics 365. They had integrations with Dynamics 365 Customer Engagement, Microsoft Power Apps, and Power BI. The company plans to implement Dynamics 365 Commerce and Warehouse Management mobile app in Microsoft Dynamics 365 Supply Chain Management in future rollouts, but they needed to move all financial data and their CRM system to the cloud in time for the end of the year, their busiest time.

At the end of October, the team was excited and a little nervous about the approaching go live. The impact of changing from a system in place for many years would be huge. The team was used to their manual and inefficient processes but were ready to move to a system that would automate and improve processes and increase revenue growth and profitability along with scalability. Change management was critical for success.

The implementation team needed to make sure that everything was ready for going live. With go live four weeks away, it was time to start reviewing their readiness for the implementation. SIT was completed, and UAT and performance testing were almost completed. The system seemed performant—although not optimal, it was considered ready for production.

The team was ready to start the go-live review, to switch to the Prepare phase. They prepared the go-live checklist and asked themselves the following: Everything was thoroughly tested, but was the system really ready? Were the users ready? Was all the infrastructure ready? Were all the systems up to date? Was the mock cutover planned? Were the ISVs working well?

Concerns came up over risks and issues identified during the assessment. For instance, they realized that the mail service and other services were in another tenant, so they needed to perform a tenant move to enable single sign-on for their users. In addition, Microsoft was planning an automated version release by their go-live date, so they needed to update their environments to the latest version. This would necessitate conducting another smoke test and verifying that the ISVs worked correctly with the version. They missed dates because they were focused on UAT and making fixes and addressing new requirements identified during the testing phase.

The team realized that open transactions weren't ready to be closed in the old system and therefore they created them in the new one. The data migration plan wasn't solid. In addition, performance was weak, so the team needed to troubleshoot the code.

Business stakeholders made a go/no-go decision after this assessment. It wasn't an easy decision, but they decided not to go live. Despite all the heavy lifting they had done, they didn't have a mock cutover plan, which they realized was necessary to complete the readiness process.

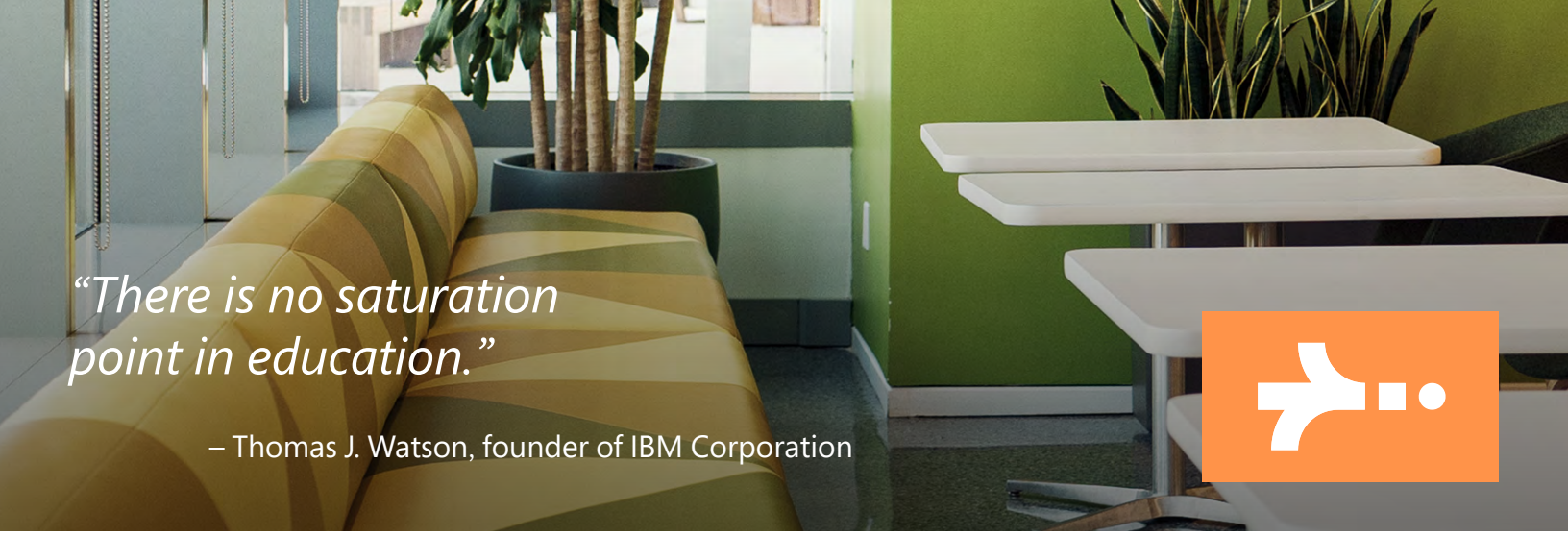
Would they need to change the go-live date that was so close to the holiday season?

What could they have done better? They couldn't have changed the date because they needed to be ready for the holidays, but they could have planned the go-live date earlier so that they had more time for the ramp-up and to address any delays or issues. They could also have had an earlier assessment review, with UAT almost complete. The timelines were tight, and they had the choice to go live with what they had, which was risky because that might mean stopping operations in the

middle of the busiest season, or move the go-live date to January—December wasn't an option. A third option was to hire more resources and work more hours to try to deliver on time, which was risky because the ramp-up would be during November. A big-bang go live wasn't optimal for this situation.

The team learned that it's important to start the readiness review on time and to schedule sufficient time and resources. Additionally, it's crucial to have a solid support plan for production. The importance of the Prepare phase also shouldn't be underestimated—plan with enough time to mitigate any issues and risks.

19 Guide Training strategy



Introduction

At its core, the goal of a training strategy—and training—is to ensure that all necessary users of your system are educated on the new application so that their knowledge of how to complete their work results in successful user adoption following go live.

In this chapter, we define a training strategy and determine at a high level what components you need to embark on a methodical approach to a successful training execution for your Dynamics 365 implementation. Each of these areas is covered in detail in this chapter.

For a successful training strategy consider these main areas:

- Training objectives
- Training plan
- Scope
- Audience
- Training schedule
- Training material
- Delivery approach
- Assumptions, dependencies, and risks
- Training as an ongoing process
- Training best practices

Training is not the only factor in meaningful user education and adoption; empowering users to perform their necessary tasks in the system correctly and efficiently should be the ultimate “why” in the development of a training strategy. As part of user adoption, organizations should strive to train in a way that gives users confidence in the application and inspires a sense of delight when using the system.

In this chapter, we cover several high-level objectives, as well as examples of more organization-specific objectives, that should be included in your organization’s training strategy. A proper training strategy should center around the creation and execution of a comprehensive training plan. Furthermore, the training plan should align to the broader training strategy of your Microsoft Dynamics 365 implementation. We discuss how to create an appropriate scope for your organization’s training as part of this plan, as well as how to confirm the different groups of users that need to be trained.

We cover training topics that should be part of your organization's overall project plan, and how best to keep your training plan and project plan coordinated to avoid significant issues. This chapter also dives into guidance surrounding different types of training material and their effectiveness, as well as how to create a training delivery plan and conduct effective training that can serve all your users (before and after go live).

Dynamics 365 is an ever-evolving and growing application. Because of this, training is an ongoing process that requires constant adaptation and updates, and we cover how to best prepare your organization for this continual update process. We also recognize that training strategy and guidance can vary depending on your choice of Dynamics 365 application. We provide suggestions on how best to successfully adopt a training strategy, whether your organization is implementing applications built on Dynamics 365 Customer Engagement, Finance and Supply Chain Management, or both.

Lastly, we discuss best practices by highlighting the behaviors and strategies that we have seen organizations successfully employ when developing a training strategy. We share our collective experience regarding these best practices toward the end of the chapter, in the hopes that you will carry them forward in your own projects.

By following the guidelines laid out in previous chapters of this book, your organization should be well on its way to rolling out a successful Dynamics 365 application that fulfills the needs of your business and users. Successfully training your users on using modern technology, as well as understanding business processes that they will need to incorporate to do their jobs safely and efficiently, is integral to any organization's training strategy.

Training objectives

One of the first things that your organization should consider when beginning to develop a strategy surrounding training users, and a plan behind it, is the objectives. Defining the objectives of a successful training strategy is key—and it can help shape the crafting of a training plan as well as lead to a more successful rollout of training itself.

At a high level, every Dynamics 365 application training strategy should include some version of the following objectives. Many change management objectives can also be addressed by ensuring that your training strategy explicitly addresses these training-focused objectives:

- Employees will gain necessary knowledge required to use Dynamics 365 applications as a part of their day-to-day jobs in a safe and efficient manner.
- Training should be meaningful and not overwhelming, to provide users with an efficient way to do their jobs.
- Training should help support business goals for the application (in other words, employees understand the "why" behind training materials and scenarios).
- Training should be inclusive of everyone (for example, experienced users as well as new hires) so that all users of your application receive the education necessary to do their jobs.
- Training should be an ongoing process that does not stop when the system goes live.

Your Dynamics 365 implementation will have additional training objectives that are specific to the applications you are developing, and to your organization. It is important that you work to define these objectives as they, along with the more general objectives of training outlined earlier in this chapter, should formulate the "what" of your training strategy goals. An effective way to evaluate your organization's training goals is to follow the SMART acronym—that is, specific, measurable, achievable, realistic, and timely. While it might not be possible for every training objective to meet every one of these metrics, we recommend that you create training objectives that are relevant to your organization's goal of user adoption and fit the outlined criteria.

What not to do

As a counterpoint to the recommended strategies, let's spend a little time discussing the training objectives that an *unprepared* organization might create.

In this example, let's say the organization creates the following objective: "**Objective:** Dynamics 365 Field Service users should be trained on the

mobile application." While this objective captures at a basic level what needs to be done, it falls short of accomplishing what a proper training objective should. When rewritten using the previously mentioned criteria, and separated for increased detail and specificity, the objectives become much clearer:

- **Objective:** 100 percent of Field Service technicians attend the in-person mobile training session or view the complete online seminar series prior to system cutover.
- **Objective:** All Field Service technicians finalize the Work Order completion process on their mobile devices without assistance, and are able to complete the lead-generation process on their mobile devices in under three minutes.

These rewritten objectives also serve as good examples of making sure your training objectives specifically relate to the application being deployed. A poorly written training plan often does not include application-specific objectives; in addition to not following the SMART philosophy, these objectives aren't thorough enough to capture the essence of the goals of training. Clearly defined and measurable training objectives also help an organization determine the effectiveness of its training. It is significantly easier to judge the success of training in your organization if you can track against objectives that have been well-written. Later in this chapter we discuss specific ways to assess the effectiveness of your training; this process starts with creating proper training objectives.

Ensuring that your users are comfortable with the most challenging obstacles to user adoption is important to achieving training success. While your training objectives do not need to specifically reference tasks or challenging business processes directly (as these would be too specific and thus not useful), they should reflect knowledge of areas of your application that may require additional attention from a learning perspective. The mobile objective in the previous paragraph is a good example of this strategy because we recognize that Dynamics 365 mobile apps can represent a significant change in business processes for users and, therefore, could require additional attention from a training and user adoption perspective. Thus, a key objective should be that all users receive proper training on this specific job function.

Creating a training plan

Proper training is critical to user adoption. Organizations must develop a training plan at the start of the project and engage resources from the beginning.

It's important to remember that managing a training plan is an iterative process. As you progress in the project and learn more about your business requirements and training needs, you might have to update your plan. This sample training plan charter gives you something to start with.

We recommend you put together a training plan that, at a high level, consists of at least the following elements (as illustrated in **Figure 19-1**):

- Training objectives
- Scope
- Audience
- Training schedule and resource availability (project planning)
- Delivery approach
- Validation of training success/objectives
- Assumptions/dependencies
- Risks
- Training environment management
- Training materials
- Training as an ongoing process
- Training resources

Fig. 19-1 Training plan elements

Just as the project plan is key to project timelines and execution, so too is a training plan key to training timelines and training execution. End-user training is one of the last tasks on a project with

dependencies on several key items, such as a near-complete solution, good quality representative business data, and reserved time from end users. If a project is running even a little late, the final preparations for training often get squeezed. A robust, well-thought-through training plan that reduces the number of items in the critical path is essential to avoid last-minute compromises and lower-quality training outcomes.

Scope

A crucial element of ensuring that your organization's project has a proper training strategy is defining an accurate scope for the training. Here are different areas that cover the breadth of scope that you will need to build into your training plan:

- Business processes completed by end users, derived from your user stories and functional requirements
  - Example: Users must be able to create and process an invoice from end to end in Dynamics 365 for Finance and Operations.
- Any nonfunctional requirements that require training users for the requirement to be met
  - Example: Users should have the ability to provide feedback directly from the application.
- Any administrative tasks that users might need to accomplish while in the application, separate from their business-driving work
  - Example: Users must be able to configure and use the Advanced Find tool inside Dynamics 365.
  - Example: Users should be able to upload templates using the Dynamics 365 data management tool.

There could be other areas of your application that should fall into the scope of your training. If there is a task or process that your users need to accomplish while working directly or tangentially with your Dynamics 365 application, it should probably fall into the scope of your training, as you should assume there are users who need to be trained on that topic. An example of this would be training project team members during your implementation who will need to work inside the application from a development or configuration perspective.

Once you have the full set of business processes, requirements, and tasks, you can use them to create a true scope, consisting of actionable training areas, for your training. There might not be a one-to-one relationship between them, as there may be overlap between processes as well as additional training topics that need to be added. This is our guidance on how to avoid overlap and how to add more training topics.

- **Avoiding overlap** If you have 10 sets of business processes that are different in terms of function but almost identical in terms of system execution, and each takes 30 minutes for a user to complete, consider setting your system training areas to represent the similarity between completing these in your application, rather than to represent the disparity between business processes. You can still reference different business functions and even take time to walk through any differences in execution, but it is important that the scope (that is, each training area) is framed in such a way to be efficient and not duplicate training effort.
- **Additional training topics** If you require users to switch between their mobile and desktop devices frequently, with minimal delay between each, it might be important to cover this transition even if it is not necessarily covered in any business processes or other requirements or tasks.

The next step in setting the scope for your training is to categorize and prioritize the list of training areas that you have created. As an example, some business processes in your new application might be different from the "as is" processes in your organization's current application. This is important for a few reasons:

- System training and business process training are two separate efforts, even though training might address both types.
- Training surrounding changed business processes should be given special attention in training sessions and in training material.

It's common for business process training challenges to affect system training effectiveness. As part of your scope of training, and combined with your training materials and delivery, you will want to separate system training on the new processes and application from topics that veer more into organizational change management.

Also, you should prioritize, or at least highlight, any training related to critical nonfunctional requirements that have impacts on legal regulations or company policies. For example, if your organization is legally required to fill out fields on your sales order form in a particular manner, this section of training should be emphasized so that users are aware of its importance.

One process that might help determine priority of training areas is the creation of a change-impact assessment, whereby you detail the changed areas, the topics and stakeholder groups that are affected, and the impact that the changed areas will have. This will allow you to have a consolidated list of training topics, an estimated list of people that will need training on that topic, and the beginnings of a prioritization around each.

Figure 19-2 shows an example of a change-impact assessment.

Audience

Now that the objectives for your organization's training strategy have been defined, along with the scope of content that training should contain, we can define the next piece of the training puzzle: who to train, and what type or types of training they should receive.

Fig. 19-2

Change impact	Training topic	Impact (1-5)	Action
The purchase department's legacy system will be replaced with Dynamics 365 Supply Chain Management and business wants to use the standardized process in Dynamics 365	Procure to pay business process	3	Trainers will hold a series of webinars detailing differences between the legacy system used for purchasing and Dynamics 365 Supply Chain Management
Field technicians are moving from a pen-and-paper-based work order management system to a mobile application	Using the new mobile application to complete a work order	4	Because of the impact this change has, multiple live training sessions will be held for technicians to ensure proper education

Who receives training?

It is critical to identify the different groups of people who need training. Dynamics 365 offers excellent role-based access control and application segregation as a means of providing access to your users based on how they need to use your application. As a result, it is important that all employees who will need to use the product are trained in the correct way, on the correct topics.

The following section identifies the core groups of people who should be trained on your Dynamics 365 application. Note that each group of users can be broken into subgroups. Breaking down these users into personas is important to successfully executing training; if personas are incomplete or incorrect, your organization might not complete proper training on the correct business processes or, worse, it might not complete training at all for specific user groups.

System end users

System end users are the people who will use the system daily and drive direct business value. Your application could have multiple types of end users, separated by licensing, security role, or even the application itself (Customer Engagement versus Finance and Operations users). Each of these groups will require different training. This group of users might cover multiple departments and job functions; as a result, the training should both be general, as to provide high-level knowledge of your application regarding key core entities that all users may need to access, as well as specific to aid these users in their specific job functions.

Fig. 19-3

As part of your training plan, be sure to identify each business role that will need to use your application differently and will require training. This is related to, but distinct from, creating groups of users based on security role. Certain personas might have identical security roles in Dynamics 365 but different job functions; these groups of users will need to use your application differently and will, as a result, require different training.

As an example, consider a salesperson and a sales team leader (see **Figure 19-3**). Even if your organization has defined that their required access in your application is similar enough to warrant identical security roles, it

is important that these groups of users are treated distinctly during this portion of your training strategy and receive correct training. Salespeople will need to be trained more on the "doing" functions of your application, like creating records, updating records, and sending emails.

Sales team leaders, on the other hand, might work on similar tasks as salespeople, and they might require some or all of the same training, but they will also need additional types of training, such as reviewing and approving records, and creating and viewing charts and dashboards.

In many organizations, certain end user roles have higher turnover than other personas receiving training, so your training material should be structured in a way that can help staff onboard easily.

Another benefit of being specific when defining training personas based on business role rather than security role is the value it can provide for new hires. Having a new hire in a specific business role undergo an exact set of training based on what was identified previously in this chapter can make onboarding significantly easier and reduce time in trying to match job titles to security roles, and then back to business processes in your application.

Trainers

Trainers are a special category of users that require training. They should be knowledgeable in all areas of the system and are responsible for developing training material and training other users. "Train the trainer" is a common approach to onboarding this type of user to your application. Depending on the size of your organization, you might have several trainers. Since the goal of your trainers is to create material and conduct training for your other users, it is important that at the end of "train the trainer" sessions, your trainers are aware of and able to perform their job function—training other users—effectively. The goal of training these users is different from that of your end users; trainers should be part of the training strategy process and will be advocates of the new application.

Super users

Super users are an elite group of (usually) end users who act as champions of the application. This group is key to driving excitement and adoption of the new system. These users are often involved in early feedback cycles with the project team. As subject matter experts (SMEs),

Executive buy-in is important for the success of a new application.

they become educated on the application early on so that by the time end-user training begins, your super users are experts in various business processes. As part of your training delivery plan, your organization can use super users strategically (by having them attend training sessions and serving as application advocates) to aid your trainers in their job functions. Additionally, these super users can act as a "first line" to help answer employee questions—keeping your support desk from being overwhelmed and at the same time keeping morale high.

We have mentioned several times that meaningful user adoption is critical to the success of an application and relies heavily on a well-executed training strategy; super users are instrumental in this process. If your organization is deploying several different applications, it is recommended to have multiple super users, spread across each application, and corresponding to their area of expertise.

Project stakeholders and executives

If your project stakeholders and executives need to be able to access and use your application, they need to receive training as well. This training should be tailored to their specific role.

Executive buy-in is important for the success of a new application, and meaningful user adoption applies no differently to this group of users than it does to your end users. To take advantage of the additional effect that executive buy-in may have, you should consider "visibly" training these users so that other members of your organization are aware that training is ongoing. You can also notify these stakeholders of training that other roles are expected to undergo, which can help from a buy-in perspective.

Support desk and administrators

Another critical group of people that will need training will be your support and administrative users. While these users might need a certain amount of training on the application and certain business processes, they will need to become familiar with the application from a maintenance perspective. Training for these users will focus on tasks related to system maintenance, troubleshooting, and administration.

Training scope

Once you have defined the distinct groups (and subgroups) that should receive training, you should work on matching the different training areas defined in the Scope section of this chapter with the groups of users defined earlier. Each group of users will require a specific set of training (based on training areas). We recommend creating a matrix in your training plan—with training role against training subject area—that you can refer to when creating training materials, planning training delivery, etc.

You should structure your assignment of training topics to users in a way that is comprehensive but not superfluous.

When determining which personas need which types of training, refer again to your user stories as a baseline. You should also consider which groups need training in non-functional areas and administrative areas. Note that certain groups of users might require different levels of training on identical subjects. For example, trainers might require only basic training on Dynamics 365 administration (to assist other users during training and to implement training tools), while administrators and support desk personnel might require advanced training on the same topic. You should structure your assignment of training topics to users in a way that is comprehensive but not superfluous.

Training schedule

Successful integration of a training strategy into the main project effort requires coordination with project management resources. Your organization's training strategy does not exist independent of other ongoing efforts in the project, and it is important that key milestones, resource requirements, dates, and tasks are acknowledged and accounted for as part of your training plan and are incorporated as tasks or dependencies in your overall project plan.

Regardless of the size of your organization and complexity of the application being deployed, you must consider the following training-related factors. We have included here examples of common questions project teams encounter as a part of their training initiatives. If these questions are applicable to your project, you will need to

address them as part of your training strategy and project plan. Note that this is not a comprehensive list of questions that your organization will need to answer, but this should help you get started:

- Project team resources
  - What is the availability of the trainers and super users?
  - Who are the super users, and if not defined, when will that decision be made?
  - Are there external resources involved in this process that will require onboarding or administrative setup before they can begin their work?

- Trainees
  - Who is necessary to take part in training activities, and when will these activities happen?
  - Are there any resource dependencies (e.g., Resource A is required before Resource B)?
  - How will training, and attendees, be staggered so that there is no gap in coverage for day-to-day business functions?

- Dates or key milestones
  - When should training begin for distinct groups of users?
  - What dependencies do these trainings have?
  - Are any project-related activities dependent on training?
  - Is training reliant on the creation of material or development work, whose schedules could shift?
  - What milestones should training include, and are there any impacts of these milestones?

- Training plan
  - When will the training plan be completed?
  - Who will review it, and what will be the plan for updating it if necessary?

- Training execution and feedback
  - When, and where, will training occur?
  - Will training have an impact on other project efforts?

- If there are subsequent trainings, how will feedback be evaluated and what is the cycle for processing this feedback?

Fig. 19-4

Question

Are any project-related activities dependent on training?

Answer

We often recommend conducting training prior to your user acceptance testing (UAT). The major benefits of this order are:

During UAT, users are already aware of how to navigate through the system and accomplish tasks. And as a result, there should be fewer items logged as bugs that should be listed as training issues.

UAT serves as a measure to validate the effectiveness of training. If a large number of training-related issues are identified during UAT, it may be necessary to make updates to your training material and even potentially conduct future training on topics with which users are struggling.

As you can see, many questions related to a training strategy have an impact on, or are reliant upon, activities that occur separate from traditional training activities. In **Figure 19-4**, we examine one question from this list to underscore the importance of asking and getting answers for these questions as part of your project planning.

This example shows why it is crucial that your training plan and overall training strategy be reflective of and connected to your organization's project plan, and that updates to either plan are reflected in the other so as to not create conflicts, which can lead to project delays.

Training materials

There are multiple consumable types of content for Dynamics 365 applications that your organization can create. Depending on your project, you could create multiple types for use in different trainings. Here are a few common examples of the types of training materials that an organization could create.

Documents Your organization can create written documents that guide users through the application and educate them on business processes. These documents can also serve as reference material. This type of material is the least interactive but is the easiest to create and update.

Videos Trainers can create videos that explain key training areas in detail and walk through the application, providing guidance on how users can navigate the system and complete their job tasks. Videos can be more helpful than documents for many users, since they "show" rather than "tell" certain features of the application. Videos can also be recorded in small chunks that represent discrete processes in an application, as opposed to creating one longer video that walks a user through an end-to-end flow. This benefits trainers from a maintenance perspective; instead of having to rerecord the entire video when changes are made to your application (and create rework around having to record training

for concepts that have not changed), they can focus on updating specific videos around new features and functionality.

Microsoft Learn Microsoft has published several labs via the Microsoft Learn portal that provide a wealth of knowledge on Dynamics 365 applications, as well as other applications on which your organization might require its users to be educated. Microsoft Learn allows the user to follow any number of predetermined paths specifically designed for different user personas. Individuals can also create their own knowledge path that is specifically tailored to their learning requirements. Users can learn on their own schedule and can browse videos related to topics when necessary and engage with other community members.

Guided help Dynamics 365 includes the ability to create custom help panes and guided tasks, out of the box, to assist your users in walking through the application and completing basic tasks in the system. Guided help is easy to set up and implement and does not require an additional solution on top of your application. Additionally, Dynamics 365 applications can install the Guides application, which allows for visual or holographic displays to show step-by-step instructions on tasks that your users need to perform.

Virtual material via a digital adoption platform Built on top of Dynamics 365 and allowing users to directly interact with the application while training, digital adoption platforms are the most interactive form of training material. This training material can be easily updated, and once published, it has an immediate effect for all users who need to view it. It reduces the need to update physical documents or re-record parts of video material. Digital adoption platforms can come in multiple flavors. One effective method is a tool that lets trainers create multiple "scripts" that walk a user through a business process in Dynamics 365. For example, say that all users must learn basic navigation through an account record, but only system administrators must know how to customize a price list. Earlier in this chapter, in the "Scope" section, we talked about reducing overlap; using digital adoption platforms is an excellent way to manage scope in the creation of training material. Instead of having to explain the same material twice (in the case of video trainings that cover similar subjects) or create

multiple reference guides or a single complicated guide (in the case of written training material), a trainer can create one set of scripts and apply them to different groups of users.

What to consider when selecting training material

Be thoughtful when choosing the type or types of training material to create for your training audience. Consider the benefits and drawbacks of each. For example, if your end-user base consists mostly of users who do not have considerable experience working with web-based cloud applications, you might want to consider a more hands-on approach—a digital adoption platform—to help train your users. On the other hand, system administrators and support desk users might find documents more helpful, as they would rather have the large volume of written information that do not require the more hands-on training that digital adoption platforms provide. When creating training content, keep in mind that training is ongoing. Your organization can start to create training material and training early on in your project, and then refine it later with minimal rework needed. In the world of cloud applications, where software is constantly changing, it is important that your training strategy prioritizes being adaptable, so that you can update materials easily as a response to application changes.

You should also consider language and accessibility needs in your organization. If you have a multi-geo or multi-language deployment, any training material you create might need to be available in different languages. This material must also comply with any federal, state, or local regulations regarding accessibility so that all users have an equal opportunity to consume effective training material.

Who will develop and review the content?

During the creation phases of your project plan and training plan, you'll need to identify the trainers, SMEs, and business process owners who will assist in developing the training content. Trainers will usually create content, which will subsequently be reviewed by the SMEs and/or business process owners. Depending on the importance of the

course, as discussed earlier, you might want to consider bringing in multiple people to review the material.

How long will it take to develop the content?

Your organization must make sure to allocate time in the project schedule and training plan to design, develop, and review content. The length of time required to create training content will depend on the complexity of your application and the type or types of content to be developed. To estimate the time, we recommend that you identify the scope of training areas and subsequent material that must be created based on those areas, and the format or formats the training material must take. You can then estimate total time to create training material based on experience or an extrapolation calculated on how long it takes a trainer to create a subset of that material.

When is the content needed?

Your project plan should list when content creation begins and your training plan should specify when training material will be available to certain groups of users. These two schedules should be coordinated to ensure that content creation begins on time, and to ensure the material is ready when needed. End-user training can begin roughly when development is completed (that is, when the system is code complete and no changes to the application are anticipated).

Where will the content be stored?

If more than one person will be editing content, your organization might need a versioning strategy.

We recommend your organization employ a content storage control method, especially if multiple types of content are being created for groups of users. Training content should exist in a single centralized source instead of being passed around by email or shared links. This also ensures proper source control and the ability to institute access control for your user groups. Additionally, if more than one person will be editing content, your organization might need a versioning strategy.

Delivery approach

As a subset of your training plan, your organization should create a training delivery document that contains specifics about the actual

execution of training itself. While many of these topics are covered in the overarching training plan, it is important to have a specific plan for training delivery to ensure that your organization is prepared to execute training.

We'll now discuss the topics that your training delivery document should cover.

Who will be involved, and how?

We have defined the audience for training in the "Audience" section earlier in this chapter, and explained the details of incorporating resource scheduling into your project plan in the "Training schedule" section. To help you create your training delivery document, we will go a step further in organizing these resources into training sessions. This is important if your organization will schedule multiple training sessions at the same time. In addition to allocating your trainers and trainees, you will also need to assign them to the correct training based on the content to be covered.

What methods of training should the organization conduct?

Like training material, diverse types of training suit different situations and users, so it's important to consider which training or combination of trainings best fit your organization. Consider the advantages and disadvantages of different methods. **Figure 19-5** shows common types of training, and the anticipated advantages and disadvantages of each type.

When should training occur?

As a part of your training delivery plan, you should decide when different types of training will occur. In an earlier section of this chapter, we discussed how super users and trainers should be trained in necessary areas of your application earlier than your end users; this type of consideration should be built into your training delivery plan. Plan to conduct training for your end users late enough in the project—post-development and pre-UAT—so your users don't forget concepts and tasks that they have been taught during training, but also be sure to leave enough time after training occurs to accommodate any delays in your project or training schedule.

Fig. 19-5

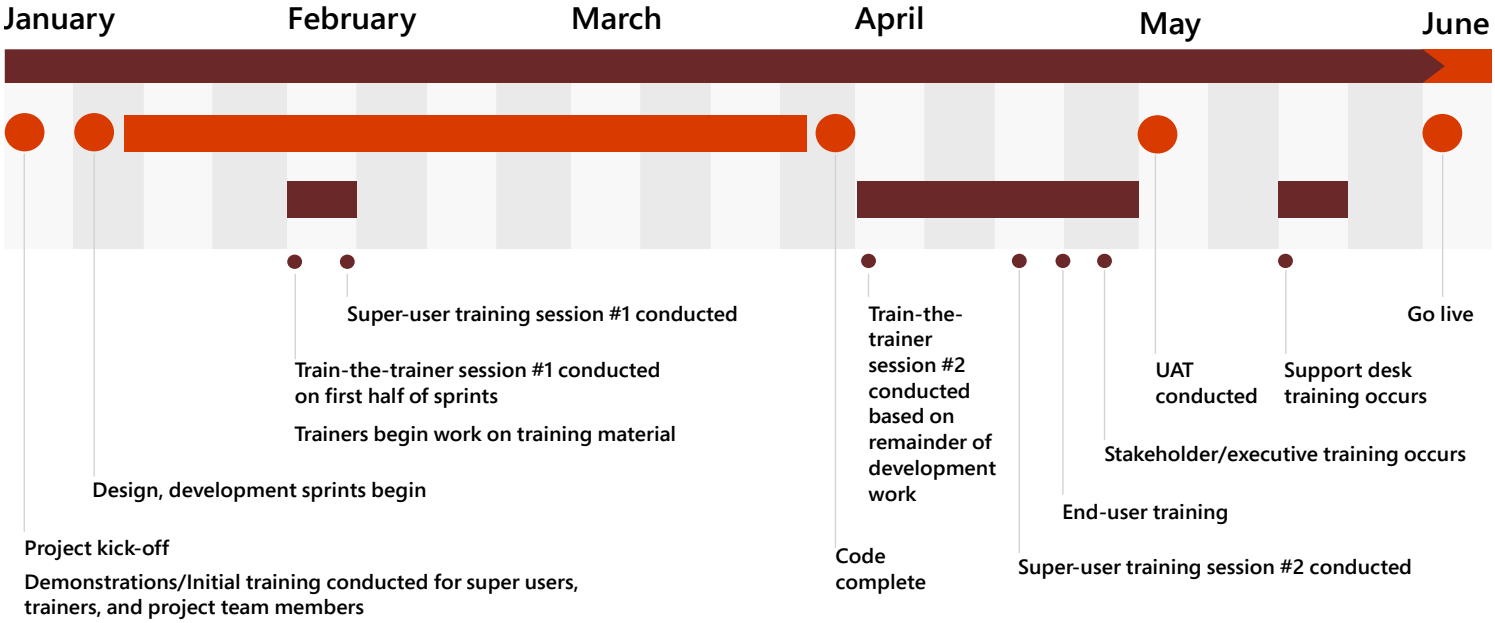
Type	Most effective when	Advantages	Disadvantages
Live training (in person or virtual)	The content is critical for business function and you want strict control over delivery. Content contains non-functional requirements, or business processes that require specific devices.	Better interaction between trainer and participants as well as between participants. Immediate feedback. Collaboration in business processes is significantly easier (i.e., front office and back office need to work together to complete an order).	Scheduling challenges. In-person variants of this type of training require a physical room with computers, since Dynamics 365 is a web-based application. In-person variants of this type of training are limited to the number of people per room, per session.
Interactive web-based training	Content is moderate in difficulty. Business processes are best learned through repetition and ongoing training, since users' access to the application is necessary.	Web-based content can be built once and consumed by an unlimited number of users. People can train on their own. No scheduling requirements and no physical location needed.	Trainings are less effective than in-person, since there is less interaction and the end user faces more distractions. Trainings are not usually real-time, meaning any questions or challenges that users face during training might not be answered quickly. Web-based training, when built with digital adoption tools, can be costly and also require technical maintenance.
Self-paced training	Content can be easily explained in a user manual or video series. Content is best consumed as written reference material that can be used as needed.	Written and video content can be updated quickly as future releases and updates occur on your Dynamics 365 application. Training content is easiest to create.	This is the least interactive form of training, meaning users who rely on interaction either with a trainer or with a web-based application might struggle to learn when having to read/watch content.

We believe that it's never too early to begin training super users and trainers. Since the main goal of training these users is to help them become familiar with your application early in your project, you should start conducting training with them (whether formal or informal) during development and continue it as an ongoing process. These early training sessions should familiarize super users and project team members with basic application functions and serve as a backbone for future train-the-trainer sessions.

Figure 19-6 is a high-level sample schedule that shows training sessions for each important group that needs to be trained, alongside key relevant project milestones:

Trainers and project team members should be aware of ways in which training differs depending on the size of your organization. Larger organizations with more users are likely to require multiple personas and an increased number of business-use cases. Larger organizations, especially international companies or companies with numerous locations, might require different training sessions and venues, and they might need to occur at different times in your project. In addition to making sure training materials are available in multiple languages, employing multiple trainers or multilingual trainers is important.

Fig. 19-6



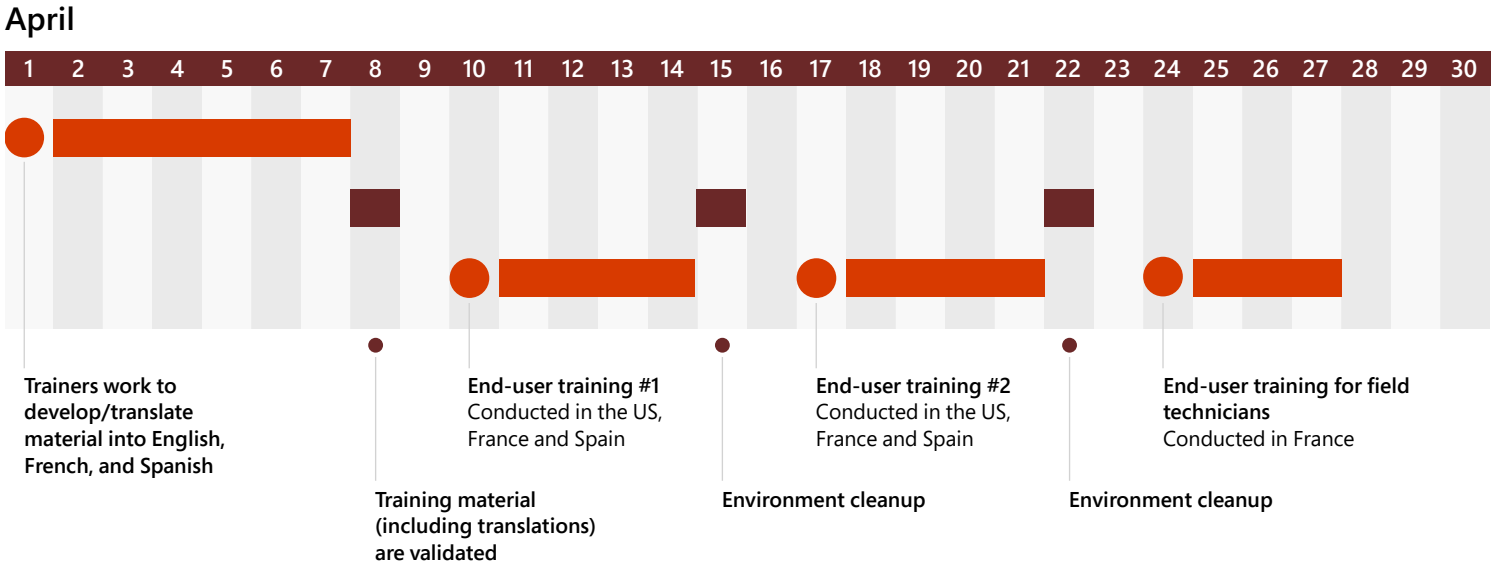
In the sample project plan seen in **Figure 19-6**, end-user training takes place over the course of a single week, in a single location. If this organization were international and multilingual, most of the items in the plan become more complex. To illustrate the increased complexity, we consider what end-user training might look like in a multinational organization in **Figure 19-7**.

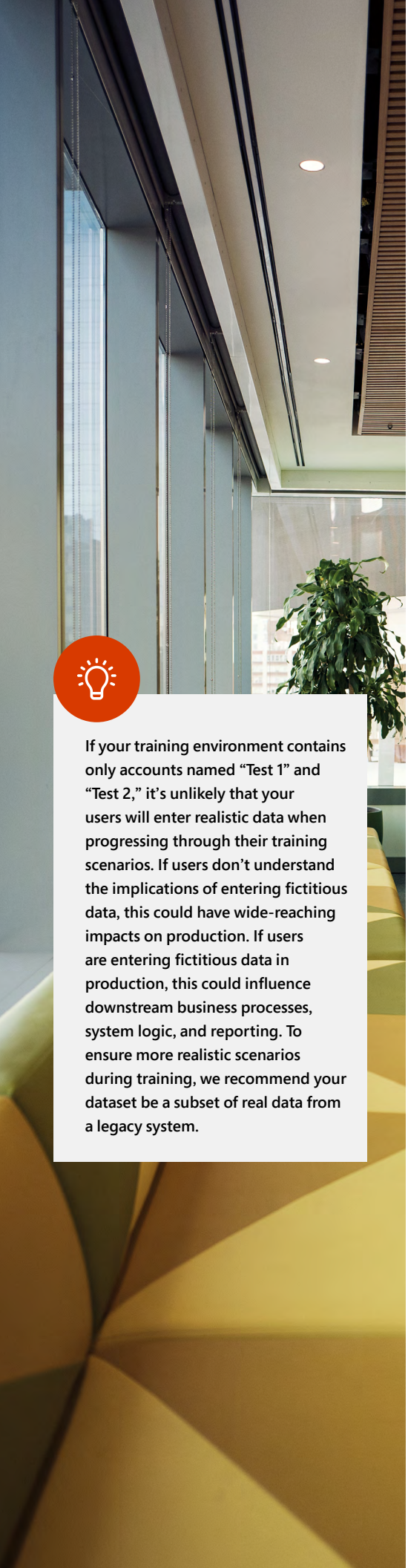
As this figure illustrates, training schedule and planning complexity increase dramatically when the scope of the implementation increases. Instead of taking roughly a week, this updated training schedule now takes almost a month, in three different countries/regions, and requires drastically increased coordination of resources. If additional phases with different business process areas need to also be covered, this would further increase the complexity of training. We hope these examples are useful in demonstrating the value of planning the when, where, and how of training and why it is important to understand that your organization should be prepared to make additional investments to train its users successfully.

How should your organization set up and maintain a training environment?

If your organization has created a Dynamics 365 environment to train users as part of a broader environment management strategy (covered in Chapter 9, "Environment strategy"), using application lifecycle management (ALM) best practices (covered in Chapter 11, "Application

Fig. 19-7





If your training environment contains only accounts named "Test 1" and "Test 2," it's unlikely that your users will enter realistic data when progressing through their training scenarios. If users don't understand the implications of entering fictitious data, this could have wide-reaching impacts on production. If users are entering fictitious data in production, this could influence downstream business processes, system logic, and reporting. To ensure more realistic scenarios during training, we recommend your dataset be a subset of real data from a legacy system.

lifecycle management"), you will want to make sure this training environment is deployed with code and data that is representative of production use and is of adequate quality to meet your training objectives. Both are extremely important points:

- If your organization's training environment does not have up-to-date code and application features, your users will be left confused and frustrated, which could decrease the overall effectiveness of your training. Your users might feel the training was unprofessional or incomplete—especially if the quality of the training environment is insufficient to accomplish training objectives.
- If your organization's training environment does not have sufficient and relevant test data, your users might fixate on the quality of the data and not on the training material. This will lead to improperly trained users from multiple standpoints:
  - Because users are distracted, overall training effectiveness will decrease. This could have an impact on user adoption.

Poor data quality in your training environment might impact the training's real-world relevance. One problem we see frequently in poorly conducted training sessions is that users complain the data is not realistic. This in turn can have an impact on the quality of the data that these users subsequently enter during training, which can affect the quality of the data that these users enter once the system goes live.

How to assess competency and effectiveness of training?

One of the most effective ways to improve training is to collect feedback and use it to make enhancements. Training is not meant to be a static exercise in which material is created once and delivered the same way in every iteration; it should be a work in progress that is updated frequently. Once a training session is completed, there is still work to be done. Many organizations require multiple training sessions, and all of them present opportunities to revisit, evaluate, and improve future training sessions.

You can measure the effectiveness of your training through several different recommendations. The first is by assessing your users' learning at the end of, or shortly following, training—in particular, prioritizing the processes that they will use most frequently in their jobs. It's much more important that sales users understand how to create and

update an account record in Dynamics 365 without assistance if it's a process they will need to do every day. Such prioritization should be emphasized during the creation of any assessment method you choose. Earlier in the chapter, we discussed how creating solid training objectives would help you assess the effectiveness of your training. We can now take those training objectives and use them to form the basis for how we will assess our users, so long as the training objectives we created contain the most critical topics that we are training users on.

Many larger organizations create exams to assess knowledge of their users or they might even create a certification program following training to assess readiness at a user level, team level, or even larger level, such as a business unit, subsidiary company, or geographical region. Trainers can create metrics based on exam results and certification statistics to evaluate training quality. If certain groups have scores or statistics that are lower than desired, the organization can work to improve future training or host follow-up trainings for these groups.

Accuracy is important to ensure data quality, but it is also important to know if certain business processes take your users 10 minutes or 2 minutes to complete.

Day-in-the-life testing, which can be used to measure real-world scenarios and the effectiveness or speed with which your users complete tasks, also measures training effectiveness and organizational readiness. Accuracy is important to ensure data quality, but it is also important to know if certain business processes take your users 10 minutes or 2 minutes to complete. This could be used to measure your organization's readiness for the new application. If certain business metrics must be met regardless of the application used, make sure these metrics are met within the Dynamics 365 application. As an example, if your call center staff is expected to process at least six calls an hour, it is important to ensure that your staff has been trained to complete a single end-to-end call process in your application within 10 minutes.

Another way to measure the effectiveness of your organization's training is by analyzing the results of your UAT. As we briefly discussed in the "Training schedule" section, you can use UAT as a barometer to gauge user competency of your application by looking at the number and types of non-system-related bugs that are logged by users, and by measuring (usually informally, since the goal of UAT is accuracy, not speed) the amount of time users take to complete the processes they

are testing. If you are finding that your testers are logging many items that are not system bugs but rather due to knowledge gaps on the function of your application, you might conclude that training in those areas needs updating. It could also mean that additional user training in those areas is required.

Similarly, after go live, you can use helpdesk statistics to determine the longer-term effectiveness of training. If users are submitting help desk tickets, or requesting assistance, on specific areas of your application, it's a good idea to review training or training documents related to those features to see if they should be updated. Again, if necessary, additional user training might be needed.

Another common approach to collecting feedback on training is to conduct pilot training sessions with a limited audience. You can invite super users to attend these pilot training courses and gather feedback to improve. A side benefit of conducting these pilot trainings is that it helps super users in their process of becoming experts in your application.

Question

What if, during a training session, a critical bug is discovered that blocks training from going any further?

Recommendation

We believe your training environment should be treated similarly to a production environment during a training session. Bug fixes should not be made directly in production; likewise, bug fixes should almost never be made directly in a training instance while training is taking place. Depending on the severity of the bug, a work item should be created for a developer in their respective development or hotfix environment, and the bug should be fixed and tested before being pushed to downstream instances (including training), as per standard ALM processes.

If the bug is critical enough to block entire business scenarios and prevent trainees from accomplishing any tasks, it might be necessary to fix these bugs directly in the training environment. However, we recommend this only in situations where the bug prevents any workarounds or other sessions that could be taught. Fixing the bug directly in a "live" environment could cause additional, unforeseen issues to occur, which would further lower users' confidence in the application. Furthermore, any bug fixes that happen in this instance would also have to be fixed in the lowest instance where development is occurring, and regression testing would have to be completed and any potential conflicts would have to be resolved.

Assumptions, dependencies, and risks

As discussed in the "Training schedule" section of this chapter, successful training relies on many external factors that should be defined in your training plan. These factors can be assumptions, which are statements assumed to be true of the project that are often critical to training success. Here are a couple examples of assumptions:

- Appropriate trainer resources will be allocated to create training material and train users.
- The tenant management and development team will create and maintain a training environment for use during the project.

Your training plan and project plan should also include dependencies, which can be represented in a process workflow. They effectively show that certain tasks or activities must occur before others as part of your training strategy. **Figure 19-8** depicts a sample process workflow.

We recommend making sure each training participant has a user account with the same level of access as their forthcoming production user account.

We do not recommend giving system administrator access to training users. This is a common mistake we see organizations make during training. They offer a variety of reasons for doing this, including the desire to not run into access issues during training and the security model having not been finalized.

If you are conducting multiple training sessions in a single training instance, it's important to take these proper actions at the conclusion of the training session:

- Any bugs that discovered during training are properly documented (with screenshots, if necessary) and sent for review or triage.
- If restoring from a Dynamics 365 backup, ensure that a backup for the correct date is available. (Work with your partner or IT staff, as well as Microsoft if necessary, to ensure this is not a blocker.)
- If restoring from source control as part of your ALM process, ensure all code and data are ready for an environment refresh.
- Refresh your environment using your chosen method.
- Validate user access and data via a smoke test, prior to the start of the next training.

This tree is an example of a straightforward dependency list for creating a specific type of training document. Your plan will contain more complex dependencies that might rely on multiple, unrelated project tasks. Defining and organizing these dependencies at the start of a project will help reduce the amount of time your trainers or other project members are waiting for required tasks to be completed, by scheduling upstream project tasks specifically to minimize this downtime. This must be done in coordination with the broader project team and project schedule.

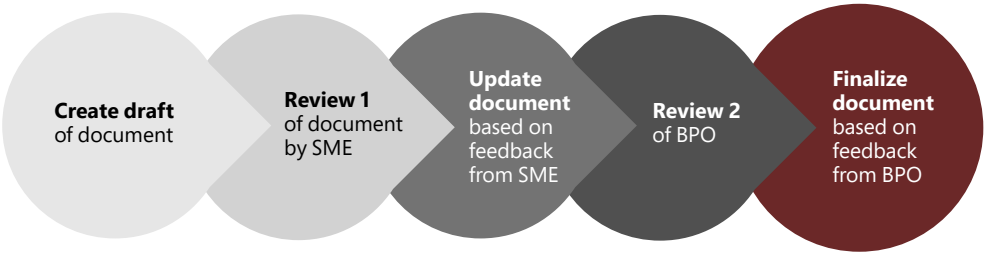
Lastly, creating and maintaining an active training-risk log, as part of the overall project risk log, is important to ensure proper attention is paid to training risks that could impact the success of training or other project activities reliant on successful user training.

Training as an ongoing process

As mentioned in the introduction of this chapter, your organization's training journey begins long before the execution of user training, and its ending extends long after those users complete their first tasks in your production application. Training is an ongoing and constantly evolving process that is reflective of changes in your organization and application, as well as changes to the Microsoft Dynamics 365 ecosystem. While developing a training strategy, it's important to consider the implications of both types of change.

Fig. 19-8

Create training documentation on lead/opportunity management



Internal change

Internal change, for the purposes of this section, refers to change that, at a transactional level, is unique to the organization experiencing it. Each of these types of change could require training material to be updated and subsequently conducted on the same or different groups of users.

It's important that new users are provided support during and after onboarding, as they might receive fewer training resources than users who were present during the application's initial rollout.

Personnel change

User turnover is inevitable—either when employees leave an organization and new ones take their place, or when employees change job roles within the company. All new users are expected to quickly learn their job roles and how to navigate and complete their work in the application; to do this, these users need to be trained. Note that it's unlikely that an organization would conduct multiple lengthy in-person training sessions for a single user's onboarding, and would instead rely more on self-paced or web-based training as a means of education. It's important that these new users are provided support during and after onboarding, as they not only are initially behind more experienced users who have been using the application for a longer period of time, but they might receive fewer training resources than users who were present during the application's initial rollout.

Your organization's training material for users who joined after go live will be similar to that used by your end users before go live, but again, be sure that these users are supported and given the assistance they require to train on the application.

Application change

Many Dynamics 365 projects have adopted a phased rollout approach, in which features of each application are released to production over time, as opposed to a "big bang" approach, in which a single deployment to production prior to go live has all included functionality. While the rest of this chapter covers the development and distribution of training regarding the first release of application functionality, future phases of the project must include updates to training material and new training sessions.

Your organization might want to treat each phase of a rollout as a distinct project from a training perspective, wherein each future phase contains a distinct, more condensed training plan and training delivery schedule to ensure proper user training is conducted at every stage. These future phases will focus heavily on making updates to training material based on new business processes, and training users—both new and experienced—on the application.

Consider a building-block approach when designing your training material. Phase one training material and training can be the foundation on which all users can learn to use the application. Future phases can build on that foundation; training material for subsequent phases can be updates of earlier material as well as newly created material.

External change

In other chapters, we discuss Microsoft Dynamics 365 as a software as a service (SaaS) that provides continual updates to organizations, both from a technical standpoint and a business standpoint. It's important that your organization prepares for these updates as part of its training strategy and that they are acknowledged in the training plan and execution. Note that we allow customers to activate updates to software at their own pace.

As updates to the Dynamics 365 application are announced and begin to be rolled out, your organization should set aside an instance used for regression testing product functionality against your custom application. As part of these activities, trainers should be involved and assess the potential impact these updates have on business process.

Certain updates, especially improvements made on the software's back end, will result in little to no change in the way your users do their work in your application. Thus, they should not require a significant update to training material, nor additional training. Other updates that are more focused on user experience or new features within the platform might drastically affect certain end users and will necessitate updates to training material as well as additional user training. **Figure 19-9** contains examples of how software updates might or might not require changes to your training offerings.

Training best practices

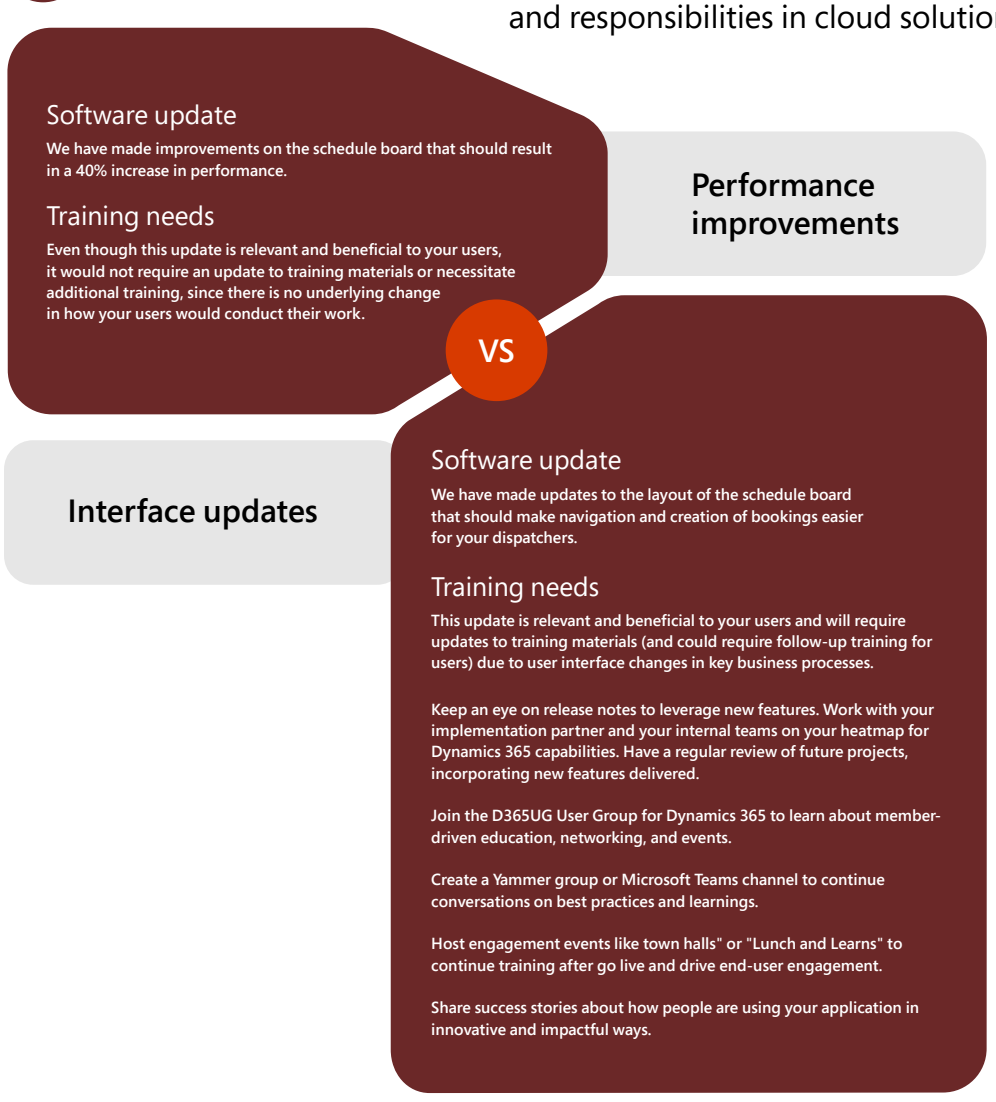
The authors of this book have experience with Microsoft Dynamics 365 implementations of all sizes and complexities. We have created and reviewed training plans, participated in and hosted training sessions, and evaluated the outcomes of training on end-user adoption. From most

of these projects, common themes have emerged. We share the most important recommendations here as best practices—things to keep in mind during your organization's training process. These best practices are by no means comprehensive, nor are they set in stone, but they are meant to guide a project team to a more successful training outcome.

Earlier is better

Starting early (with a training plan) is important. Dynamics 365 projects are often run on tight timelines with little to no leeway. It's crucial that a training plan, complete with staffing and resource availability, be completed at the start of a project, to ensure no unexpected resource requirements or shortages appear midway through implementation. Give yourself sufficient time to prepare a training plan, develop content, and deliver training. Content development should be completed well before the start of training delivery. Additionally, project roles and responsibilities in cloud solution deployments are rarely separate; often a single IT resource will wear multiple hats, and one of these hats might be to create a training plan, review training material, or maybe even conduct training. Making sure these individuals are aware at the outset of their contributions to the training plan will help them balance their workloads and complete certain activities earlier, which could prevent delays from occurring.

Fig. 19-9



More is (usually) better

In Dynamics 365 applications, a user can often access the same record or complete the same business process in several ways. Thus, training that initially seems

repetitive is useful in that it teaches multiple ways to accomplish a task depending on the situation and context. While too much training can mean a small amount of wasted effort, for an organization adopting a new application, the cost of time is far preferable to having users who are insufficiently trained on a sizable portion of their job. The latter situation occurs in organizations far more frequently than the former situation does, even for similar amounts of training.

It's also important that your training material does not contain too much duplicate content. While it's certainly critical that both the web-based and live content cover the same features (especially if users are not consuming multiple types of content), too much overlap can result in training fatigue, and overall apathy regarding training. An organization would certainly want to avoid a situation in which users skipped valuable content because they considered it repetitive.

Train with a goal in mind

It's important to not lose sight of the end goal: successful user education and meaningful adoption of your Dynamics 365 application.

Conducting a variety of training for all users using different methods is important. But what is much more important is not losing sight of the end goal: successful user education and *meaningful* adoption of your Dynamics 365 application. It does the organization no good if every technician understands how to use their new mobile application but none of them uses it correctly because of a failed training process. Having comprehensive training material is but one step in this process. As stated earlier in this chapter, conducting *effective* training that leads to excitement and high user adoption with accomplishment of critical KPIs, in addition to the more obvious goals of high training participation and education, is the key that can separate adequate training programs from excellent ones.

Consider your audience

In addition to having distinct job roles, your users come from diverse backgrounds and probably different generations, too. Avoid alienating groups of people with references, case scenarios, or even pictures or stories that could confuse or even offend certain groups of people. Your training should keep users engaged, but be careful not to be divisive.

For example, say you have a junior salesperson in their 20s and a senior salesperson in their 60s attending the same introductory training. Your application includes an embedded LinkedIn widget on your contact entity. Trainers should not assume that both users (or, more divisively, only one) are familiar with LinkedIn and its capabilities. A good rule of thumb is to assume all training participants arrive with the same level of background information and context of the application, plus any technology knowledge required to use it effectively.

Accessibility concerns

Your organization must consider accessibility while creating training material. All Office 365 products have an accessibility checker that can help you spot potential issues in your content.

Identify ongoing challenges

As discussed earlier, feedback is an important part of any improvement cycle. In addition to identifying areas of improvement for future trainings and training material, we recommend that the people conducting training or collecting feedback from the training identify key areas where users are struggling with new functionality. This information can be used to create more focused training material that can be distributed to users, or to create webinars or other sessions specifically focused on these problem areas.

Product-specific guidance

Up to this point in the chapter, our training guidance has applied to Dynamics 365 Operations as well as Dynamics 365 Customer Engagement application projects. While both applications live in the Microsoft Dynamics 365 ecosystem and customers frequently adopt both systems (often simultaneously), there are differences between the two, which can mean differences in how each project should train its users. In this section, we highlight some of these differences and how to apply them to your training material.

Dynamics 365 Operations

A number of resources provided in Dynamics 365 Finance, Supply Chain Management, and Commerce can assist with product help and training.

Help on docs.microsoft.com

The Dynamics 365 documentation on Microsoft's docs.microsoft.com site is the primary source for product documentation for the previously listed apps. This site offers the following features:

- **Access to the most up-to-date content** The site gives Microsoft a faster and more flexible way to create, deliver, and update product documentation. Therefore, you have easy access to the latest technical information.
- **Content written by experts** Content on the site is open to contributions by community members inside and outside of Microsoft.
- **Content is personalized based on language selected** If, for example, a user is set up for the German language, then any Help content they access will be provided in German.
- **Help is now on GitHub** Customers can copy and create personalized Help content and link it to their application. Customers can create a personalized and contextualized Help resource for users of their custom business processes.

You can find content on docs.microsoft.com by using any search engine. For the best results, use a site search, such as site:docs.microsoft.com Dynamics 365 "search term".

In-product Help

In the Dynamics 356 client, new users can enter the Help system to read articles that are pulled from the Docs site's Dynamics 365 area and task guides from the business process modeler (BPM) in Lifecycle Services (LCS). The help is contextualized to the form that the user is in. For example, if a user is in a sales orders screen and wants to know how to create sales orders, the Help system will show the Docs articles and task guides related to sales orders (see **Figure 19-10**).

Consider the number of concurrent users using the training environment as you choose the right tier for the environment. Do not automatically select the default tier 2 environment if the volume of expected users exceeds the recommended volume for the tier. For more information about different environment tiers and the volume of users each supports, read this documentation on selecting the right environment for your organization.

Task guides

Other useful help and training features are called task recorder and task guide. Task recorder allows you to record a user's activity on the UI. You can capture all the actions as well as any UI fields and controls that were used. This recording can then be used in a task guide.

A task guide is a controlled, guided, interactive experience that leads you through the steps of a task or business process. You can open (or play) a task guide from the Help pane. When you select a task guide, the Help pane shows the step-by-step instructions for the task (see **Figure 19-11**). Localized task guides are available, and you can create custom task guides. The data you enter in the system is saved in the environment.

To begin the guided, interactive experience, select Start task guide at the bottom of the Help pane. A black pointer shows you where to go first. Follow the instructions that appear in the UI and enter data as directed (**Figure 19-12**).

Dynamics 365 Customer Engagement

Dynamics 365 Customer Engagement has product-specific tools that can aid in training users. Gamification and custom help pages are two of the most common tools used; we describe them in more detail here.

Fig. 19-10

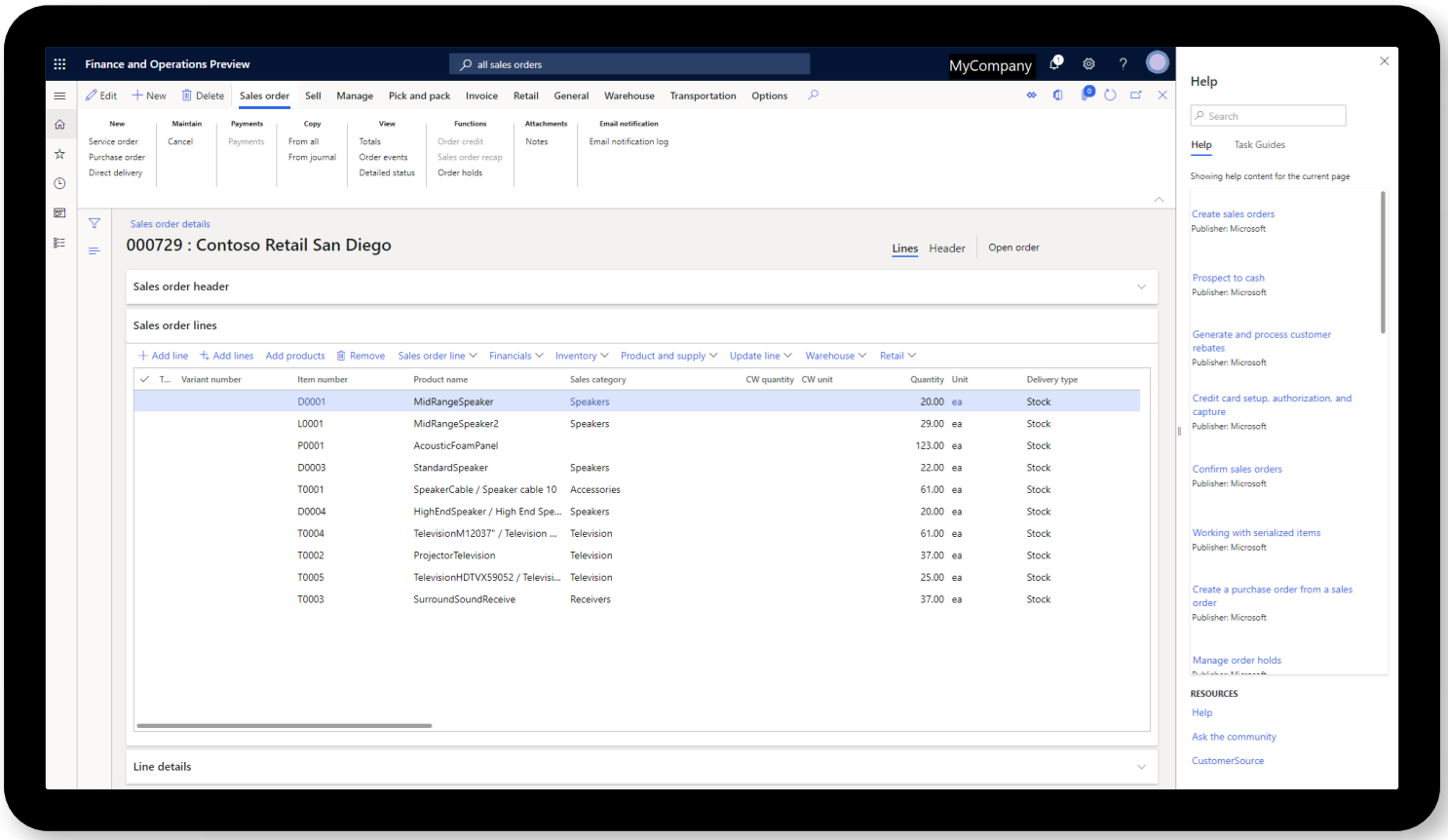


Fig. 19-11

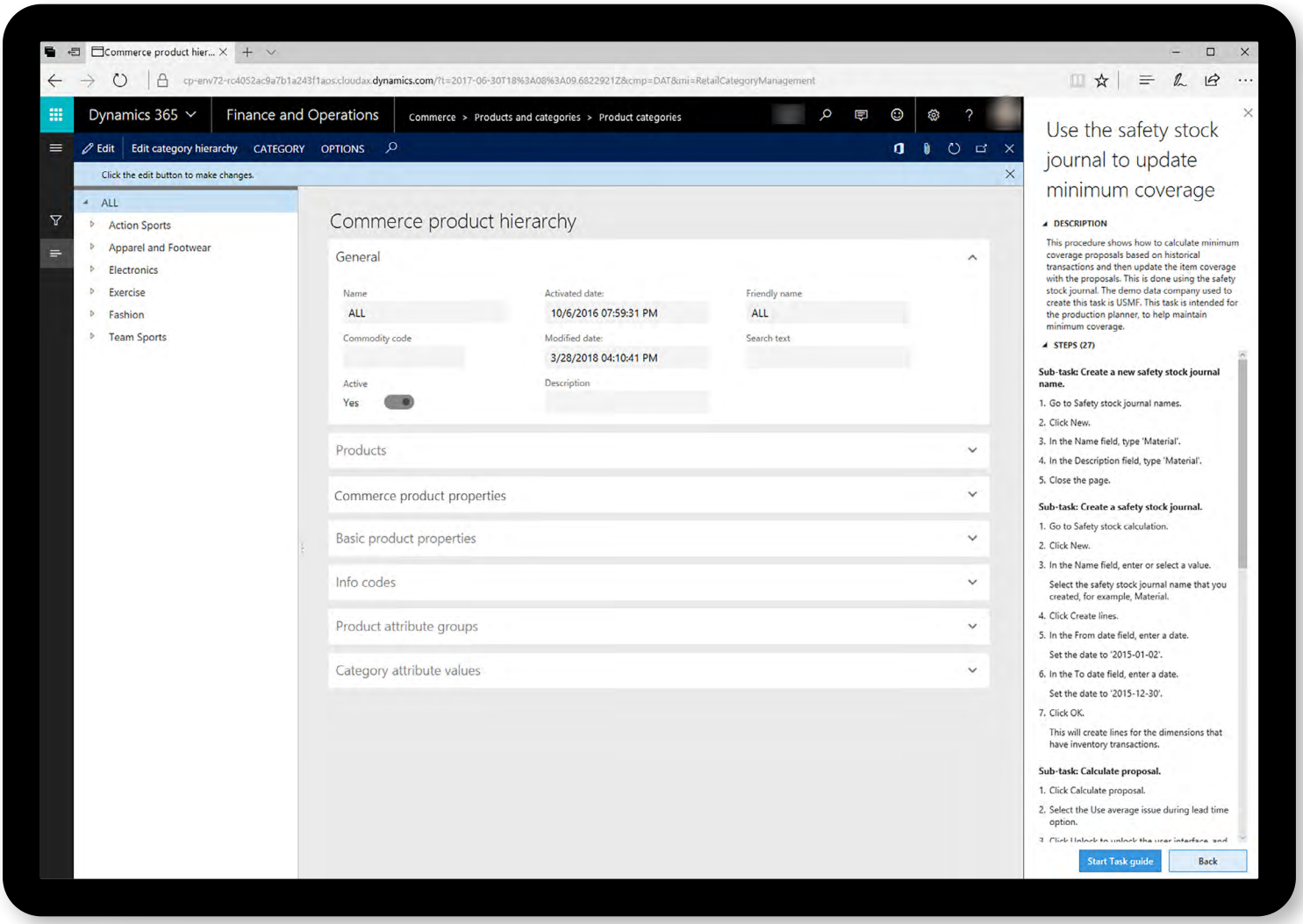
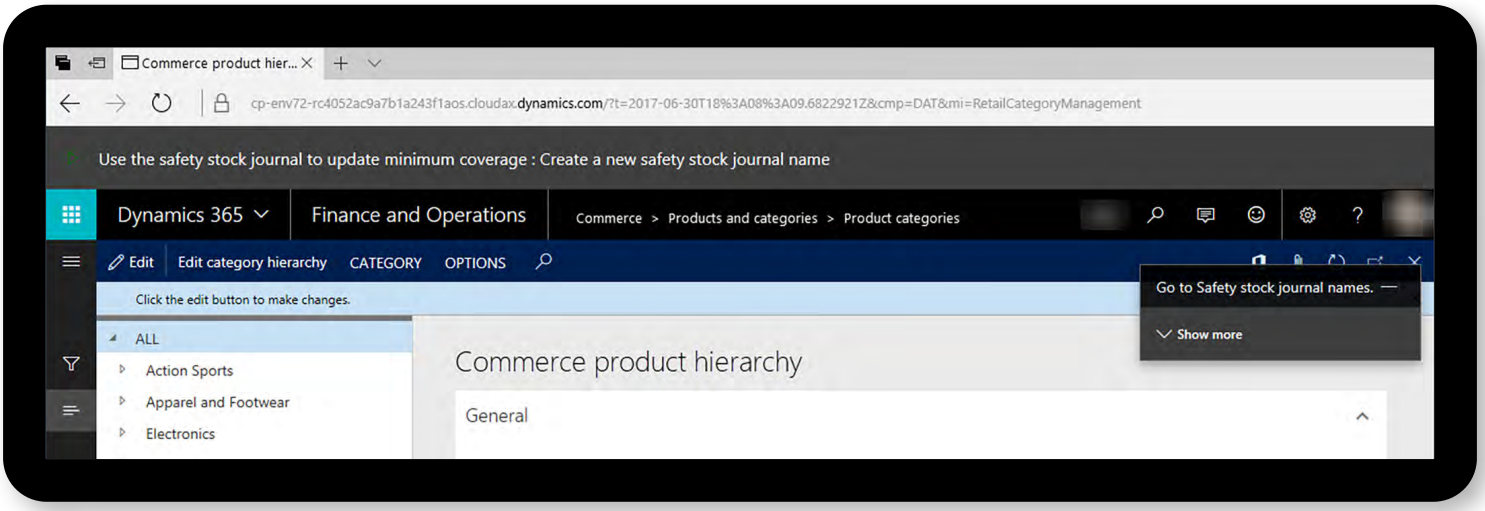


Fig. 19-12



Gamification

Gamification is a tool available from AppSource that is attached to Dynamics 365 Customer Engagement. Primarily used to increase the productivity of users within your Sales, Service, and Field Service/ Project Service applications, Gamification can also be utilized as an effective training tool when configured properly for users.

From a training perspective, an administrator can set up the Gamification solution in an instance within your Dynamics 365 tenant (usually Training, though games can also be created in Production as a means of "learning while you work"). You can define players, KPIs, and games to be hosted. Say your end-user training involves the participants following a complex business process from start to finish several times, and it concludes with a Case record being resolved. The administrator could create a game out of this. A KPI would be the successful completion of a Case by game participants (the trainees). To help motivate trainees, Gamification allows the configuration of prizes in Dynamics 365 Customer Engagement.

While Gamification and using games as part of training can be an especially useful motivational tool for your employees, you will want to ensure that competition is only so fierce as to foster a learning environment for your users. You do not want to create an adversarial culture in which collaborative discussion is absent and the goal of training—education and adoption—is lost.

Numerous custom ISVs also offer similar gamification solutions that can be attached to Dynamics 365 Customer Engagement. Pick the one that addresses the needs and goals of your organization best.

Custom help pages

Dynamics 365 Customer Engagement lets you create guided help pages that provide your users with in-product assistance. This assistance can be customized to your application and user base and can include text, links, images, and video links. For more information about custom help pages, read "Custom help panes and learning path."

References

Training plan template

Training plan charter template

TechTalk Series: Training Plans and Content for your Finance & Operations Project (Microsoft Dynamics blog)

Dynamics 365 training offerings (Microsoft Docs)

In-product Help – Finance & Operations (Microsoft Docs)

Checklist

Implementation

- [ ] Have training objectives that are SMART (specific, measurable, achievable, realistic, and timely), relevant to your organization's goal of user adoption, and fit the organization's overall change management goals.
- [ ] Have a process to update the training plan incrementally as necessary to reflect scope, changes, risks, and dependencies to ensure adoption and engagement.
- [ ] Consider what to include regarding system process and business process, so that the training provides the best possible foundation to end users.
- [ ] Clearly define the match between personas and roles, and business processes and user level, for each training session, so users receive relevant training.
- [ ] Ensure the training schedule accounts for key project milestones, resource requirements and availability, dates, and tasks, and is in alignment with the overall project plan.
- [ ] Consider different types of content, the time to create it, the effort to deliver and maintain it, storage, and the skillset of the authors and trainers.
- [ ] Set up the training environment with realistic data, user profiles, and scenarios. Refresh and clean it after training sessions to maintain quality.
- [ ] Ensure trainers and super users are trained well in advance of end-user training, which should ideally occur prior to UAT.
- [ ] Ensure the training plan appropriately addresses accessibility.
- [ ] Define a process for feedback and continuous improvements to training materials.
- [ ] Identify a process to provide for continuous training in alignment with updates and changes to the solution as well as changes in roles and responsibilities.

Case study

A proven path to training success

This global fire-protection company has earned a reputation for providing customers with maintenance and service of the highest quality. In this mid-size company's 75-year existence, they built their business organically through natural growth, marketing, and geographic expansion, as well as through numerous acquisitions of smaller fire-protection companies. They recently hired a new chief information officer to replace the CIO who retired after 15 years in the job. The new CIO had extensive experience working with Microsoft and SaaS products. The CIO's immediate focus was to consolidate and modernize the applications that employees were using to conduct their business.

In the training plan, the company included a mix of high-level and detailed training objectives. The high-level objectives included these goals:

- To not allow access to the live system without a solid training program
- To prepare a core team of trainers to help support the initiative
- To continue receiving feedback and improving the training approach
- To develop the training materials early and schedule classes early
- To prepare all application users to efficiently use the application (Dynamics 365 Finance and Dynamics 365 Supply Chain Management or Dynamics 365 Customer Engagement), as well as address any key business process changes required in their job function

Specific objectives were centered on business processes like Prospect to cash—for example, "All sales team members should be able to execute the standard Prospect to cash flow on their own." Their specific training objectives, listed here, helped support business goals for the application:

- Increase sales by 10 percent year over year
- Improve productivity by 25 percent by measuring Work Order completion time
- Increase user satisfaction (covered in annual poll) by 10 percent and user competency (measured by metrics within the application) by 15 percent within the accounting business department

The team understood that for training to be successful and for meaningful user adoption to be achieved, they needed to begin planning early and set up a strong team to support it.

The team benefited from using a change-impact assessment matrix to understand the changed business areas, topics, and user groups impacted. The matrix helped them prepare and prioritize training sessions.

The company had experience developing and conducting trainings for previous projects. They knew they wanted to incorporate a variety of learning modalities—written documentation, videos, hands-on learning, and multiple training labs using real and recognizable data.

Their legacy application infrastructure was a reflection of the way business had grown. The back-office system ran on an on-premises application that had been in use since 1990. Front-office users (salespeople and field dispatchers), as well as field technicians, did not employ any application; they were still using pen-and-paper-based methods. Due to the number of acquisitions they had made over the years, these methods varied—sometimes by business unit, sometimes by other organizational structure—and there was no overarching governance model consolidating these different methods.

As a result, employees that were in identical roles used different methods to accomplish their day-to-day work. This discrepancy in job

execution versus job role could be attributed to the fact that many of them joined the organization via acquisition and had never been required to change. It was necessary to capture all "as is" processes and assess the impacts of moving to the new application, from a technical perspective and from a business process change perspective.

Given the numerous challenges of supporting a wide variety of applications for all users, the company made use of the BPM, task guide, and custom guided help features for content development and user training. Having all business processes documented on one platform gave the organization a single source of truth. Task guides not only provided consistency to all the business processes in use across the different departments, but they also made the employee orientation and training a much smoother experience. Task guides were also embedded in the product help experience to provide a 24/7 interactive help experience.

By clearly defining all these processes, as well as the group of users to be trained, they were set up for success. Its subsequent training delivery was a smooth process that included all user persona business processes.

Because of the evolving nature of Dynamics 365 applications and organizations, as well as the fact that the project was being rolled out in multiple phases, the company developed an ongoing training process by which training was executed, reviewed, and updated as a cycle.

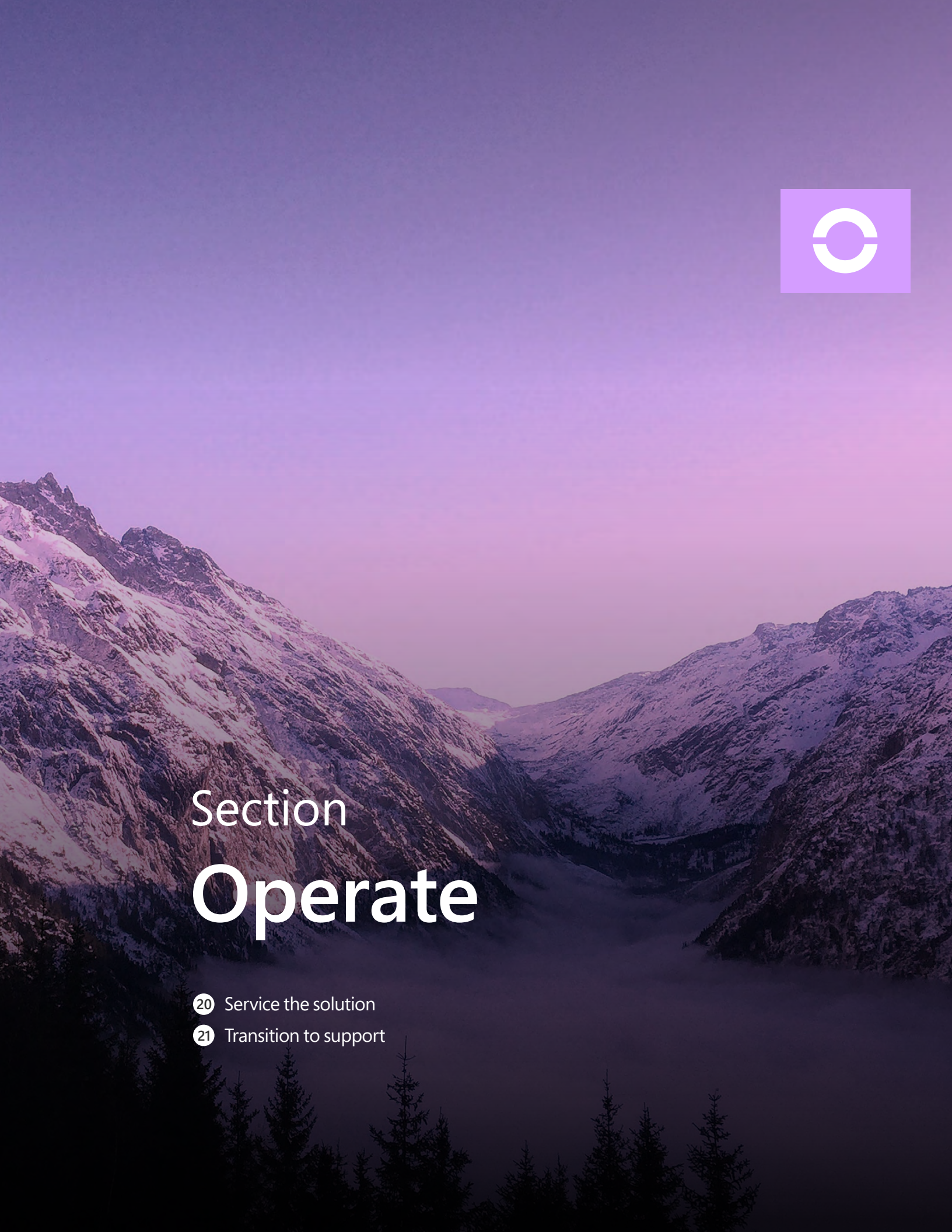
Formal feedback was recorded after the trainings and Microsoft Teams channels were created for employees to continue providing feedback to the team. Users were encouraged to share knowledge, ask questions, and suggest improvements to the training materials. The team was also able to collect feedback and create metrics using help desk tickets, which helped them identify areas of the application that users found particularly challenging.

The organization determined that mobile Field Service technicians were logging more tickets than other application users. By learning the types of tickets being created by their technicians, trainers were able to pinpoint a specific area of the Work Order process that needed to be

made clearer during training. Adjustments were made to the training material, too.

The objective “To not allow access to the live system without a solid training program” was a difficult objective to meet in the initial days, but the company learned over time that making sure every user received adequate training drove a significant reduction in business process issues for the company.

In the first few months, an evaluation of key KPIs showed that the organization was on track to meet all the detailed objectives set by the team.

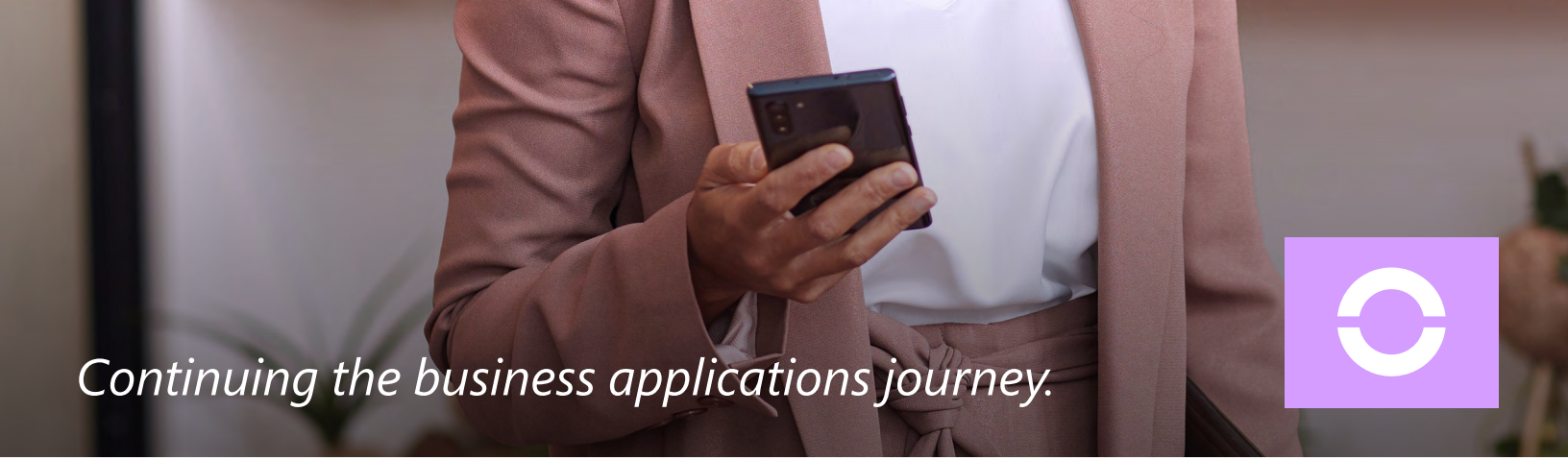


Section
Operate

20

Guide

Service the solution



Introduction

Your solution has been deployed to production. Users have been trained and onboarded. A support process is in place.

Most implementation projects focus on building and deploying the solution, but perhaps not nearly as much emphasis is put on educating the owner of the new system in keeping it optimal and healthy.

But you might be thinking, "Dynamics 365 is a SaaS platform. Isn't Microsoft responsible for maintaining these environments?"

Let's take a scenario where you rent an apartment in a large building complex. The owner of the building is responsible for cleanliness and upkeep of the building and common areas. They make sure to have proper plumbing and electricity in each apartment. They may add or improve amenities such as an exercise gym or a garden in the yard. But it is each renter's responsibility to keep their apartment in good condition. The renter should clean the apartment and throw out their garbage. They will need to replace the lightbulbs in the fixtures if it

For successful project governance, consider these main areas:

- Monitor service health
- Service updates
- Environment maintenance
- Continue the business application journey

goes out. If there are any issues with the plumbing, they need to contact the building manager so that a repair person can be dispatched to help. In a similar way, Microsoft, the owner of the software as a service (Saas) system, will be responsible for the platform—the building and property in our scenario. The customer is responsible for the upkeep of the health of their solution—the apartment.

In this chapter, we first look at how to proactively monitor the health of the solution. Then we cover the continuous updates provided by Dynamics 365 and best practices for working this pattern into your project plan to minimize impact to the changes in your environment. We also address environment maintenance tasks to maintain a healthy system. Finally, we challenge you to continue your education in the ever-evolving Microsoft business application journey.

Monitor service health

A key principle for a successful onboarding experience to Dynamics 365 is knowing the health of your environments at all times. Your team must be able to troubleshoot issues right away.

Dynamics 365 and the Power Platform have gone through rigorous testing, so why do customers need to check that it's running smoothly? It's because of each organization's unique usage patterns and the extensibility of the platform.

Just about every customer makes changes to their Dynamics 365 environment. It may be as small as adding new fields and forms. Or your business may require heavy customizations with complex logic or performance-impacting integrations with other systems. Or your organization may have more concurrent users than the typical use case, so data management needs are also growing at a rapid pace.

Another reason to monitor your system is to understand usage patterns for better business intelligence. Through this analysis, business stakeholders can pinpoint where to invest and prioritize improvements and focus areas for training.

Microsoft highly recommends monitoring service health and provides you the tools to do so. That way, you know when irregularities are found and action is needed.

As administrators and power users of Dynamics 365, you can focus on several key areas when monitoring usage. These areas impact performance, security, cost, and supportability, all central to keeping the system performance at optimal levels.

This section addresses key areas for monitoring the solution:

- Performance monitoring
- Licensing and operational cost
- Storage allocation and usage
- API request and service protection
- Security and privacy
- User access and resource usage
- Application logs and errors
- Messages from Microsoft

Performance monitoring

Microsoft recommends that Dynamics 365 project teams consider adding performance testing to the project's test cycle. Performance testing is an effective way to gauge the impact that your customizations may have on the baseline Dynamics 365 solution.

Performance testing is typically conducted in a simulated environment. This provides insight into changes in solution design or configuration prior to production rollout at scale. By conducting performance testing early, Dynamics 365 administrators have a baseline performance expectation of the new system that can be used as comparison over time.

Many factors can influence how performance is affected in live production scenarios:

- Integrations with other production systems have fluctuating response times that are different than in the simulated tests
- Workflows such as back-office jobs could be running, which stresses the systems

Chapter 17, "A performing solution, beyond infrastructure," covers the importance of having a performance strategy that includes elements such as defining performance requirements, establishing baseline metrics, and ongoing performance testing.

- The network traffic can vary throughout the day depending on an organization's usage patterns
- For remote workers, reliance on individual internet connections could cause different outcomes for each user

Ultimately, the responsiveness experienced by end users is caused by a mix of multiple factors that aren't limited to the performance of the Dynamics 365 application itself.

To understand the performance impact in these scenarios, proper end-to-end logging and collection of telemetry is required. This provides a complete picture of how long each step of the pipeline takes to complete. This information can reduce the time required to triage and fix any issues.

You can use many different tools to monitor performance. For Dynamics 365 Finance and Dynamics 365 Supply Chain Management, Lifecycle Services (LCS) is the first place to go for performance monitoring.

For Dynamics 365 Customer Engagement, Dataverse analytics is helpful. With it, you can gauge and monitor performance from within the Power Platform Admin Tool.

You can also use Azure Application Insights to monitor applications like Dynamics 365 for custom telemetry needs.

As this data is collected, monitor the performance of the solution and set up notifications to warn you when performance of any aspect of the solution varies from a defined range.

Poor performing solution leads to low user adoption. But you can stay ahead of this through proper monitoring and alerts.

Licensing and operational cost

As your solution matures and new users are onboarded to your Dynamics 365 solution, keeping track of the rate of licenses being consumed will be very important. Insight on the number of available licenses will help you plan ahead and adjust your purchasing as needed.

Stay compliant with user licensing requirements and view the independent software vendor (ISV) license status for Finance and Supply Chain Management for more information.

Review the business subscriptions and billing documentation for Microsoft 365 and the Microsoft 365 admin center to manage your subscription for Customer Engagement and Power Platform.

The Dynamics 365 Licensing Guide provides details on licensing requirements.

Another aspect is the operational cost. With solutions that include cloud computing, usage volume heavily impacts this expense. Staying aware of any changes in your organization's usage will help business and IT managers make key budget decisions.

Storage allocation and usage

With a Dynamics 365 subscription, you start with a set minimum amount of storage space for your environment.

Depending on your needs, you can add more data to suit your business size and expected growth.

Administrators must be mindful of the amount of storage that is available for the organization's subscription as usage grows. Different data classifications (such as transactions, files, and logs) have different allocations depending on your subscription.

Refer to Chapter 10, "Data management," for details on storage entitlements, segmentation, and impact to allocations when backing up and restoring instances.

For information on storage capacity for Finance and Supply Chain Management, see the Dynamics 365 Licensing Guide.

Note that Dataverse storage capacity entitlements and usage changed in 2019.

Users and integrations aren't the only cause of storage growth. Logs from system jobs, indexes created to help performance, and additional application data added from new modules also contribute to storage growth. Another scenario that impacts storage allocation is in the copy and restore operations of an environment. Depending on the type of copy, the size of the database can be very different. As an administrator, you need to be mindful of who can create new instances and what their true needs are to minimize impact on the storage allocation as these copies are being restored.

Administrators should monitor the volume of storage that is currently used as well as its growth rate. This information will help you budget for any additional storage needs or look into data archiving and deletion to free up space. Scheduling and cleaning up data from time to time will help as well. This is covered in the "Environment maintenance" section of this chapter.

API request and service protection

For Finance and Supply Chain Management, use LCS to monitor API usage. For more information, review priority-based throttling and how to configure priorities for special circumstances such as integrations.

For Customer Engagement, use the Dataverse analytics available in the Power Platform Admin center. Refer to the service protection API limits and the ways to maximize throughput to avoid hitting these limits.

One of the advantages of choosing a cloud solution over on-premises software is that your application is scalable. Among other things, you can adjust the amount of processing and memory resources based on the usage. But to provide consistent availability and performance for everyone, Microsoft monitors and applies some limits to how APIs are used. These limits are designed to detect when client applications are making extraordinary demands on server resources.

The intent is to prevent over-utilization of resources to preserve the system's responsiveness and performance for environments running Dynamics 365. When an API limit is hit, the service may be throttled

or even prevent API calls to be run. Error logging shows you when these limits are exceeded.

Administrators can also pull historical telemetry reports to see if any areas of the application are at risk of hitting these limits. Then you can work with the implementation team to make appropriate design changes. Or better yet, tools like Azure Application Insights allow you to set thresholds on these API calls so that when they're exceeded, the administrator is notified and can mitigate the risk of throttling or being shut down.

Security and privacy

From a security perspective, you want to prevent unwanted access to your system, applications, and data. The unified Microsoft 365 security center combines protection, detection, investigation, and response to threats for your Dynamics 365 system, all in a central portal.

At the solution level, you need strategic planning to create a robust security policy that can be implemented during deployment. As the administrator who services this solution, you must also monitor and act upon other aspects of security.

Take a scenario in which an employee changes roles or leaves the company. You need a process to change the security role and permissions for that user. When a vendor is granted access to a system as a consultant, they often continue to have access to it even when they're no longer working with the company. Microsoft recommends that you have a process to address these scenarios.

"Who has access to my customer data?" may seem like a simple question, but as the number of users and system usage grows, it can be daunting to keep track. You need to be able to identify who is accessing the system and what they're doing with the data.

Depending on the industry or the region of the organization, you may also have regulatory or compliance-related requirements. You might need to provide reports about customer data handling or delete customer data upon request.

Refer to Chapter 12, "Security," for details on security strategy and planning, and see the security center overview and the compliance center overview for more details. The Microsoft 365 security center provides the ability to search through Dataverse activity logging for Customer Engagement.

Your organization may be subject to rules such as the General Data Protection Regulation (GDPR) that give users specific rights to their personal data. You may need to respond to data subject requests (DSRs) to delete a user's personal data.

Microsoft recommends that you have a proper auditing strategy to capture the information needed to track user actions so that you can satisfy these types of requests. Most common areas are covered by Dynamics 365. Because auditing takes up more storage and potentially impacts performance, administrators need to turn some of these capabilities on where they may not be by default.

User access and resource usage

For any application or solution, it's important to understand your organization's resource usage patterns. Business sponsors want to know who is using (and not using) the system and the frequency of business processes and use cases that are being run.

Monitoring and Diagnostic tools in LCS provide telemetry for usage patterns for Finance and Supply Chain Management.

Microsoft Dataverse analytics provides user access and usage pattern telemetry for Customer Engagement and the Power Platform.

You can clearly understand user adoption through telemetry on usage patterns, and you can take proven actions to improve it. For example, if a certain capability isn't being used much, you can query the users to get their feedback on it. By using telemetry to identify areas with the greatest impact to the user group, the project team can work with business sponsors to prioritize features.

You can also use this information when estimating license needs. For example, some licenses may be assigned to users who rarely access the system; you can reassign these to someone new. Insights provided by the telemetry are readily available in Dynamics 365. Use this data to help improve business processes and allocate resources in the right areas to maximize your investment.

The platform is constantly adding functionality to the standard usage reports. If you have any gaps between your needs and these options, you can tailor tools such as Azure Application Insights to your usage pattern tracking requirements.

Application logs and errors

Typically, system errors are found by end users who report issues through an IT service management (ITSM) process. But you may have unreported issues that can only be found in the application logs.

Turning to notifications and application logs to proactively look for entries is a good way to find trouble spots and address them before they impact users.

Monitoring and Diagnostic tools in LCS allow administrators to monitor and query logs for issues detected in the system for Finance and Supply Chain Management.

Trace logging in Dataverse provides plugin error information for Customer Engagement and the Power Platform.

You can also use Microsoft 365 service health to identify service issues, and administrators can be notified via email or through the mobile app.

You can use tools such as Azure Application Insights for Dynamics 365 and other applications and services that are part of the overall IT solution for your organization. Application Insights lets you collect telemetry both in and outside of Dynamics 365.

For example, if a process is initiated by a user in Dynamics 365 that calls on an integrated, but external system, Application Insights can still detect performance and exceptions at different stages of the execution pipeline. You can see these issues whether they occur at the user interface level, in Dynamics 365, or the external system. This empowers the administrators who monitor alerts to react quickly the moment the exceptions surface. They also have immediate access to information on the source of the issue.

Messages from Microsoft

As your SaaS provider, Microsoft provides ample information about the health of your service, upcoming release information, service exceptions and interruptions, and other important details. The Microsoft 365 Message center is the central location for technical and licensing communications with our customers. As an administrator of your solution, this will be a very important area to frequently check for new messages.

Typically, organizations have multiple people who manage Microsoft services. You may want to provide appropriate team members with access to these messages.

You can also configure your preferences on receiving emails. A mobile app is available to manage these communications.

As described in this section, having the information needed to understand the health of your system is key in knowing when to act. Multiple tools are available to monitor your solution, which can feel daunting. But with tools to alert administrators when services are running less than optimal, the task will be much more manageable.

Key alerts you can receive from Microsoft include the service updates to your Dynamics 365 solution. In the next section, we discuss when and how Dynamics 365 is updated and what you can do as an administrator to take advantage of the predictable nature of the service updates.

Service updates

To enable businesses everywhere to accelerate their digital transformation, Microsoft is continuously enhancing Dynamics 365 with new capabilities. We add product enhancements and performance improvements at a rapid pace, so it's important to make sure Dynamics 365 provides an optimized user and administrator experience. Our objective is to help you keep your computing environment current in a consistent, predictable, and seamless manner.

Service updates are continuous, touchless updates that provide new features and functionality. They eliminate the need to do expensive upgrades every few years. The service updates maintain backward-compatibility, and contain application and platform changes that are critical improvements to the service, including regulatory updates.

Microsoft's push to One Version

One of the benefits of Dynamics 365 is that every customer runs on the same version of the service. With market demand for increased agility to gain incremental benefits of these cloud offerings at reduced operating expense costs, Dynamics 365 took a bold step to refrain from multiple releases. Now every security patch, bug fix, performance enhancement, and functional improvement accrues to all implementations across the globe.

We call it One Version. This is also referred to as Run One, a solution that is evergreen in the modern SaaS world.

One Version promises to bring predictable release management to these areas:

Refer to "Modernizing the way we update Dynamics 365" by Mo Osborne, Microsoft COO Business Applications, for details on the benefits of continuous deployment.

- **Predictable updates with continuous deployment**
  - Regularly scheduled intervals for major releases of performance and reliability improvement updates throughout the year
  - Using safe deployment practices and monitoring updates closely for any issues
  - Knowing the cadence up front so you can work it into your project schedule
- **Early visibility of upcoming changes**
  - Access to release notes well before the release for readiness so you know what the new changes are and the dates to start planning early
  - Visibility into new features and updates for business sponsors that can help their team be trained in new features
- **Test new capabilities ahead of time**
  - Ability for customers to opt in for early access to the release
  - Fully supported, production-ready updates that can be tested in your sandbox environments
  - Turning on features when you're ready instead of having it forced upon your environment
- **Opportunities to advance your knowledge**
  - Faster access to new features
  - Ease in sharing ideas and collaborating with the greater community because everyone is on the same version

Fig. 20-1

With One Version, Dynamics 365 addresses the tough challenges faced by customers and partners and reduces common rollout concerns (**Figure 20-1**). The solution automatically enhances the predictability of product updates. For example, One Version reduces the effort and capacity needed to test the update. It makes it easier to manage business change and facilitate appropriate adoption.

Although One Version greatly reduces the impact of deployments, the solution owner and the implementation team are still responsible for making sure certain tasks are addressed. These include planning for the update, assigning ownership of the tasks, raising awareness of coming changes, and supporting adoption. We cover these items throughout this section.

Release readiness

System administrators and others who have signed up for service notifications are alerted about upcoming releases, minor updates, and bug fixes. Being properly prepared is the key to successfully managing your solution. This isn't just up to the Dynamics 365 administrator. You need tight coordination between systems administrators, the Dynamics 365 project team that works on the platform, and the business user groups and subject matter experts (SMEs) who are the end users and champions of the system.

These notifications provide the information you need to start your planning—dates of release availability, release notes, and the process to opt in for early access.

Release notes and feature deprecation

Great planning starts with education—learning when and what is being rolled out. Your notifications explain the upcoming changes with a link to online documentation for more details. This information can help you gauge the impact to your organization and determine when and how to take action.

Administrators are notified of upcoming releases through the message center and by email. Notes for major releases are made available months before availability and provide several key pieces of information. First, the notes list the features in the release, including a description of the feature and the general availability (GA) date. The notes also show if the feature will be available for public preview and early access. This information will help in your readiness tasks.

Let's break down the three types of release dates:

- **Public preview** These capabilities are pre-release features intended to collect early feedback and aren't intended for production use. For planning purposes, these public previews can give you a good idea of what's coming up in the short-term roadmap. Not every feature is available through a public preview.
- **Early access** You can apply these production-ready, fully supported updates and features to your environment prior to the GA date. See more details in the following section about the process of

Refer to the Message center for detailed information on notifications, email preferences, and recipients for service updates.

For Finance and Supply Chain Management, refer to the One Version service updates overview for details on release planning, release notes, deprecations, and release cadence.

For Customer Engagement and the Power Platform, refer to the release notes, deprecation announcements, and release cadence.

opting in early.

- **General availability** This is the date that these features are deployed to your environment if the administrator hasn't opted in for early access.

Each release wave includes features and functionalities that you can enable for different types of users:

- **Users, automatically** These features include changes to the user experience for users and are enabled automatically.
- **Administrators, makers, or analysts, automatically** These features are meant to be used by administrators, makers, or business analysts. They're enabled automatically.
- **Users by administrators, makers, or analysts** These features must be enabled or configured by the administrators, makers, or business analysts to be available for their users.

If you choose to opt in to early access updates, you get features that are typically mandatory changes automatically enabled for users. Each feature in the release notes indicates which category it falls under.

There may also be announcements about deprecated features—those with a timeline to stop being supported. These announcements are equally important because they allow you time to gauge the impact and work with your business sponsor and technical team to address any concerns.

Deprecation of features is an important part of a solution's growth. As business and technology needs evolve, the solution needs to change to keep up with new requirements for security, compliance, and overall modernization. As a result, some features are deprecated and are replaced by a better solution.

Deprecated features continue to work and are fully supported until they're officially removed. After removal, the feature or capability no longer works. The deprecation notes provide information on what features are being removed, when this will happen, why it's happening, and what actions you can take to address the impact. Just like when you're getting ready for new features, organizations

must plan and prepare well before the removal of features to avoid negative impact from the deprecation.

Opt in for early access

At this point, you have the release notification and reviewed the release notes to do an impact assessment. The next step is to work with the project team to test the release with your solution. Some features available through our public preview program may be of interest for your organization.

Microsoft recommends creating a sandbox instance from a backup of your test instances prior to update deployment. The environment should be running your latest solution with enough data for meaningful testing. If you already have automated tests, you can speed up this process by validating that your solution functions as expected. If you need to fix any areas, you can flag them and work them into your project plan.

If your solution has been developed using supported methods for customization, the release in most cases will have little to no effect on it. Scenarios regarding deprecation that weren't addressed previously may surface from testing and need to be fixed.

Another important area to cover for early access is to work with the business sponsors to help them understand the impact to the end users. As part of the release, some updates may impact user experience, such as user interface (UI) navigation changes. Even small differences can have a meaningful impact. Imagine users in a large call center scenario, in which every additional second on a call with a customer can impact their service goals. In such a case, business managers want to make sure that the user group receives proper communication and takes time to provide training if necessary.

After this due diligence, your organization can schedule and apply the new release to the production environment. You should time this task for when it's most convenient to the users and administrators. Your organization can determine the best time to apply the release—when there will be the least amount of disruption to end users, other technical dependencies, and impact

Refer to the Dynamics 365 for Finance and Operations Cloud Application Lifecycle for more information.

to other projects. Don't wait for the release to be automatically applied to your environment.

Microsoft has deliberately planned the release cadence so that it allows you enough time to prepare. Aligning your own internal deployment planning with this schedule allows you to benefit from the new features and bug fixes and make sure that any customizations specific to your environment continue to function as expected. Thorough planning is key so that there are no unexpected results from the updates and you have control of when these releases are applied to your environment.

Product-specific guidance: Service updates for Finance and Supply Chain Management

With continuous touchless service updates in Finance and Supply Chain Management, the system maintains backward-compatibility, so you don't need to merge your code.

Safe deployment practices

Three new innovations provide additional enablement for the new modern application lifecycle—safe deployment rings, automated regression testing, and the customer Release Validation Program (RVP). In this section, we explain these and how they apply to you.

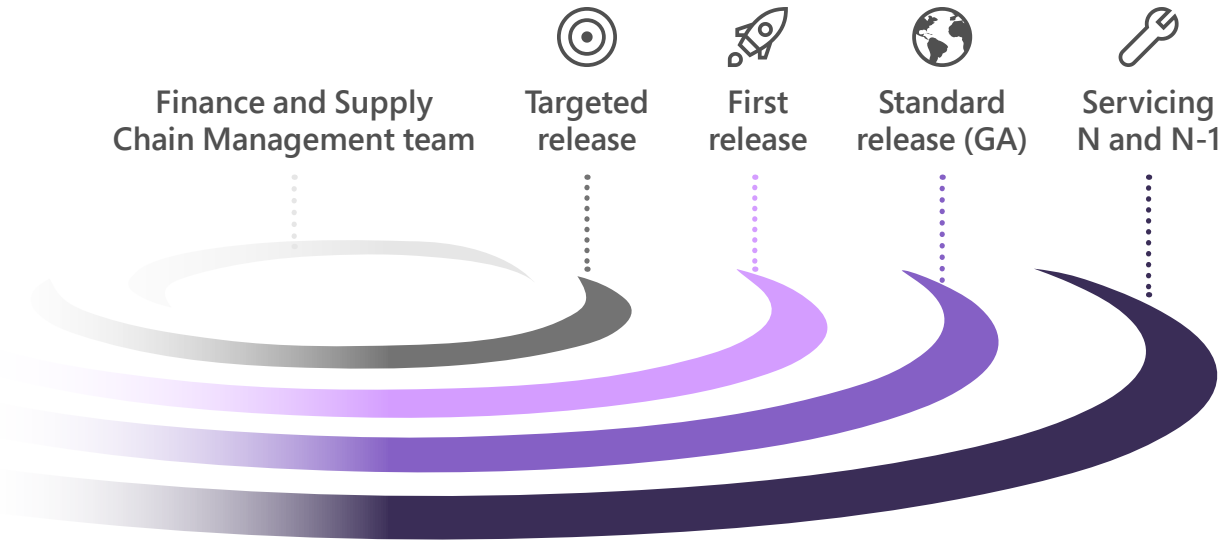
Deployment rings are a technique used to decrease any risk associated with rollouts for Azure and other Microsoft cloud services managed at global scale. As Dynamics 365 updates are created each month, they progress through a series of rings, with each ring providing broader exposure and usage and validation through system telemetry (**Figure 20-2**). The GA update benefits from extensive Microsoft testing as well as validation through each of the earlier rings.

Update cadence

Customers are required to take a minimum of two service updates per year, with a maximum of eight service updates per year (**Figure 20-3**).

Fig. 20-2

Safe deployment practice for Finance and Operations



Finance and Supply Chain Management team

- Extensive validation
- Compatibility checker
- Over 100 customer RVPs

Targeted release

- Preview early access program
- Preview build
- No production use

First release

- Select customers
- Auto update
- Production ready

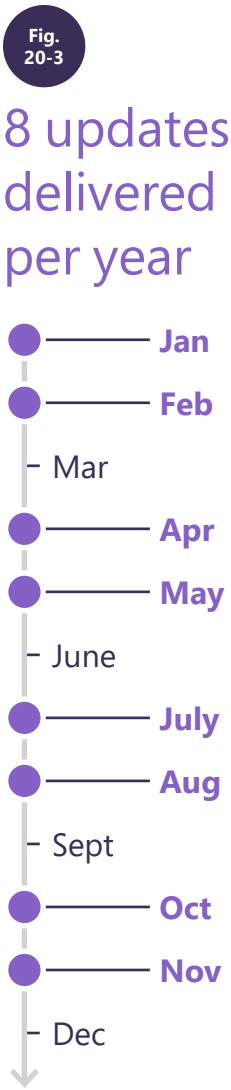
Standard release

- Predictability
- Customer-controlled patching window
- UAT followed by production on consecutive weeks

Servicing

- Hotfix support for critical issues
- Available for N and N-1 updates
- Cumulative updates

Fig. 20-3



You can choose to pause up to three consecutive updates at a time to accommodate your project schedule.

Pausing a service update can apply to the designated user acceptance testing (UAT) sandbox, the production environment, or both. If the pause window ends and the customer hasn't self-updated to a supported service update, Microsoft automatically applies the latest update based on the configuration selection available in LCS.

System updates follow these guidelines:

- Updates are backward-compatible
- Updates are cumulative
- Customers can configure the update window
- Quality updates containing hotfixes are only released for the current version (N) or previous version (N-1)
- System updates contain new features that you can selectively choose to enable

Release readiness

We strongly recommend that you plan ahead and work the updates into your internal project release cadence. To do so, take the following steps:

- **Step 1: Plan** Have a good understanding of the release schedule and a plan to work this into your application lifecycle management (ALM) strategy. Because you can pause updates up to three months, you can set aside plenty of time for testing, impact analysis, and developing a deployment plan. Use the impact analysis report from LCS to identify areas of change that may affect your solution and help determine the level of effort needed to remediate any impact.
- **Step 2: Test** We recommend using a non-production environment such as your UAT instance to opt in early and apply the release. You can configure service updates through LCS and specify how and when you receive service updates from Microsoft to your environments. As part of the configuration, define the update environment (production) and an alternate sandbox (UAT). Use the Regression Suite Automation Tool (RSAT) to perform regression testing to identify any issues. Work any fixes into your ALM cycle and deployment plans.
- **Step 3: Deploy** After you define environment and schedule for

the service updates through LCS, the back-end tools automatically update the system.

Product-specific guidance: Service updates for Customer Engagement

Dynamics 365 is continually updated to provide valuable functionality, fix any issues, and maintain the overall health of your solution. System administrators receive communications and are alerted to when changes will be made to their environments. Organizations need to plan to enable these service updates in alignment with their internal ALM process.

Safe deployment practices

When rolling out the major and minor releases, Microsoft Dynamics 365 follows a series of progressive, staged deployments across the world to adhere to safe deployment practices. At each stage, feedback is gathered and quality is monitored for continuous improvement.

For example, for Customer Engagement apps and Power Platform releases and updates, a station-based deployment plan is followed (**Figure 20-4**).

Check the latest release of station mapping and their corresponding regions. **Release updates to Station 1 through Station 6 follow the dark hours defined for each geography.**

After thorough internal integration tests and validations, Customer Engagement and Power Platform updates are rolled out to Station 1. This first release station consists of select customers and production-like environments. This is set up for early validation testing of the service update before it's rolled out to customer production environments. Based on the pass rate of the first release, the service update gets rolled out to the customer environments by Stations 2–6.

Some organizations require instances to be in different regions, which may place them into different stations than that of their primary geographies. Therefore, as the deployment cycle of a new release commences, instances in different stations can be on different versions of the solution. As your project team is developing new features that are consumed into your ALM deployment process, set up a version check for Dynamics 365 and make sure that there is a match so that your project doesn't encounter incompatibility issues. When the version of the source environments

matches the destination, you can safely deploy your solutions.

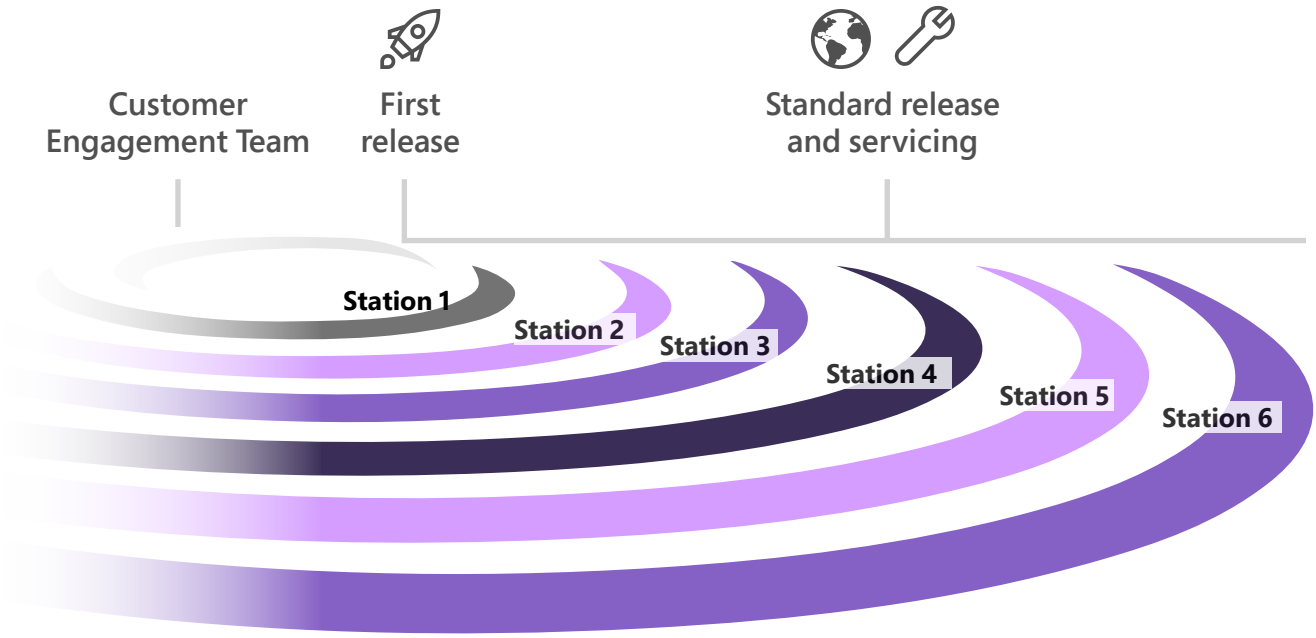
Types of service updates

There are four types of updates:

- **Major release** These happen twice per year in April and October. Major releases offer new capabilities and functionalities. These updates are backward-compatible. New features with

Fig. 20-4

Safe deployment practice for Customer Engagement



Customer Engagement Team

- Extensive integration testing and validation
- Solution checker

Station 1: First release

- Production quality
- Early view of weekly release
- Select customers

Standard release and servicing

- Predictable weekly releases
- Hotfix support for critical issues

Station 2 — JPN, SAM, CAN IND

Station 3 — APJ, GBR, OCE

Station 4 — EUR

Station 5 — NAM

Station 6 — SPECIAL

changes that might be disruptive to the user experience are turned off by default.

- **Minor service updates** Minor service updates are deployed on a weekly basis, region-by-region. They contain software customization changes that support new features, product improvements, and bug fixes.
- **Planned maintenance** Planned maintenance includes updates and changes to the service to provide increased stability, reliability, and performance.
- **Unplanned maintenance** Applications sometimes encounter unexpected issues that require changes to maintain availability. Microsoft strives to provide as much notification as possible around these events.

Update cadence

Customers receive two major updates per year, in the April and October GA releases (**Figure 20-5**). You can get early access and opt in months before the GA dates. These updates apply to both Power Platform and Dynamics 365 apps. We encourage you to opt in early to test and apply the release. The releases are production-ready and fully supported even when applying prior to the GA date. Activation for major updates is automatic through safe deployment processes for the region where the Dynamics 365 instance resides, on the deployment dates specified for the region.

Dynamics 365 apps have a different cadence from the major releases. For example, Dynamics 365 Marketing and Dynamics 365 Portals have monthly updates. Apps from ISVs from AppSource, Microsoft's app marketplace, may also have a different cadence. You should consult with the ISVs for any third-party apps you're using.

Release readiness

We strongly recommend that organizations work updates into their internal project release cadence. To do so, take the following steps:

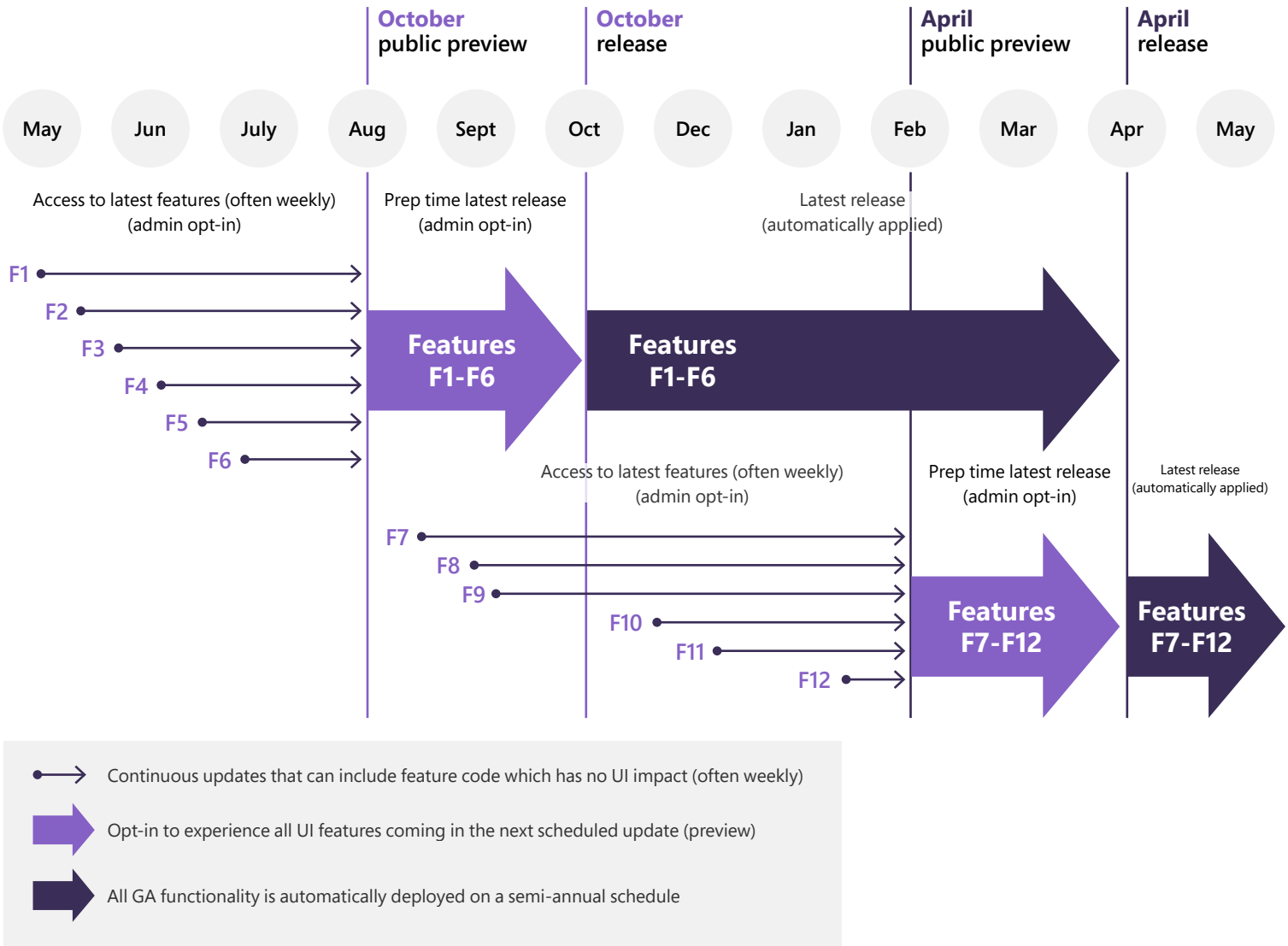
- **Step 1: Opt in for early access** Before you apply the changes to existing production or non-production environments (which may disrupt users and developers), we recommend you create a new instance. You can't revert back to the previous version, so all testing should be done on a new instance. Take a copy of the test environment that has your latest solution and data and create a new instance from it. Enable early access to apply the new release capabilities. After you opt in, some features are turned on by default; others may require an administrator to explicitly configure them. The details are documented in the

Refer to the Dataverse storage capacity guidance to understand the impact on your storage allocation for your tenant when creating new instances from a backup.

release notes.

- **Step 2: Test** Run regression testing to make sure that the solution continues to function as expected. Early opt-in features are production-ready and fully supported, so if you encounter any errors, you can submit a service request to report and get help with issues. Another important aspect of enabling the early access capabilities is that some UI and navigation changes may impact users. Work with your user group to do a side-by-side comparison between the current and opt-in versions. Use the release notes to identify the areas where UI or navigation changes have been made. Document the change with screenshots to be shared with the users. Depending on the significance of the changes, it may warrant some level of end user training as well as communications out to the group.
- **Step 3: Deploy** When the testing is complete, you can turn the

Fig. 20-5 Feature release



early access features on in other non-production instances. This ensures that new features and customizations by the project team are developed on the upcoming early release solution. You may keep one or more instances without the early access features turned on to support production bug fixes or anything that needs to be deployed prior to enabling the early access features in the production environment. When it's time to deploy your solutions to production, you enable the early access features on the production instance. You should notify business users of the new features (within the product as well as anything custom built by the project team) and any changes to how end users will navigate through the system. Timing the deployment is important. Microsoft doesn't recommend opting in at the same time you have a new project release in the production environment. If you encounter deployment issues, it's easier to troubleshoot when you're not deploying multiple solutions to the environment.

Environment maintenance

Protecting your solution and providing continuous availability of service is your primary goal as the system administrator. In a cloud environment, these maintenance jobs are automated, but it's critical for an organization to have a strategy so that these routine jobs are appropriately configured and scheduled. In some cases, you may need to perform these tasks manually but still in alignment with your overall planning and strategy.

You can manage users and licenses in both Finance and Operations apps and Customer Engagement.

You can also use Azure Active Directory (Azure AD) groups to simplify user management and provision and deprovision users.

User access and security

Members of business and technical teams fluctuate. New employees are onboarded and existing ones may change roles or leave the company. As an administrator, you need to work with other teams to manage user access, assign the appropriate licenses, and assign security roles that meet their business and technical needs. You may want to hand over some of these responsibilities to business and IT managers

who are more familiar with their resources' user profiles and know if any frequent changes in their roles require immediate changes.

Your organization's data is likely one of the most important assets you're responsible for safeguarding as an administrator. The ability to build apps and automation to use that data is a large part of your company's success. You can use Power Apps and Power Automate for rapid build and rollout of these high-value apps so that users can measure and act on the data in real time.

Apps and automation are becoming increasingly connected across multiple data sources and multiple services. Some of these might be external, third-party services, possibly even social networks. Users generally have good intentions, but they can easily overlook the potential for exposure from data leakage to services and audiences that shouldn't have access.

Microsoft recommends that you create data loss prevention (DLP) policies to act as guardrails to help prevent users from unintentionally exposing organizational data. You can scope DLP policies at the environment level or tenant level, which provides the flexibility to craft sensible policies with the right balance between protection and productivity. For tenant-level policies, you can define the scope to be all environments, selected environments, or all environments except those you specifically exclude. You can define environment-level policies one environment at a time.

Although the backup and recovery operation is dependable, it could also be time-consuming depending on the size of the backup.

In a Customer Engagement scenario when solution imports fail, it's often better to fix the import issue instead of restoring from a backup. Fixing the import should take significantly less time than restoring.

Dynamics 365 environment management

Dynamics 365 provides point-in-time restore (PITR) capabilities for databases. This means that all databases are backed up automatically by the system and retained for a set number of days. In the event of accidental corruption or deletion, administrators can choose to restore the database from any of the backups taken. The automated backup system and PITR provides a zero-admin way to protect databases.

If your organizations require a proactive approach to manually take backups (such as before a deployment of a new release of your solution),

the administrator may be called on to assist. You should perform these tasks in line with your organization's environment strategy.

Data management

Data is central to all applications. It drives business decisions through analytics and artificial intelligence. It also reveals crucial information about the overall health of the system and what administrators need to do for maintenance. The theme of this chapter is to be proactive in planning and strategizing around the upkeep of your system. Data maintenance is no different. In this section, we discuss the service aspects of data management. To explore the broader topic, refer to Chapter 10, "Data management."

Refer to Chapter 9, "Environment strategy," to explore the importance of having a strategy for creating and maintaining environments.

Learn more about database movement operations in Finance and Supply Chain Management and review environments for more information about managing instances for customer engagement.

Data volume grows quickly. Dynamics 365 has routine jobs in place to manage data that maximizes system performance at the platform level. You can also perform maintenance jobs like reindexing databases that adjust to variations in transactional data. But more improvements can always be done to better fit how you use the services. Developing a strategy to archive and delete data is an important step toward maintaining a well-functioning system.

How fast is the data growing?

You first need to understand how fast data is growing in your environment. The rate of growth is a key metric to monitor. Set alerts to let you know when this growth rate exceeds a threshold determined by your organization.

You can find details on storage allocation and purchasing additional storage for Finance and Supply Chain Management in the Dynamics 365 Licensing Guide.

Read about the storage capacity model for Dataverse and how to check storage growth.

The rate of growth can fluctuate depending on the number of users or even during certain times of the year if your business practice has special circumstances that may impact record creation. Monitoring storage growth and using historical trends will help estimate data growth. This information can help you determine how often the data archiving and removal process should take place.

What data can be removed?

Planning starts with identifying the types of data that need to be stored over particular timeframe. Besides cost, there are performance implications of having a large dataset. Building a data removal and retention strategy will help determine what to do when data is no longer useful.

Transactional data may help you make key business decisions, assist customers, and use AI to determine the next best action. But after years of inactivity, the data may be taking up storage space and not providing any value.

You may have logging requirements like auditing customer records to understand which users have access to read and modify personally identifiable information. But excessive data auditing leads to rapid storage growth and can negatively impact performance. Storing large files and attachments to emails is also an area in which datasets can quickly expand. Planning around what needs to be stored and for how long should be done deliberately and conservatively.

You may have logs, notifications, and other system records that you can delete with no impact to the business. You may also have other transactional data that can be deleted. Because Dynamics 365 applications have heavy parent-child relationships between records, pay careful attention to how records are deleted and any impact to related records. Look for triggers that run extension code or workflows when a record is modified or deleted. A delete operation could potentially write a new entry in the audit log to record the transaction. You must account for all these things when planning for bulk deletion.

Review the cleanup routines for Finance and Supply Chain Management to delete historical logs and notifications. You should only run these cleanup routines after the business has completed a detailed analysis and confirmed that the data is no longer required.

You can also free up storage space for Customer Engagement.

Archive and retention strategy

What happens if the database is growing too large, and you're seeing the impact of it, but deleting records isn't an option? You may need to retain the data due to organizational, compliance, or regulatory requirements. In this case, you can archive the data instead.

Your organization may be subject to rules such as the GDPR that give users specific rights to their personal data. To comply with privacy and security regulations, you may need to respond to DSRs to delete a user's personal data.

We recommend reviewing the guidelines for privacy and personal data for Microsoft Dynamics 365.

We recommend the following strategy:

- **Identify retention requirements** Get a full understanding of the requirements for the types of records that you need to retain and how long you need to do so.
- **Develop an archiving strategy** Once you have a clear understanding of what data to retain, plan for how to archive these records. Where will you store the archived data? How will you access this data when needed? What is the process to periodically move records from the Dynamics 365 systems into the archives?

- **Remove the data** For any data that doesn't meet the retention criteria, plan out your process to remove it from the archives.

Continue the business application journey

The importance of continuing your education on the business application platform is often overlooked. In the on-premises software world, you would go live with a version of your solution and it would mostly remain the same until you decided to develop or purchase add-on functionality. As mentioned earlier in the service update section, the benefit of continuous updates in Dynamics 365 is that you always have access to the newest features to help your business. This also means you have a constant stream of new and better ways to meet your business and technical objectives through your solution.

For example, let's look at integrating Dynamics 365 with external systems. It wasn't very long ago that you needed custom code modules in order to efficiently pass data to and from Dynamics 365. But with tools like Power Automate and Azure Logic Apps, you can build very powerful integrations through configuration with little to no code.

Reporting is another good example. The out-of-the-box experience for reporting had limitations in Dynamics 365—you could only build reports with data stored in Dynamics 365. Now, with tools like Power BI and its capabilities to build reports from data in and outside of Dynamics 365, you have much more flexibility to quickly build out and embed powerful reports. Also, some advanced Azure and Microsoft 365 services coexist with Dynamics 365, and are going through the same type of evolution. Everything from business functionality to ALM build and deploy tools are constantly seeing incremental improvements.

The only way to truly understand these benefits and how to best apply them in your solution for the most return on your IT investment

is by continuing your education. Gain an understanding of where the business application domain is headed. Strive to understand what is in the Dynamics 365 roadmap, why it's included, and how it can make your solution and your organization more efficient and robust.

Each year, Microsoft creates many events and learning opportunities to help developers, IT and business professionals, partners, and educators extend their expertise. Here are some great resources for furthering your education:

- Microsoft Events consolidate links to stay up to date on upcoming events for all kinds of audiences—both business and technical focused.
- Microsoft Business Applications Summit helps you maximize on using business applications through live and on-demand recordings and discussions.
- Microsoft Ignite is for anyone who is motivated to be on the frontier of innovation and tech. There are live and on-demand sessions tailored for different audiences.
- Experience Dynamics 365 provides access to Dynamics 365 experts and the Insider program to participate in preview features. You can also provide feedback to Microsoft engineers and vote on new product features.
- The Dynamics 365 YouTube channel delivers the latest product news, announcements, events, demos, and customer success stories.

Conclusion

In summary, take steps to be aware of your solution performance, be proactive in taking action, and be prepared with a solid strategy for maintaining your environments. Keep up on new trends and tools that can help improve your solution and your organization.

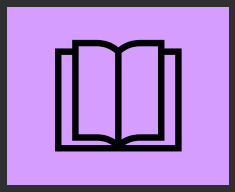
It all starts with visibility into the health of the system through proper monitoring. Having the right telemetry and managing notifications from the system as well as Microsoft will help you to prioritize and act to address maintenance needs.

A large part of the overall maintenance is simply understanding

the continuous update model of the Dynamics 365 service. We recommend that you be proactive in planning, communicating, and deploying service updates to gain the benefits of new features and enhancements made available throughout the year.

Appropriate maintenance of your environments and data will help protect and optimize your solution. This is a team effort, not just the responsibility of IT administrators. By working across the business and technology resources, you can make sure that the solution performs well technically and maximizes results for the user community.

Finally, just as your solution is continuously updated, organizations also need continuous education in trends and tools related to your business applications. With awareness of the product roadmap as well as improvements in anciliary products, you can maximize your organization's investment in the Dynamics 365 platform.



Case study

Fruit company learns the importance of servicing the solution

An agricultural business that grows fruit implemented Dynamics 365 Finance and Dynamics 365 Supply Chain Management as soon as the cloud version of Dynamics 365 became available. The company has been a global leader distributing fruit across different regions, and warehouse operations and transportation are part of the company's core business.

Before implementing the Finance and Supply Chain Management apps, the fruit company was using Dynamics AX 2012, and was familiar with the ALM of that on-premises enterprise resource planning (ERP) solution—but moving to the cloud required a mindset change.

Dynamics 365 has been evolving since its initial release, when the application and the platform were released as separate components. By the time continuous updates and a single, unified version became the norm for Dynamics 365, the fruit producer's operations in the cloud were mature in taking updates under this modality. The company was ready to adopt the modernized update pattern and take advantage of the continuous innovations from Microsoft. They also wanted to fulfill one of their expected returns

on investment (ROIs) by entrusting Microsoft to bring new functionality, instead of developing it on their own.

The fruit company's solution involved standard Dynamics 365 apps, some extensions, and an ISV. As the direct implementer of the entire solution, the ISV created a strategic partnership with the company and provided IT outsourcing services.

In the beginning, the incentive for the company to update Dynamics 365 with new releases was the continued support from Microsoft per the software update policy. The app functioned properly and applying the releases was manageable, so the company didn't have hard requirements to stay up to date with the release cadence.

But the company noticed that while the ISV kept the solution up to date with the Dynamics 365 releases, the ISV always provided an update of their solution using Microsoft's last update ring—or missed it entirely. Because of this, the company had to apply Microsoft's updates late in the release cycle or, in some cases, not at all.

To grow and distribute fruit effectively, the company relies heavily on time-sensitive warehouse operations where every hour matters as fruit is moved from the fields to climate-controlled warehouses to shipping containers that are sent worldwide as quickly as possible to avoid spoilage. They also were expanding into distribution of seasonal fruits in new markets, which made management of warehouse operations even more critical.

Then Microsoft notified the fruit company about new functionality for advanced warehouse operations in an upcoming release of the Supply Chain Management app. Because of the expansion of the company's operations and the complexity of their warehouse management, these new features were crucial to increasing productivity and saving time and resources while managing the expansion.

To adopt this new functionality, the company had to test the features in advance to align their processes. They realized that being late on updating standard features wasn't workable, and they wanted to

optimize their ALM and validate their solution in early update rings. To be ready for their peak season, completion of testing was time-sensitive, and the company would only have enough time to adopt new functionality and do proper regression testing if all solution components were in place and ready. So, they asked their ISV to provide software earlier than the general availability of their new version.

The challenge came when the ISV wasn't aligned to that timeline. The ISV's usual practice of adopting Dynamics 365 releases when they became generally available—in the last ring—meant the fruit company frequently was one version behind on Finance and Supply Chain Management releases.

The ISV wanted to honor their partnership and be flexible in meeting the timing requirements for updates, but due to dependencies on their internal roadmap and ALM strategy, the ISV couldn't make the software available to meet the company's timeline. As a result, the fruit company couldn't take advantage of the Supply Chain Management app's advanced warehouse operation functionality in time for their expansion.

Even though the company's ALM strategy could adapt to Dynamics 365's continuous update delivery model, reliance on the ISV's software prevented them from using the latest version of Dynamics 365.

After conversations with the fruit company, the ISV agreed to align with Microsoft's update rings and give the company—and their other customers—an opportunity to test their entire solution when the standard release was available.

Servicing the solution involves more than just Microsoft providing continuous updates. Microsoft customers must synchronize updates to their own extensions and any ISV solutions they're using, so they can benefit from the latest Microsoft innovations as soon as new features are available.

21

Guide

Transition to support

Poor service is always more expensive than good service.

Introduction

When introducing a new system into an organization, we need to think through the various areas that will influence how the project transitions from project mode to support mode.

In this section, we discuss how to construct a strategy to help you prepare, define, and operate a support model.

Fig. 21-1

Support scope

- Enterprise architecture
- Business and IT policies
- Dynamics 365-specific considerations
- Business processes
- Business continuity
- System update cycles

Support models

- Support model options
- Support organization

Support operations

- Transition
- Requirements management
- Change management
- Hypercare

Organizations that spend the necessary time and energy to construct a strategy that explicitly addresses how to create a fit-for-purpose support organization for their Dynamics 365 application tend to have better user satisfaction, better adoption of the system, and therefore higher-quality outcomes for the business.

If this is the first Dynamics 365 business application for your company, you may not have experience in setting up the support organization, support infrastructure, and support procedures for this application. Establishing a support organization for a new system (for example, Dynamics 365) needs more due diligence and may require a different perspective that better reflects the new technologies and new ways of working that come with it.

This chapter is organized to reflect the journey of preparing, defining, and operating support through the categories of support scope, support models, and support operations (**Figure 21-1**).

Support scope

A key aspect in preparing for production support is compiling the list of topics, tasks, impacts, and responsibilities that supporting the new Dynamics 365 application needs to cover. The support scope defines the "what" of the support model.

Support models

The scope definition influences the decisions that need to be made to identify the best support model and the constitution of the resulting support organization. This helps define the "who" of the support model.

Support operations

Finally, we discuss the distinct support requirements that emerge from the transition and hypercare project phases.

Support scope

- Enterprise architecture
- Business and IT policies
- Dynamics 365-specific considerations
- Business processes
- Business continuity
- System upgrade cycles

Support scope

Support scope activities should be started in the Initiate phase. It may be tempting to defer defining the scope until much later in the project because it's not seen as urgent compared to getting the project off the ground and creating the system solution. However, as this chapter discusses, early attention to this topic enables the support team to become engaged in the project from an early stage, which pays dividends during the project lifecycle.

Enterprise architecture

In most organizations, the Dynamics 365 system is embedded within the wider enterprise system landscape. The Dynamics 365 application has connections to multiple systems and to underlying and coexisting technologies. When considering the strategy for defining the support model, the focus is often solely on the Dynamics 365 application architecture. It's worth accounting for the changes, constraints, procedures, and influences from the surrounding architecture on the Dynamics 365

applications, and vice versa—how will the enterprise architecture and environment be impacted by installing and operating the Dynamics 365 application?

Firstly, from the enterprise system architecture documents, confirm that you have identified all the various systems, software, and data that interact with Dynamics 365 applications. You should make a distinction between new systems being introduced into the enterprise architecture and those that may be influenced by the new systems.

During the project, the new Dynamics 365 system will probably be implemented within a special sandbox (test environment) and not necessarily be subject to all the influences and rules that the production system is subject to. This also applies to the third-party test systems that are part of the middleware or integrations. For example, production environments have more limited access to the Dynamics SQL database, and the process by which custom code is promoted to production or how the Microsoft system updates are applied isn't the same. You should specifically examine the implications of the production system environment on support operations and not rely solely on the experiences of the test environment.

Identify the specific, new, or changed support requirements impacted by the following:

- Enterprise-level architecture configurations in areas such as Microsoft Azure, operating systems, any existing Microsoft 365 tenants, Azure storage, and Azure Active Directory
- Dynamics 365 Azure environments managed by Microsoft (for example, the Dynamics 365 production environment)
- Identity and access management systems, such as operating systems, browsers, mobile devices, and firewall configurations
- If this is the first significant cloud, software as a service (SaaS) system in the enterprise, the related system architecture, and servicing needs
- Systems to which the Dynamics 365 application interfaces directly
- Middleware systems to which the Dynamics 365 application may connect

- Any systems that will be decommissioned but have residual tasks that may need to be reabsorbed elsewhere by the support team
- Any document management systems such as Microsoft SharePoint or third-party systems
- Any code configuration management systems such as Azure DevOps or third-party systems
- Reporting systems to which the Dynamics 365 application interfaces, including Azure Data Lake or data warehouse

The impact of, and on, the surrounding architecture can be difficult for the customer project team working on the Dynamics 365 business application to fully identify. In almost all cases, you need the enterprise architects from IT and the business to be involved in identifying the changes. Some changes may also impact the roles of individuals formally part of a support organization and those in peripheral organizations.

Business and IT policies

All new systems need to think about how they will operate within the wider organization's policies and standards. When Dynamics 365 is introduced into the enterprise, it needs to follow the applicable policies already in place and may require new operating policies. In either case, you need to review the existing policies and standards to determine which policies to add, which to review, and which to apply to the support model (**Figure 21-2**).

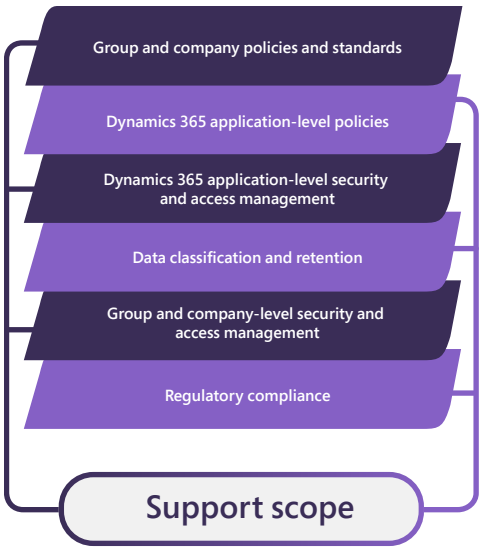
In many cases, the policies apply not only to the creation of the support model and its scope of responsibility, but also to how it operates as an organization and how it addresses the lifecycle of a support request.

Group and company policies and standards

When evaluating group-level policies or standards, you may have to review the impact of business policies and procedures on various levels:

- **Contracting with and managing third-party vendors** The

Fig. 21-2



support model needs to include some type of contract with technology partners and Microsoft

- **Operating hours** Current and future operating hours for the business can directly impact the working hours for support
- **Financial period end and seasonal deadlines** This may impact the level of service expected at times when the business has critical business activities
- **IT policies on change control on enterprise IT assets** This may impact the access and approval chain for changes required for the Dynamics 365 application troubleshooting or servicing (for example, creating new Azure virtual machines, copying databases between environments, and building IP allowlists)

The overall support model on Dynamics 365 business applications operates across many of the company's business processes and is therefore shaped by the relevant business policies.

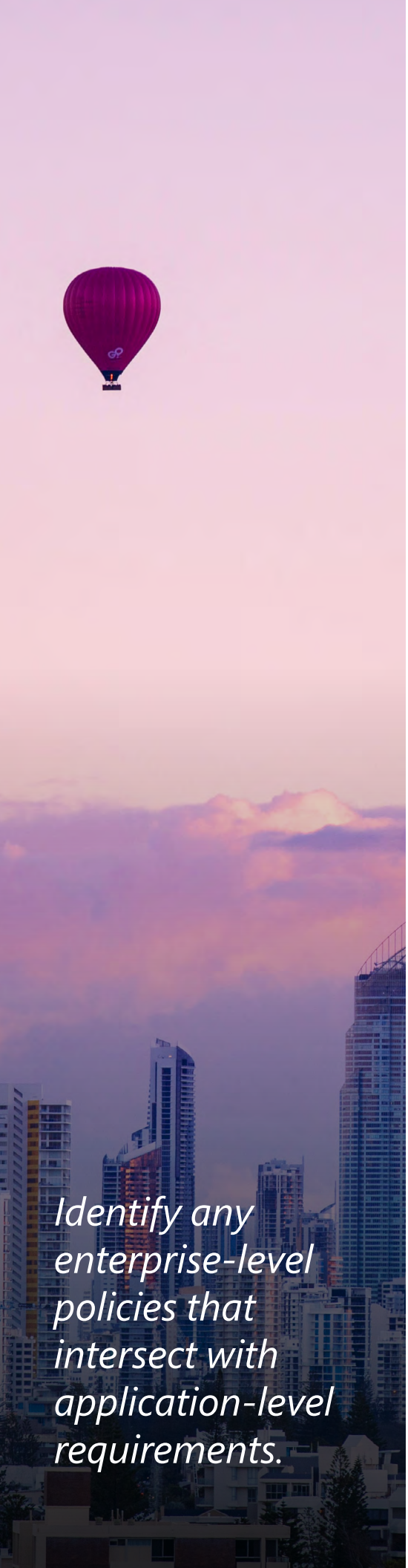
Dynamics 365 application-level policies

Some policies are managed in the system by the business administrators, and some may be delegated to the support team. Even if the administration of a policy is the responsibility of the business leads, the monitoring and auditing of the business process is often the support team's responsibility.

You could set up some of these policies within the Dynamics 365 application, such as new vendor approval policies, customer credit limits, purchase order approval thresholds, and travel and expense policies. The support team may need to help provide information on compliance or enforce compliance as part of their duties.

Dynamics 365 application-level security and access management

Administration and review of application-level security is usually an area of responsibility of the support team. The support team needs to prepare by gaining an understanding of the Dynamics 365 role-based application security model and the related access policies.



Group and company-level security and access management

In addition to the security and access policies and procedures that you need to define for Dynamics 365 applications, you may have to consider enterprise-level security policies that intersect with the application-level security requirements.

In larger organizations, enterprise-level security and access management may be handled by a separate IT organization. Some of the support team members will need elevated access not just at the application level, but possibly for other systems within the enterprise.

The Dynamics 365 support team needs to work with these other enterprise IT and business teams to define the rules and procedures for managing some of the security topics that may impact Dynamics 365 applications:

- Azure Active Directory groups and identities
- Single sign-on (SSO)
- Multifactor authentication
- Mobile device authentication and management
- Authentication and management for custom applications working on Microsoft Dataverse (such as Power Platform apps), which requires an understanding of Dataverse security concepts
- Application access for third parties (such as vendors and customers)
- Application access for third-party partner support organizations (such as technology partners and Microsoft)
- Service account and administrator account use and management
- Password renewals and security certificate rotations
- Secure and encrypted communications within the enterprise and outside the enterprise (such as those involved with integrations with internal systems, or with external web services or banking systems)

The Microsoft Trust Center can help your organization consider overall security and managing compliance in the cloud. Chapter 12, “Security,” provides a more detailed discussion of security for Dynamics 365 business applications.

Data classification and retention

Consider how the organization's data classification and retention policies reflect on and need to be expanded to include the new Dynamics 365 application:

- How is the support team expected to enable and enforce these policies?
- What is the impact on the backup, restore, and archive process?
- What is the impact on creating and managing database copies?
- Do any data classification properties flow between systems, or do they need to be recreated or audited by the support team?

Regulatory compliance and policies

In addition to any data retention policies, the organization's businesses may be subject to national and supranational regulations. Some may impact business processes that are expected to be assisted by the support team (such as e-invoicing and digital tax).

The support team may be expected to support, monitor, and audit other regulations on a regular or ad-hoc basis, such as General Data Protection Regulation (GDPR) or Health Insurance Portability and Accountability Act (HIPAA).

All of these different areas of business and IT policies shape the nature, size, and scope of the support organization. Early examination of these factors will help the team be effective from the start.

Dynamics 365-specific considerations

In this section, we examine some topics specific to Dynamics 365 business applications that we should consider when designing the support model. These can be broadly grouped by operational and maintenance topics and business process topics.

Operational and maintenance topics

We can further divide operational and maintenance areas into the

management and support of the following:

- Dynamics 365 environments
- Integrations
- System and data maintenance
- Performance

Dynamics 365 environments

When Dynamics 365 is in production, various aspects involved in supporting the system environments need to be factored into the support model preparation:

- Maintenance and operational tasks (both scheduled and ad hoc) such as environment data refresh, database movements between environments, configuration moves, and code promotion
- Support organization roles and resources needed
- Skills required to support the environments
- Resulting budget impact

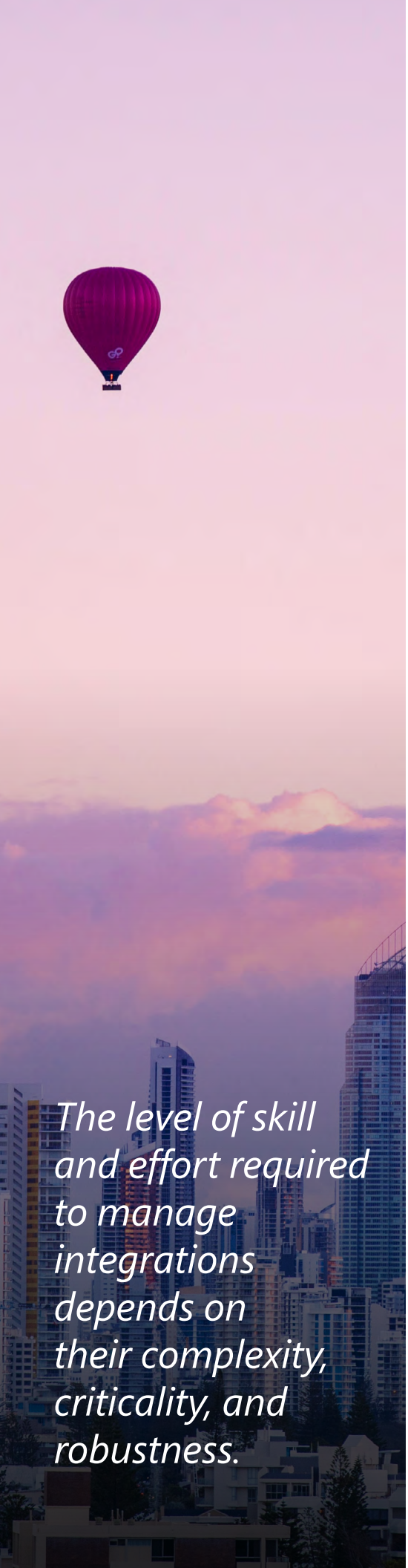
Typically, you need to apply these tasks and considerations for the following environments:

- Dynamics 365 application support environments, which are recent copies of the production system
- Test environment for testing the next versions of the application software
- Development environments
- Any related data migration, training, or integration environments
- Test environments for integrated systems

This is not an exhaustive list of environments, but should help prompt the right discussions related to supporting the environments. For a more detailed discussion on the Dynamics 365 environment strategy, refer to Chapter 9, “Environment strategy.”

Integrations

In addition to defining the environments required for supported operations, it’s worth delving a little deeper into the support requirements



specific to managing integrations. When defining the support requirements for all the integrations, consider what areas are in scope for the support organization:

- Data management at either ends of the integration
- Security of the integration services
- Performance of the integration
- Auditing or monitoring of the integration
- General troubleshooting analysis and stopping and starting of integration services

The level of skill and effort required to manage integrations depends on their complexity, criticality, and robustness. Designs of custom integrations that are tolerant of variations of data distribution and variations in volumes, are resilient to errors, and have automated self-healing mechanisms will require less support.

System maintenance requirements

As part of the definition of the system, maintainability requirements is an area that doesn't always get the attention it deserves. This can manifest itself in unplanned activity for the support organization. So, during the Initiate phase of the project, the current support organization should look to derive the specific maintenance requirements for managing the new Dynamics 365 implementation.

Because Dynamics 365 applications are cloud and SaaS-based, many maintenance tasks and responsibilities that are common to on-premises solutions are now managed by Microsoft. In general, the burden of infrastructure provision and maintenance for production systems is reduced, which leaves more time to focus on managing and improving business process performance.

Define the system maintenance requirements and what is within the scope of responsibilities for the support teams. Typically, these are in the following areas:

- Servicing the non-production environments, which can include:
  - Requesting and configuring new Dynamics 365 environments

    - Requesting and configuring database copies and restores between environments
    - Managing any customer-managed, cloud-hosted environments
    - Performing specifically requested backups
- Managing system operations, which can include:
    - Assigning users to security roles
    - Reviewing and running periodic system cleanup jobs
    - Managing system administration messages and notifications
    - Batch calendar management
    - System update calendar

Data maintenance requirements

Data maintenance is a significant part of operating a business system, and it's essential that the scope of data management is clearly defined for the support team. Much of the data maintenance will be the responsibility of data stewards from the operating business, but the support team may be expected to manage some critical areas.

Typically, in a Dynamics 365 application, a support team may be managing system administrator data, some types of primary data, security and access data, data related to periodic processes, and so on; for example, assigning a user to a security role, or adding or revising a new customer attribute.

Performance management

Performance management for a Dynamics 365 business application is a mix of tasks and responsibilities for Microsoft and for the customer. In this section, we consider the implications on the support model.

The support team needs to have some way to proactively monitor and respond to any questions from the users on performance. Therefore, the support team needs to be able to do the following:

- Understand the impact of the reported performance issue on the business
- Assign the relevant priority
- Diagnose the issue across the potential root causes from

configuration, data, custom code, independent software vendor (ISV) code, standard code, and so on to determine the root cause, or to hand it off to a specialist team (internal or external) with a meaningful starting point
- Reproduce the steps and circumstances to demonstrate the performance issue (and have the necessary resources to do so)
- Communicate the problem to other more technical specialists and work with them to resolve the issue

To better detect processes beginning to show poor performance, it's important to have a baseline definition of required performance for critical processes, also expressed as performance goals. Having actionable performance goals allows for more focused monitoring and proactive actions to stop any decline in performance quickly.

Because performance troubleshooting is a specialist skill, the internal support team may not have all the skills and experience to examine and correct the performance of custom development. It's also a skill that may only be needed sporadically, so you may need to consider planning for access to external teams that can use performance tools, interpret the results, and decide on corrective actions.

Business processes

As a business application, supporting the users in the day-to-day processes is a key function of a Dynamics 365 support organization. The support organization is expected to provide coverage across all the key business process, or at a minimum, know where to direct questions and issues that were agreed as being outside their support scope.

Based on the definition of the key business processes in scope, consider the following for each process:

- What is the level of expertise needed in the business process?
- What are the likely support tasks and queries related to this process?
- What is the type of usage, frequency, and volume?
- How critical is this business process to the business outcomes, and

what degree of priority do the support issues need?
- What is the type and level of skills needed to support this process?
- What processes require resources aggregated at a department, workstream, or value chain level?
- What are the interactions with other systems and communication protocols required?
- Does this process require the support personnel to have specialist security or compliance training or certification?
- Do any special requirements need support services outside of the anticipated support hours (outside of business hours or during holiday periods)?

An important factor when determining resources and escalation procedures is the level of ownership and responsibility that the business process leaders are expected to take from their own organization to support the day-to-day operations for their processes.

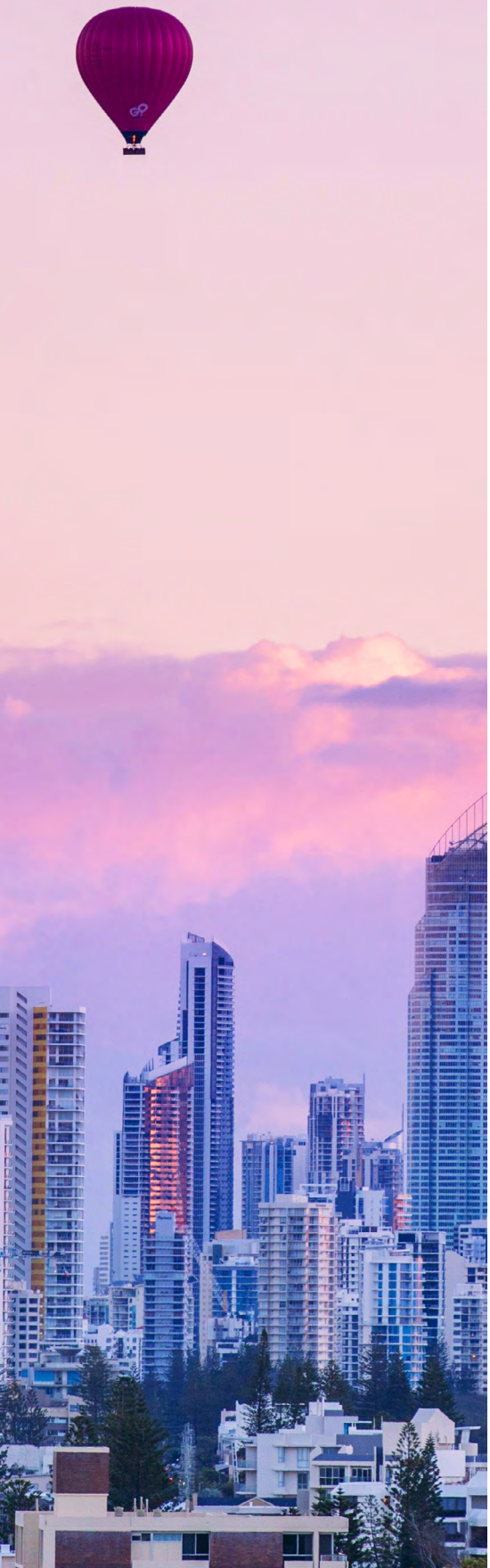
These questions and more will help shape the operating model for the support team.

Business continuity

Many organizations need to have a business continuity strategy and exercise regular checks of business continuity in case of a system-down disaster. This may be required by external regulations or by internal policies. In any case, the support organization is probably expected to play a major role.

Depending on the size and complexity of the system landscape and the types of disaster scenarios being exercised, this may need a significant amount of preparation, coordination, and timely communication between multiple parties.

As part of preparing for the establishment of the support organization, you should consider the definition of the business continuity strategy and what resources, skills, and tooling are required.



As a cloud-based SaaS service, Microsoft provides the production infrastructure, platform, and application-level business continuity services. For Dynamics 365 Finance and Dynamics 365 Supply Chain Management applications, the operations as listed in the Finance and Supply Chain Management service description provide details of the services and service-level agreements (SLAs) from Microsoft and the expected responsibilities and actions from the customer. You should bear this in mind when you build your business continuity planning, which should reflect your specific systems landscape and scenarios.

Of course, the Dynamics 365 business application may not be an isolated system—it may be connected to other enterprise systems, and your business continuity planning will need to consider your entire enterprise architecture when defining your strategy and the related support requirements.

Dynamics 365 Finance and Supply Chain Management

Dynamics 365 Finance and Supply Chain Management applications have specific high availability (HA) and disaster recovery (DR) features for cloud deployment. To ensure service availability, all production environments are protected by using default Azure HA features. HA functionality provides ways to avoid downtime caused by the failure of a single node within a datacenter, and DR features protect against outages broadly impacting an entire datacenter. When planning support requirements and procedures, consider the implications if the disaster is severe enough for the primary production site to be forced to switch over to the secondary site and the DR features are warranted. In such extreme cases, you would experience downtime for production access while access to the secondary site is being established, and the customer team is responsible for several actions. For more information about business continuity with HA and DR, see the Dynamics 365 Finance and Supply Chain Management service description. If the production system is running on the secondary site (while the primary site is being repaired), you may have restrictions (such as the inability to perform package deployments) and limitations (such as performance reduction) that the support team needs to be aware of and help

mitigate to reduce the impact on the business. You may also need to apply specific setups (such as IP allowlists) to the secondary site.

In summary, you should have a business continuity plan that includes how the support team should plan, perform, and communicate during a service outage, including during a severe disaster.

Dynamics 365 Customer Engagement

For Dynamics 365 Customer Engagement, HA provides mechanisms that can reroute requests in case of failures, planned downtime (patching and updates), and more. On the database side (relational storage), the Recovery Time Objective (RTO) for failover is dictated by the timings from Azure SQL Database. With diverse storage, the scheme extends to other storage components. For DR, Power Platform brings the ability to fail over and fail back in seconds. You can do this at the granularity of an organization (environment). There is no data loss in the case of planned DR failovers. In the extreme circumstances of an unplanned DR, well-defined processes are applied by the administrators of the Dynamics 365 (online) datacenter to recover from a service interruption; however, the support team needs to help with any potential data loss.

System update cycles

Microsoft Dynamics 365 application updates are one of the changes that the support team probably needs to manage. From the perspective of defining the scope of responsibilities for the support organization, you must understand what is involved and plan for the updates.

Creating a calendar for the Microsoft updates helps the support team plan for the resources and effort associated with the following:

- Microsoft Dynamics 365 updates
- ISV updates
- Updates for custom code
- Associated testing, including any regression testing and release management

We saw in the previous section that the solution is unlikely to remain static for long, and the solution functionality can change because of new and backlog requirements for many different reasons. For details on Microsoft system updates, refer to Chapter 20, "Service the solution."

Support models

Defining the support model is ideally started in the Implement phase so that it can be ready and exercised prior to user acceptance testing (UAT), can help with the transition during the Prepare phase, and be fully effective and working by the Operate phase.

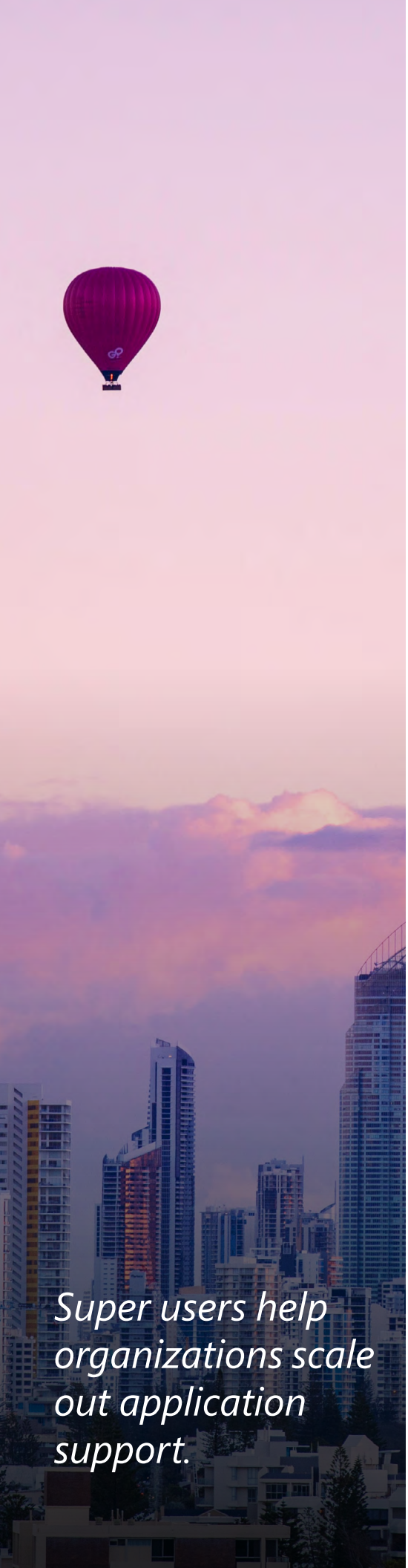
Support models

Support model options

Thus far, we have discussed topics that help define the scope of responsibilities and actions for a support team. Now we look more at the "who" and the "how," starting with an examination of the spectrum of support models.

One of the foundations of the support strategy is defining the operation model for the support organization. Some of the factors may be driven by enterprise-level strategies, including on outsourcing or existing commercial agreements. Some organizations reference independent, formal IT support and service frameworks such as Information Technology Infrastructure Library (ITIL) or ISO 20000, or other formal standards and methodologies. These are typically adopted at an enterprise level, rather than by system. Regardless of the framework, you have some key options to choose from:

- **Fully outsourced** This support model is more often seen in smaller projects that don't have many critical business processes within their scope or have insufficient internal resources to create a critical mass for any dedicated internal support members. This model requires the outsourced support organization to be more immersed in the business and understand the business well to navigate the organization when looking for decisions, diagnostic data, and approvals, as well as have a good understanding of the



business priorities. To succeed, it needs the business and partner to have a common language to communicate business needs and technical application processes between each other so that expectations remain aligned.

- **Fully internal** A support model without partner support isn't seen very often because it's a difficult model to sustain on an active project without a larger internal team that can cope with peaks and troughs of demand and has the necessary distribution of skills across business workstreams, technical areas, and servicing, including any specialist skills. If internal resources are constrained, you run the risk that the business stops being ambitious and outward looking in driving improvements and encouraging innovation. The solution can become stagnant, where assumed stability can come at the expense of extracting optimum value from the application.
- **Mixed** This is probably the most common model. It has a balanced blend of internal resources that develops expertise in the application and retains a deep understanding of the business. Partner organizations can offer strategic advice based on their wider experience across multiple implementations and provide support for any custom code, resources to cope with peaks, and specialists when required.

Considerations per support level

In this section, we break down the business considerations by each support level (**Figure 21-3**).

First level

Many customers have a concept of identifying and nominating super users within a business unit. Typically, the super users gained their deeper knowledge of the system and processes in their previous role as subject matter experts (SMEs) during the project cycle.

Super users can provide the following benefits:

- Provide immediate assistance to a user to unblock someone in most simple cases
- Filter out the simple questions such as "How do I do this?" or "Is this what I should expect?"
- Provide ad-hoc training at a business unit level

- Triage and communicate issues in a way that makes it easier for the helpdesk or resolving authority to understand the issue, replicate it, and fix it rapidly
- Collate best practices, FAQs, and issues, and provide the core team better data on areas of priority for them to tackle
- Provide early warning on issues that have the potential to escalate and help with better adoption by the business

This model helps organizations scale out application support without having to create a large central support team or pay external parties to resolve day-to-day queries on internal business process or simple application questions.

For larger business units, super users may be further divided by functional area, such as Finance or Operations. These super users are typically identified during the project phase and participate in formulating and implementing testing (such as UAT), and may also become the trainers for cascade training. This allows them to gain the necessary experience and skills to provide first-line support.

Consider formalizing the super-user role with funding, and consider formal titles or dedicated time to perform that role. Formal performance objectives and compensation also help ensure this is not just one more assigned task. Informal super-user roles don't always succeed because day-to-day operational tasks tend to be given a higher priority.

Fig. 21-3

Fully outsourced		Fully internal		Mixed	
1	Super users per business unit or function	1	Super users per business unit or function	1	Super users per business unit or function
2	Internal helpdesk to route to the partner	2	Internal helpdesk or dedicated Dynamics 365 support team	2	Internal helpdesk or dedicated Dynamics 365 support team
3	Partner	3	Dynamics 365 CoE or Dynamics 365 project core team	3	Dynamics 365 CoE or Dynamics 365 project core team
4	Partner	4	Dynamics 365 CoE	4	Partner via support agreement, or partner for custom code via support agreement
5	Microsoft support or ISV support	5	Microsoft support or ISV support	5	Microsoft support or ISV support

Second level

Almost all customers have some internal helpdesk function for IT systems, and many also have them for business applications. The size and nature of the helpdesk varies depending on the model adopted.

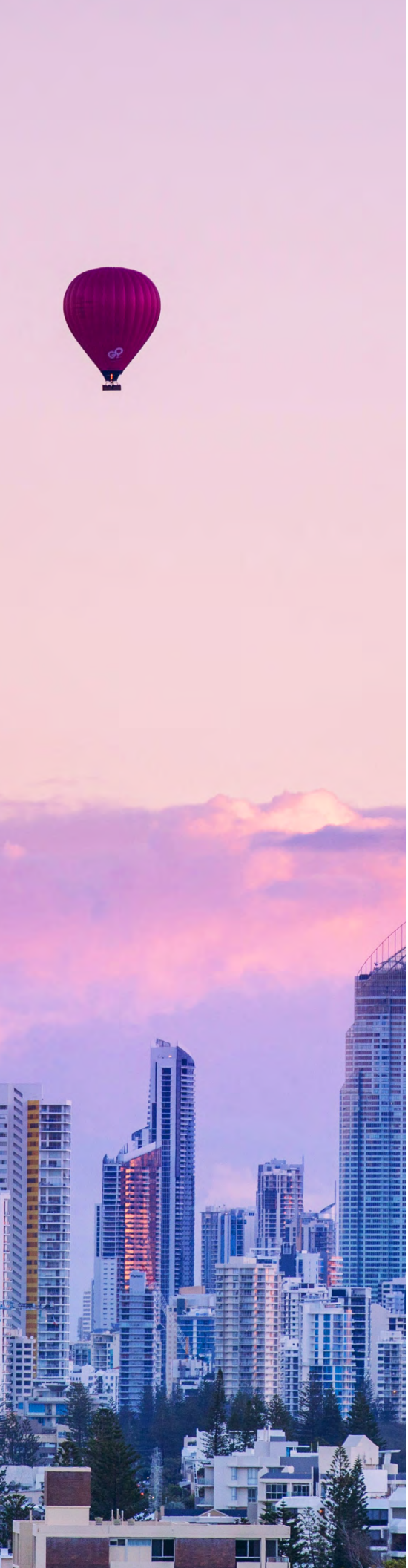
For a fully outsourced model, the internal helpdesk registers the issue and assigns it to the resolving authority (internal or external). This fully outsourced model is tougher to deliver for highly-customized business systems compared to less customized ones. Determining the correct resolving authority can be tricky in business systems—even assuming the super-user roles have eliminated the cause as a business process or training issue, you may have many different system layers, such as an internal network, infrastructure software managed internally or by Microsoft, standard Dynamics 365, custom code, ISV code, or integration software at either end.

Determining how to prioritize and who should drive the issue to resolution is a typical deficiency seen in this fully outsourced model, because very often the issue can bounce around various parties. The other consideration for business systems is managing what business data and process steps should be made available to the next resolving authority. Additionally, if the partner is driving this in a fully outsourced model, you need a clear definition of the access the partner is given to users, the business data, and what can be further passed on the next resolving authority.

In the fully internal model, the internal helpdesk also routes the issue to a small internal 365 Dynamics team that triages the issue and helps determine the best resolving authority. This team is also responsible for ensuring the issue is driven to resolution, regardless of the number of parties (internal or external) that may need to be involved. The difference is that the next resolving authority may be the internal Dynamics 365 Center of Excellence (CoE).

For multi-company or multi-project organizations, many create a CoE that works with group IT to ensure the following:

- Common standards on the Azure environment



- Operational and maintenance policies on backups, business continuity, and disaster recovery
- IT security and policies (such as password policies, MFA, SSO, and firewall rules)
- Regulatory needs, including access and security, data retention

In the mixed model, which many customers adopt, the internal helpdesk routes the issue to a small internal Dynamics 365 team that triages the issue and helps determine the best resolving authority. This team is also responsible for ensuring the issue is driven to resolution, regardless of the number of parties (internal or external) that may need to be involved.

Third level

In a fully outsourced model, the partner is responsible for triage and resolution. In the mixed model, the most common scenario is for the internal Dynamics 365 support team to determine if they can resolve the issue; if not, they work with the Dynamics 365 CoE and the partner to ensure it's well understood and correctly prioritized.

In the fully internal and mixed model, the Dynamics 365 CoE includes functional and technical experts (including developers) who can resolve issues with custom code. If the issue is seen to be with standard Dynamics 365 or an ISV, the CoE logs the issue and works with Microsoft or the ISV to get it resolved.

Fourth level

In the fully outsourced model, the partner manages the process and the customer is involved as necessary. Most customers tend to have some parts of the solution written or managed by their internal IT team (such as integrations), so you still need a definition of how the partner should work with the customer.

In the fully internal model, the Dynamics 365 CoE takes on the diagnosis and resolution if it's within their control. They only involve external parties if the issue lies with standard Dynamics 365, platform hosting, or an ISV.

In the mixed model, the internal Dynamics 365 CoE or core team typically fixes the simpler issues, but involve a partner for issues that require deeper Dynamics 365 knowledge or for complex issues.

When the issue is resolved, the method to get the fix back into production also requires a clear definition of the expected standards, testing regimen, and deployment process. Even in a mostly outsourced model, the partner drives this to a pre-production environment and the internal team deploys to production. Most customers don't give partners admin access to production environments.

Fifth level

Registering the issue with Microsoft support tends to be the common escalation point, after the determination is made that the most likely root cause is the standard Dynamics 365 software or service.

Similarly, most customers have a support and maintenance contract with their ISV suppliers. In the fully outsourced model and in some mixed mode models, the partner is contractually responsible to work with the ISV for resolution.

Support organization

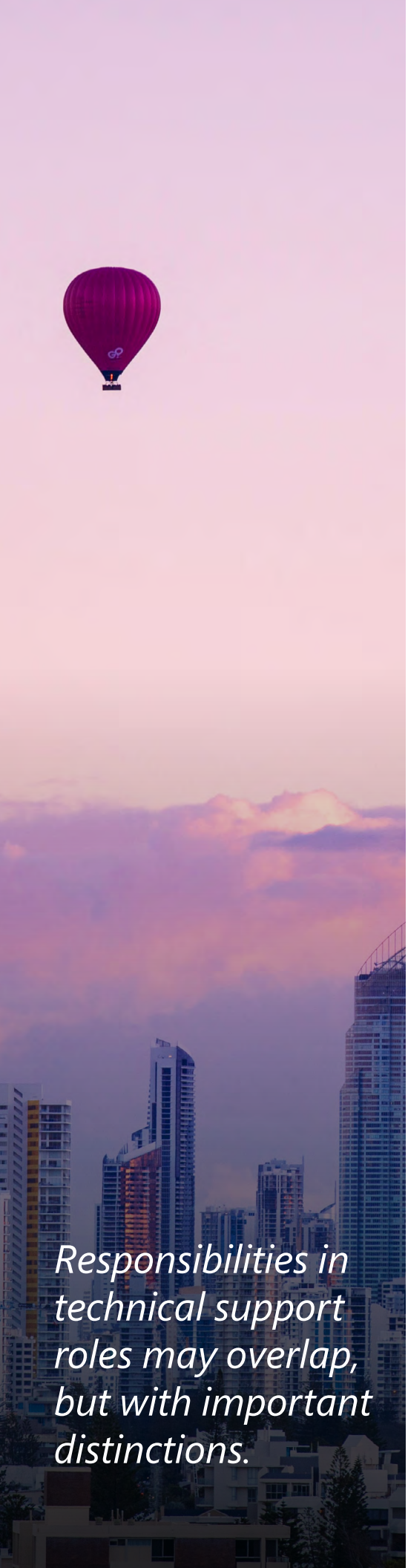
As noted in the previous section, multiple operating models exist, and within each model a spectrum of sub-options. Therefore, the specific organization structure is influenced by the variant of the model that is chosen. In this section, we explore some common patterns for roles and structures.

Support model options

Support organization

Support team responsibilities

Any organization looking to support a business application system of a reasonably-sized footprint needs to make sure that they have the necessary skills and resources—not just across the business process areas and technical topics, but also to manage data quality, maintain and improve user proficiency in the system, and help the organization remain compliant with processes and procedures.



Each of these areas needs to be defined in more detail to ensure the planning for the support model is sufficient.

Support for business process areas

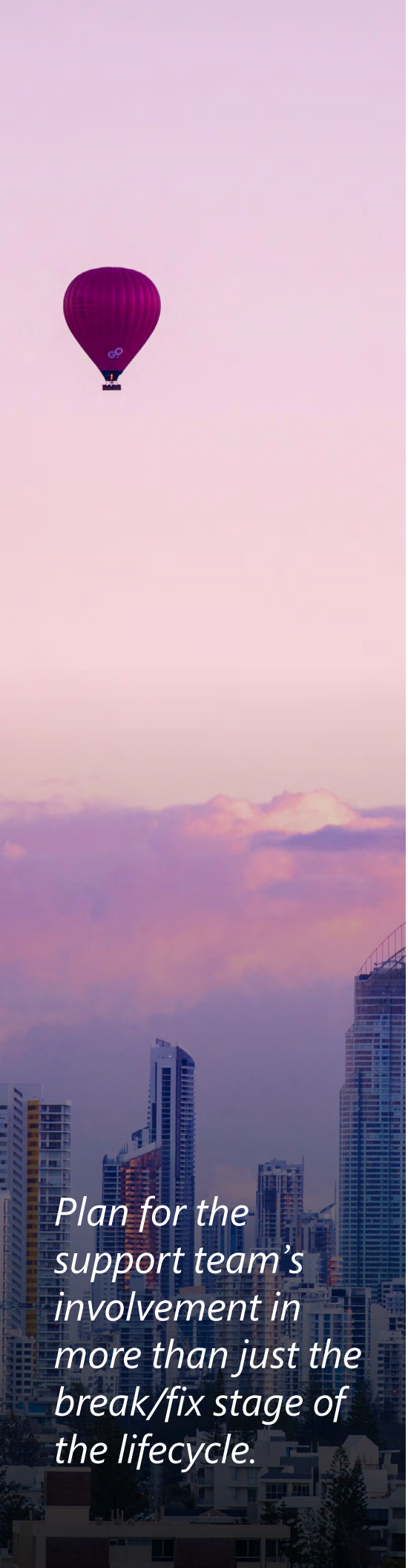
Support for business processes is usually defined by the main work-streams and includes the following responsibilities:

- Liaising with process owners on the overall health of the process
- Keeping up to date with the current business priorities in this area
- Gaining and maintaining a deep understanding of the steps and relative criticality of the various functions within a business process
- Gaining and maintaining a deep understanding of how the Dynamics 365 application and related systems deliver this process, so they can provide how-to advice and analyze potential problems
- Analyzing, triaging, and recording potential issues
- Communicating with the originator of an issue throughout its life-cycle to maintain user satisfaction, promote business awareness, and manage expectations
- Curating issues through multiple resolving authorities until resolution within agreed SLAs
- Liaising with internal and third-party resolving authorities (such as any partner organizations, ISVs, or Microsoft)
- Monitoring the lifecycle of a defect resolution from registering to promoting a fix

Support for technical areas

Support for technical areas is usually defined by the following categories:

- Service operations (database copies, new environment creation, managing system downtime windows, maintaining Dynamics 365 system quality update calendar)
- System operations via services (integrations, batch processes, reporting, business intelligence)
- System maintenance and monitoring (system health monitoring, security role maintenance, security certificate rotations, periodic system cleanup of log files)
- Dynamics 365 application management related to break or fix errors reported by users
- Dynamics 365 application management such as updating system administration parameters



- Dynamics 365 application system update management (code branching, any software development, unit testing new changes)

Responsibilities may also overlap with those for business process support, with the following distinctions for the expected areas of understanding:

- Gaining and maintaining a sufficient understanding of the relative criticality of the various functions within a business process
- Gaining and maintaining a deep understanding of the underlying technical processes involved in the system functions (especially integrations, custom code, and critical processes)

Support for data quality maintenance

In many organizations, some of the business data maintenance responsibilities are passed to the support team by the business data stewards or owners. This type of support includes the following areas:

- Core business data and configuration in the system as well as IT and system administration parameters
- Liaising with business data stewards to determine priorities and actions
- Periodic audits and reports to confirm the data quality

Support for user proficiency in the system

In some organizations, the functional experts within the support team act as a default team to provide user training both formally and on an ad-hoc basis. This level of support includes the following:

- Monitoring user proficiency within the system by reviewing the number of how-to queries and issues created by not following the agreed business process in the system
- Creating training content and conducting formal end-to-end training of new users
- Creating training content and conducting informal specific functional training of users on request or as needed
- Conducting change management activities to help improve user adoption

New features and developments

In addition to managing the day-to-day and periodic maintenance tasks and addressing queries and issues in the preceding areas, the

support team may also be expected to keep track of new features and developments (positive and otherwise) that may impact the system.

Often, the planning for the break/fix part of a support team's organization is well thought through, but the planning for other areas of responsibility may be neglected. For example, many support team members may also need to be involved in helping with the lifecycle of the next system update. This may include the following duties:

- Helping with assessing the impact of proposed changes on the current system
- Providing insights into areas that need reinforcement based on statistical analysis of the areas with the most issues
- Testing the changes
- Updating documentation to reflect process and system changes
- Training in any new processes
- Sending communications of the system update
- Managing the next release through the development, test, and production environments
- Refreshing the support environments with the latest production environment

Dynamics 365 business applications are cloud-based SaaS applications, so Microsoft handles a lot of the traditional effort (for on-premises systems) of managing all the layers of the production infrastructure hardware, operating system, and application software, and much of the monitoring of the underlying system health.

The duties and responsibilities outlined in this section aren't meant to be exhaustive, but rather prompt you to make sure you have sufficient definition of the coverage, skills, and effort needed to maintain and support a business application. Later in this chapter, we discuss the special considerations in the transition period and in the early days of a new system implementation (sometimes called hypercare).

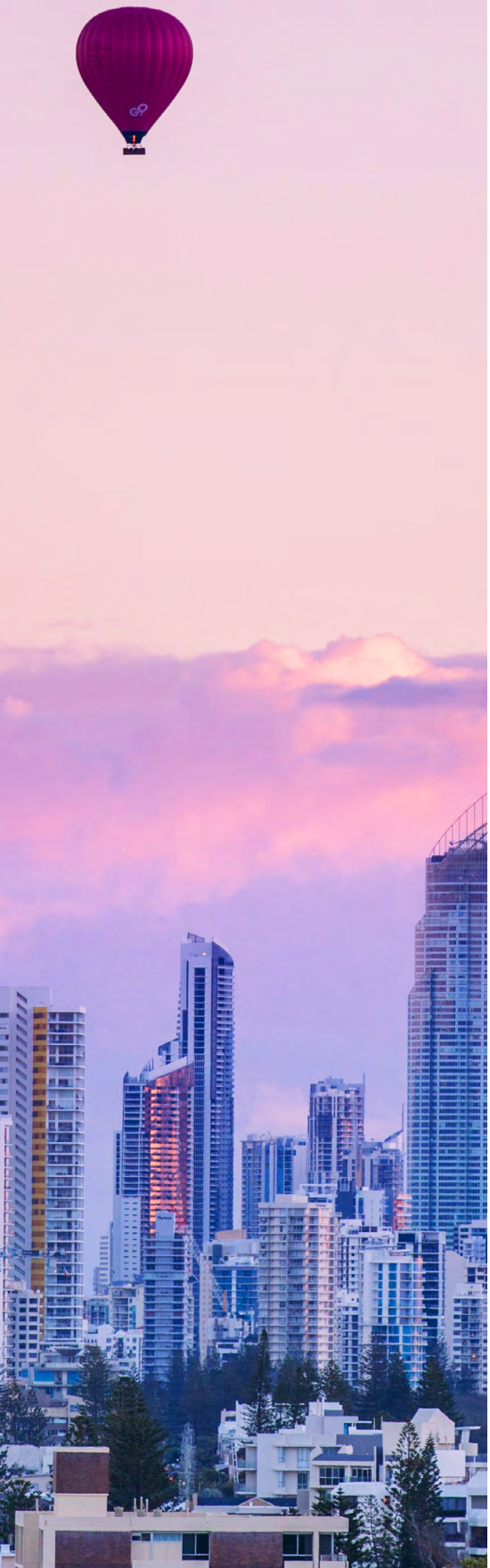
This section concentrated on the tasks and activities that your support organization may need to cover. For details on operational considerations, refer to Chapter 20, "Service the solution."

Support organization structures and roles

When considering the support organization, each business should consider the level of effort that is implied by the range of duties expected as part of support and determine how to distribute the various responsibilities.

The following is a typical set of roles (depending on the size and complexity of the implementation, sometimes multiple roles may be filled by a single individual):

- **Business super user** As we discussed earlier, super users serve as first-line support with some form of part-time role. The level of formal recognition of the role varies from customer to customer.
- **Business process expert** Also called SMEs or functional leads, these experts in the support team usually have a full-time or near-full-time role. A lead (technical architect) usually has oversight on the full solution architecture across all the business process workstreams.
- **Technical architect** Also called system architect or solution architect, this role within the support team ensures the enterprise and Dynamics 365 technical standards are met. They have oversight across all the technical tasks and duties mentioned earlier in this chapter. The technical experts, developer, and release manager roles often report to the technical architect.
- **Technical expert** This role has expertise in the technical servicing and support duties that we discussed earlier.
- **Developer** This role is responsible for bug fixes to custom code and writing code for new custom functionality.
- **Release manager** This role is responsible for confirming that a release has been appropriately validated for release to production. They're also often responsible for other servicing and support tasks.
- **Support manager** This manager is accountable for the support services for Dynamics 365 applications (and often other related systems).
- **Business and IT stakeholders** You should consider these roles significant because their sponsorship, direction, prioritization, and decisions are key parts of establishing the right support organization, defining fit-for-purpose processes for support, and meeting business expectations.

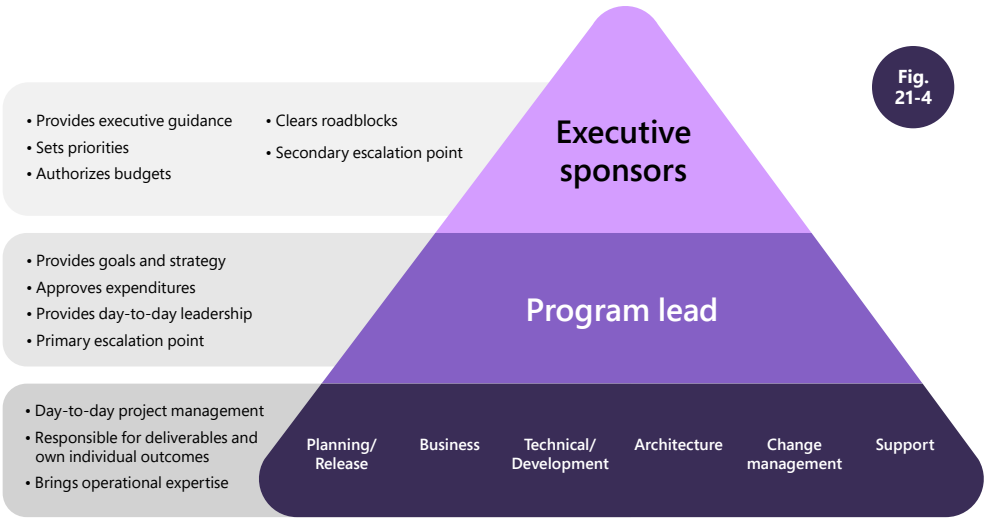


Depending on the support model chosen (fully internal, fully external, or a mix) and regardless of the organization responsible, you will likely need these roles.

Organizations that are implementing across multiple business units or companies with an incremental rollout plan should consider setting up a Dynamics 365 CoE (**Figure 21-4**). This allows the organization to bring together specialist knowledge and make it available across different business, where each individual business or implementation may not have been able to justify it. This model gives economies of scale on constrained and specialist resources, and helps disseminate good practice across implementations. In a CoE model, you may have permanent internal roles or create sub-roles that have deep specialist knowledge in business and IT processes. Because the Dynamics 365 CoE can employ its own specialists, it can also help manage other specialist third-party organizations providing support and service.

A common approach is to outsource some of the Dynamics 365 application functional and technical support to a Dynamics 365 partner. This is especially true if the application has a lot of customizations that were developed by that partner. In these circumstances, the customer may still elect to keep a small internal support team of a technical architect, developer, and release manager to help manage the outsourced services and maintain final control over the quality of the changes to the production system.

The size and nature of the support organization should be determined



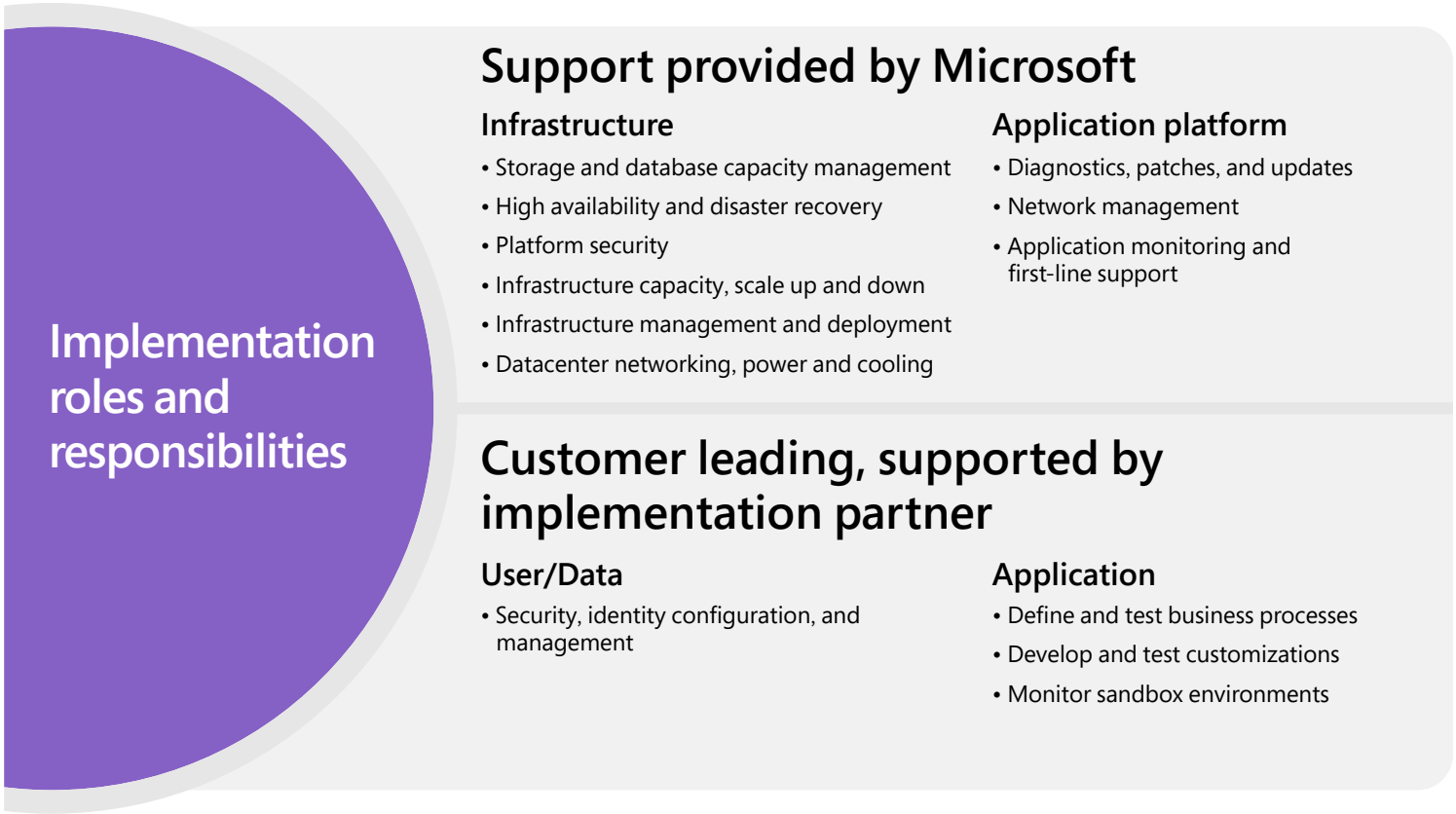
by analyzing the scope of the requirements for the support organization (as discussed earlier in this chapter) as well as the size, complexity, functional spread, and geographical (time zone, language, local data regulations) distribution of the implementation.

Standard Dynamics 365 platform and application support and servicing for production environments is also available (**Figure 21-5**). Microsoft offers different levels of support for your Dynamics 365 products. Refer to Chapter 20, "Service the solution," for more details.

Budgets, contracts, and agreements

When a new system is introduced into an existing enterprise architecture, some changes can impact related systems, such as the changes required to budgets and support agreements. A common issue that can occur is that the budgets and agreements aren't modified to reflect the new system environment and processes. This can cause delays when addressing issues that may span departments, or if the resolving authorities or third parties have their own budgets based on previous processes and agreements.

Fig. 21-5



After you have a reasonable draft of the new system solution blueprint,

to-be enterprise architecture, and support organization, you should start defining the budgets related to the new tasks and areas of responsibility.

Consider the full set of tasks and activities in scope for the support organization (as discussed earlier) and map these to the various roles and resolving authorities over the full lifecycle of a support job or request so that no gaps appear in the flow. You can use this to make sure that the specific responsibilities of all the roles and resolving authorities can be mapped to agreements and service-level expectations.

For internal parties, this may mean defining budget and resource splits and less formal service delivery agreements. For third parties, that may mean formal contracts with formal SLAs. Mismatched expectations between the customer and a third-party support provider in the middle of dealing with high-priority issues are not uncommon. A contract created based on a deeper understanding of the expected tasks and service expectations is much more likely to avoid misaligned expectations and provide a faster path through the process.

Support operations

The preparation and design work allows you to provide the level of support expected by the business. There are many aspects to consider when planning for the practicalities of operating the support services. Preparation for support is necessary, but not sufficient. The support team also needs to learn and practice to ensure they are ready for the extra pressures of the period immediately following a go live. The next section explores this in more detail.

Transition

Support organizations for business applications are rarely fully formed on the first day of go live. You normally have a transition period from project mode to support mode. However, the transition period often starts later than it ideally should, and the quality of the transition is often not sufficient to provide the level of service that is expected. This results in poor user experience, frustration from the support team, and

a drag on the project team that is called into support requests long after the transition is supposed to be complete.

What is a good time to start the transition?

Learning a new business application system, even one that is as user-friendly as Dynamics 365, can take time, and actively using the system is one of the best methods. For the business process-related support roles, if the SMEs are expected to move into the support role at go live, training happens naturally as part of the project, assuming the project approach includes a significant amount of hands-on system involvement by the SMEs. It's important that the business process experts don't just become consultants to drive business requirements and have very little exposure to how the system design meets the requirements.

If, however, the existing SMEs supporting the legacy system are expected to support the new system, consider formalizing their involvement in the project during the Implement phase, from the early build steps and especially in shadowing the business functional leaders at critical points in design and software playbacks.

If the SMEs aren't being exposed regularly and extensively to the system during its build, a gap will likely appear in their understanding and ability to support, which means increased reliance on the partner functional consultants for support (and testing).

Furthermore, consider involving the support business process roles in the formal testing phases as authors of some of the test cases or in conducting the testing. Other opportunities to get the team to directly experience using the new system can come from creating task recordings and task guides in Dynamics 365 Finance and Supply Chain Management applications, or other forms of training documents.

You can apply a similar approach to the technical roles—they can be involved in shadowing the project team at critical points and be given project tasks to complete (under the supervision of the project team).

Consider setting intermediate targets for transition readiness throughout the project so it doesn't become a last-minute process. It may help to exercise the whole support organization prior to go live and to review the effectiveness of the readiness for supporting the production system. Consider using the formal support processes during the later stages of formal testing (such as UAT) to review readiness. This way, a representative sample of issues raised during this test period can be

recorded, triaged, and resolved through the support organization tools and procedures. The project team will be expected to participate as resolving authorities to help train the support team and ensure that issues are addressed in a timely fashion. The learning from this type of exercise can help make the transition more effective than last-minute handover training sessions.

Most organizations run lean support teams, which doesn't offer many opportunities for the team to participate in the transition activities we described without explicitly planning these activities so that the support team gets time away from their day-to-day operational duties to participate fully during the project implementation.

Requirements management

When the application is in production, it doesn't mean that the system solution can now be considered static for the next few years. The solution will need to deal with new requirements for several reasons:

- Some projects implement a minimum viable product (MVP) on their first go live with the goal to incrementally add the lower-priority requirements over time
- Some projects have planned multiple rollouts that may impact the already live areas
- Some changes to Dynamics 365 are driven by changes in connected systems in the enterprise
- Businesses need to react to the changing world around them
- In a modern cloud SaaS world, incremental functional and technical updates from Microsoft and ISVs help keep the customer's solution secure and updated

Some organizations will continue to have a project team that is responsible for reviewing, prioritizing, and delivering the backlog, but many won't have a standing project team, or the project team will be purely focused on the next rollout.

In such cases, the support organization needs to have the resources

and means to keep the solution up to date for the users currently using the live system.

In any case, the support teams are often at the front line of the feedback and requests for new requirements. At the very least, they need a way to capture and channel the new and emerging requirements to the right parties.

Change management

We saw in the previous section that the solution is unlikely to remain static for long, and the solution functionality can change because of new and backlog requirements.

However, support teams may need to accommodate other sources of change in the production system, such as the following:

- Core data changes (new customers, suppliers, items as part of normal business operations)
- System parameter changes (such as in response to the growing volume of data)
- Configuration changes (such as in response to regulatory changes)
- Approval process changes in the system (such as in response to internal policy changes)
- Bug fixes

Some of these changes can be made within the agreed change control process enforced and facilitated by the system itself. In other cases, proposed changes need to be reviewed, prioritized, and approved by business or IT stakeholders and any other relevant parties.

This process should be considered as part of defining the scope of the support teams' responsibilities. It should also prompt the organization to create the appropriate change management boards for changes to the system in operational mode.

Hypercare

Most projects define a short period of time after go live in which

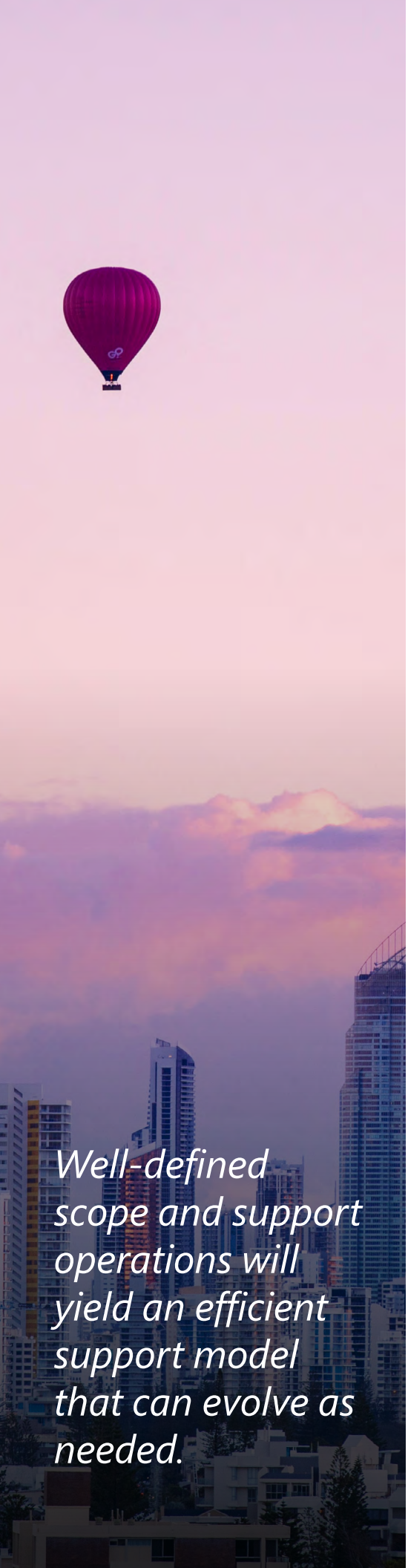
additional resources and extraordinary attention are given to supporting the business. This hypercare period is necessary to ensure that the business can operate within acceptable thresholds while addressing any initial growing pains with a new system and new business processes.

The role of the support team during this period is critical, and it's important that the expectations are clearly set out and agreed upon with all parties:

- Define a clear exit strategy for the end of the hypercare period based on meeting explicit criteria. For example, criteria might include no open P1 issues without an acceptable action plan, key business processes are operating at an acceptable efficiency rate, over 90 percent SLA targets being met for P1 and P2 support requests, or the support team can resolve over 95 percent of support requests without using resources reserved for hypercare only.
- Decide on an end date for the hypercare period and have an action plan in place to meet the exit criteria.
- Define expectations and an explicit plan for the project team (and partner resources) during the hypercare period so you can get reliable help from the team. Otherwise, you may find that the project team is committed to working on the next phase of the project and will resist being drawn into support issues.
- Make sure the support team gets the necessary documentation and top-up training from the project team and implementation partner during the hypercare period.

Conclusion

During operational use of the Dynamics 365 system, the support operations are expected to function efficiently and evolve alongside the expanding use of the system. For that to happen smoothly, the preparation and definition of the support operating model are essential precursors.



Scope of support operations

The sections in this chapter are designed to help define the potential scope of support operations; you need to convert each of the areas covered from the "what" to the "what in detail," "who," "when," and "how."

You should convert the enterprise architecture alongside the business and IT policies and business continuity requirements into practical reference architecture and detailed procedures with reliably repeatable steps that the support team can implement day to day.

You should convert the Dynamics 365-specific areas and business process requirements into the support organization charter or terms of reference with a further detailed definition of the tasks, the roles responsible, and the standards expected to keep the system operating safely and efficiently.

Use the system update cycles section, including managing new requirements and changes, to define the means by which the Dynamics solution can stay updated and evolve with the additional and improved features. This should deliver a detailed definition of the tasks, the roles responsible, and the standards expected to continuously improve the business value from the Dynamics 365 system. This is important to ensure that the support organization can keep the current operational system functioning optimally while also working in parallel on the next update.

Support operating model

The sections discussing the support model and support organization should help take the definition of the support operations scope and use the chosen support model to create the shape and size of the support organization. This resulting design should be a good fit for the requirements of support operations.

Finally, transition guidelines can help you make sure that the transition from project to support happens incrementally and through practical experience of the system during implementation. We also encourage

you to validate the readiness of your support processes and organization prior to go live.

Operating considerations

The nature and details of the support tasks are influenced by the specifics of the business application because they are so embedded in the core business. The details of the support operations are further affected by the details of the circumstances of the business being supported. We discuss some of the circumstances next.

Routine maintenance and service operations

Consider defining a support and servicing shared calendar that lists the regular, periodic tasks. This can help make sure those tasks are reliably implemented and have the right visibility across the team.

Service hours, language, and geography

In addition to defining the normal hours of operation, consider how to define out-of-hours service, especially for urgent issues.

Some larger organizations, especially multi-national ones, may need to provide support in multiple languages and across multiple time zones based on the geography of the users.

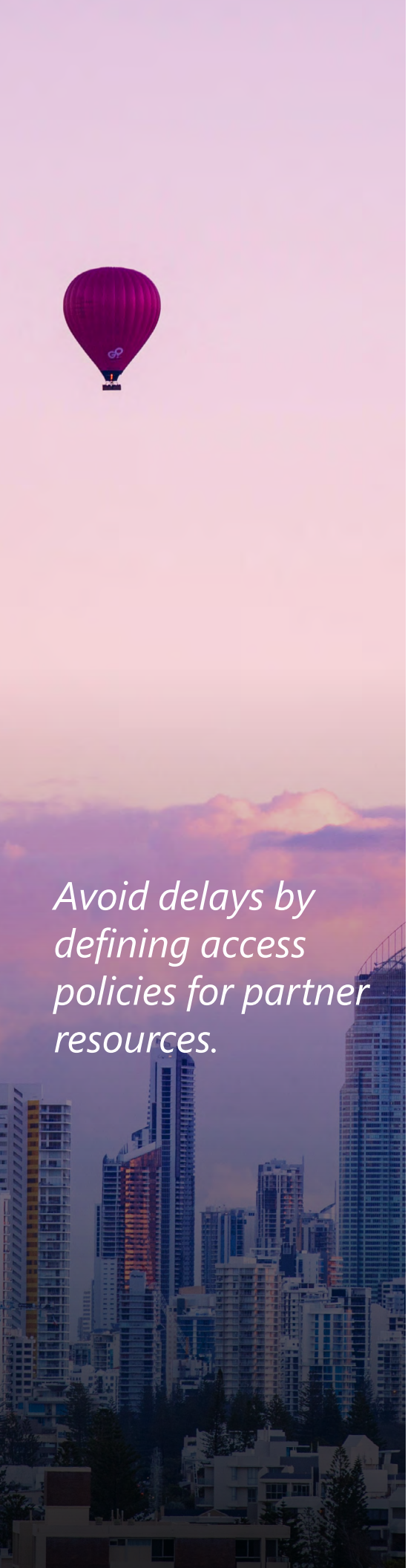
Furthermore, consider business-critical periods when support may need to provide out-of-hours support; for example, during seasonal peaks or period-end or one-off events such as company acquisitions.

All of these topics have a bearing on the operational patterns that the team needs to support.

SLAs, priorities, and escalations

Support organizations need to provide some form of SLA for the different types of support request—if these terms are based on genuine business criticality, they will help drive the right prioritization.

Establishing a formal escalation procedure with defined criteria will also



help prevent frustration on slow-moving cases and help mitigate the impact of those that escalate issues beyond their true business impact.

Tools and access

Most support organizations have some type of internal helpdesk software that is used to manage support requests across all IT services. Most continue using their existing helpdesk for Dynamics 365 applications but need to consider new internal and external resolving authorities. Third parties such as partner support organizations and ISVs have their own helpdesk systems. When the issue is diagnosed as likely to be in standard Dynamics 365, the recommended method of registering the issue with Microsoft is to use the in-application process: Dynamics 365 Finance and Supply Chain Management Support and Dynamics 365 CE Power Platform Support. Tracing a support request across multiple technologies and especially across third-party organizations requires some deliberate planning and possible configuration so that the internal helpdesk has a reference from start to finish of any ticket.

Policies on access for partner resources to the customer systems and customer data need to be defined as part of the support organization so that there are no delays in the middle of dealing with high-priority issues while policy is reviewed and special approvals are obtained.

For example, if the policy only allows for partners to view anonymized customer data, having an automated way to copy the production system that includes anonymization will help reduce delays when troubleshooting or testing.

Checklist

Support scope

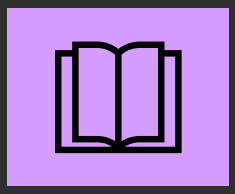
- [ ] Establish a practical reference architecture and detailed procedures with repeatable steps for the support team that can be operated day to day.
- [ ] Identify the impact of the business and IT policies on the scope and the impact on the policies after the Dynamics 365 application is in operation.
- [ ] Identify the Dynamics 365 application-specific functions and procedures as well as business processes that need to be supported.

Support model and the related support organization

- [ ] Define the type of support operating model to adopt and the nature of the role of the internal and external support teams.
- [ ] Create a support organization charter with a detailed definition of the tasks, roles, responsibilities, and standards expected to keep the system operating.
- [ ] Consider the implications of supporting a cloud and SaaS application, and potentially including a Dynamics 365 CoE in the support model.

Support operational considerations

- [ ] Establish a transition strategy that drives the support team to participate in the emerging solution and improve their support readiness over the project lifecycle, rather than a limited handover close to go live.
- [ ] Confirm the readiness of the support team's operating model, resources, knowledge, and procedures during UAT.
- [ ] Have a defined strategy to manage the critical period just after going live (hyper care) with explicit entry and exit criteria.
- [ ] Define how to keep the solution up to date and evolving with additional and improved features and capabilities.
- [ ] Determine a support and servicing calendar for regular and periodic maintenance activities.
- [ ] Identify the normal hours of operation across different locations, as well as out-of-hours service windows, multi-language and time zone requirements, seasonal peaks, period-end, or one-off events.
- [ ] Establish formal escalation procedures with defined criteria, SLAs, and different tiers of support, including internal and external support.
- [ ] Determine the tooling that the support organization requires.
- [ ] Identify the policies to access data and environments, either by internal support teams or by external support teams.



Case study

A story in two acts

Over the last few years, IT project implementation methodologies have been changing to try to keep pace with the speed of innovation of software in the cloud, especially Dynamics 365 applications. IT organizations within businesses have similarly been evolving as the cloud is changing the role and areas of responsibility for IT.

The following case study is an illustration of the dangers of assuming that an existing, well-functioning support model will continue to perform well in a different architecture and with changed business processes in new applications, without deliberate planning and action. This is doubly true in a cloud world.

Act I: The journey of learning

This is a story of a medium-sized organization (fewer than 1,000 employees) with multiple offices across three European countries and their corporate headquarters in the UK. Their core business includes professional services, field services, and bulk distribution from their own warehouses.

The group was undergoing rapid business growth. To support and accelerate this growth, they chose to replace several specialist local applications and multiple on-premises ERP and CRM systems with Dynamics 365 Finance, Dynamics 365 Supply Chain Management,

and Dynamics 365 Customer Engagement across the enterprise. They planned their first rollout to locations in two countries that had significant interdependency (accounting for approximately 15 percent of their business) and to the other businesses incrementally, as part of the next phases of the project. Subsequent phases would also include additional functionality related to Human Resources, Payroll, the Customer Service Center, and shared services on contract management.

The existing support teams were fragmented across various systems, and sometimes also by business unit. These multiple support teams were planned to be organized under a unitary structure in IT before the transition to Dynamics 365. All support activities and teams were planned to be managed by a director of support services, and who would report to the CIO.

The project was planned to go live approximately 10 months after project initiation. At the exit criteria review of the Initiate phase, they identified the following:

- The to-be end-to-end business process flow diagrams and description were adequate in most areas.
- The solution blueprint document was judged to be sufficiently well-defined to confirm the scope of the overall design.
- Budgets and project resources to complete the project were revised and approved in light of the latest solution blueprint and project plan. The project scope and go-live date had been reviewed and reconfirmed by the steering group.
- In readiness for the implementation of enterprise-wide Dynamics 365 applications, the ambition was for the support team to be reorganized by main business areas: Finance, Sales and Marketing, Operations, Field Service, Warehouse and Shipping, IT business application, and IT servicing, rather than by legacy system. However, in reality, the teams continued to be tied to the legacy systems they had existing expertise in supporting.

There was an informal understanding that the support team would be trained during the project. However, during the Implement phase, the project team stated that they were behind schedule and too focused on project delivery to help educate the support team. The mitigation

plan was for the support team to be part of an extended, one-week training and hand-off period after UAT was complete.

As the implementation progressed, the support organization continued to focus on the day-to-day needs of the currently operating legacy systems. There was a general feeling in the project and in the support organization leadership that the risk was low because the existing support team was experienced and some of the project's SMEs would join the support team, thereby seeding knowledge into the team. The underlying assumption was that the support team would be able to pick up the new duties without too much fuss, given that they had been working in a support role for a few years.

During the Prepare phase, when some members of the support team were asked to assist with the testing, the team didn't feel ready to participate until they had been through the training, which was scheduled after the testing. The project SMEs didn't feel they could conduct the internal training because they also had very little hands-on experience in the working system.

When UAT was complete, the project team had other unplanned activities that were considered to be higher priority, and so the training and handover to the support team was reduced to two days.

The go live went ahead on schedule, but the support team struggled to adequately support the business. The initial assumptions that the support team would just pick up the new system support were found to be mistaken, because the underlying processes had also undergone significant changes. The support team hadn't been informed about all the business process decisions and weren't familiar with the new control and approval requirements configured in the Dynamics 365 application. The shortened training and handover were conducted without reference to the significantly changed business processes or to the new enterprise architecture, so didn't provide the necessary context to support a production system.

The support operating model had been revised, in theory, from the distributed, system-level team structure to be more business process

and enterprise IT-based, but it wasn't exercised during the Implement or Prepare phase; the gaps and deficiencies were only exposed once the teams had to operate in production. The new operating model had also not adequately considered the implications of changed requirements of the new business processes, architecture, and policies.

In the previous support operating model, all queries and issues came directly (and with little structure) to an individual support team member, and the full lifecycle of the issue was performed with little external visibility. In the enterprise-level support, even with the first limited rollout, the support team was receiving large numbers of administrative queries, which reduced time to address the more business-critical tickets.

The team was also forced to revise IT servicing procedures to reflect the new technology implications in a cloud-based SaaS model. They had to revise policies related to data retention and backups because processes based on the legacy, bespoke on-premises processes weren't reviewed and updated for the new cloud-based SaaS architecture.

The project team had planned to start working full-time on the next phase of the project after only two weeks of hypercare. The budget hours for the implementation partner hypercare support were planned to last four to six weeks, but were exhausted within two weeks or so because the support team needed far more of their time. Many SLAs were missed in the early weeks, and the business was forced to formally extend the hypercare period for both the project team and the implementation partner because trying to get the timely attention of the implementation teams was difficult given that they had other planned priorities for the next project phase.

After the P1 and P2 support tickets were resolved and the volume of new tickets was down to a manageable level, the post-go live retrospective review made the following key recommendations to be enacted as soon as possible, to improve the support outcomes during the next rollout phase:

- The scope of the activities, tasks, and actions expected from the support team should be explicitly defined, in the context of the new system, processes, people, and architecture.

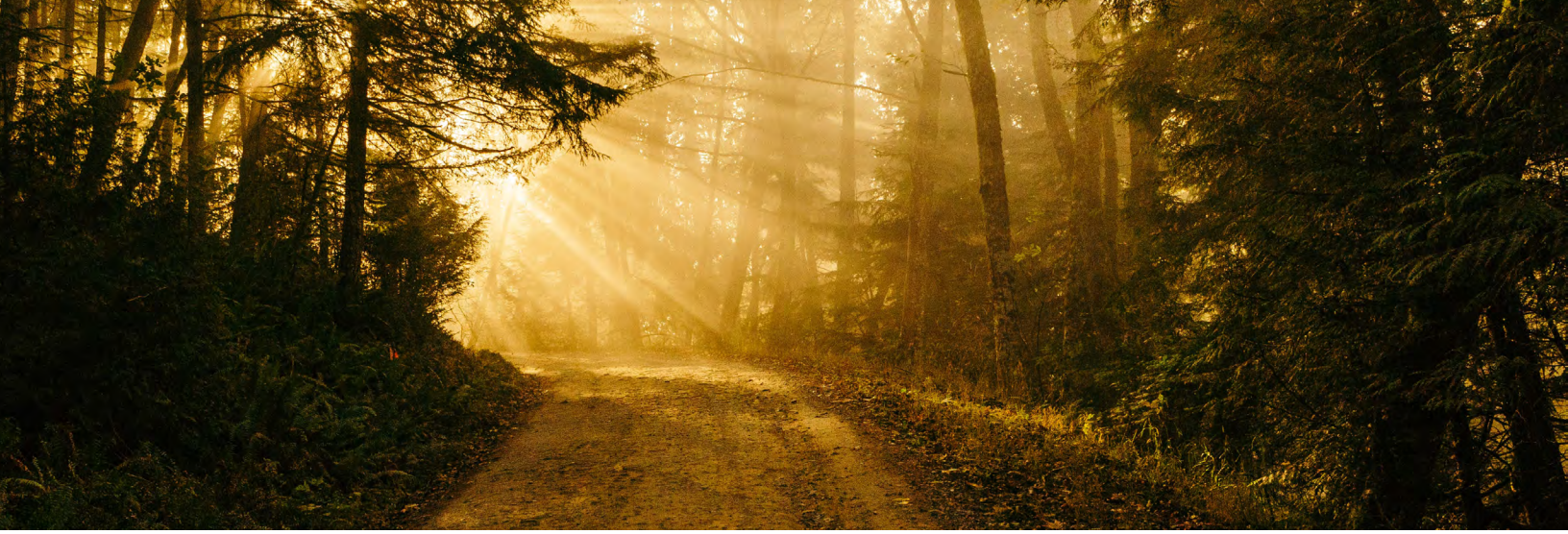
- The support operating model needs to have a more formal definition of the different level of support and a clearer agreement on the expectations from third parties. For example, using tools such as Microsoft Teams channels where super users can help address the higher-volume, simpler questions quickly. The super users would also help to better formulate issues and questions that needed to be addressed by the next support level.
- The support team needs to be formally allocated time to be involved in the implementation project to ensure their learning is sufficiently deep and is embedded in the new process and architectural context to be usable in production. The transition should not rely solely on a last-minute, half-hearted handover.
- The transition should be formally planned within the project plan with specific tasks (such as process design reviews and test case preparation) so the team has sustained and multiple opportunities to have hands-on experience of the developing system.

Act II: Fewer problems = shorter story

These recommendations were approved by the stakeholders and enacted over the following months. Even though the next rollout (six months later) had a much larger user base, it was a much smoother operation with a shorter hypercare period, higher user satisfaction, and markedly less cost than the first go live.

This story exemplifies the adage “Poor service is always more expensive than good service.”

Conclusion

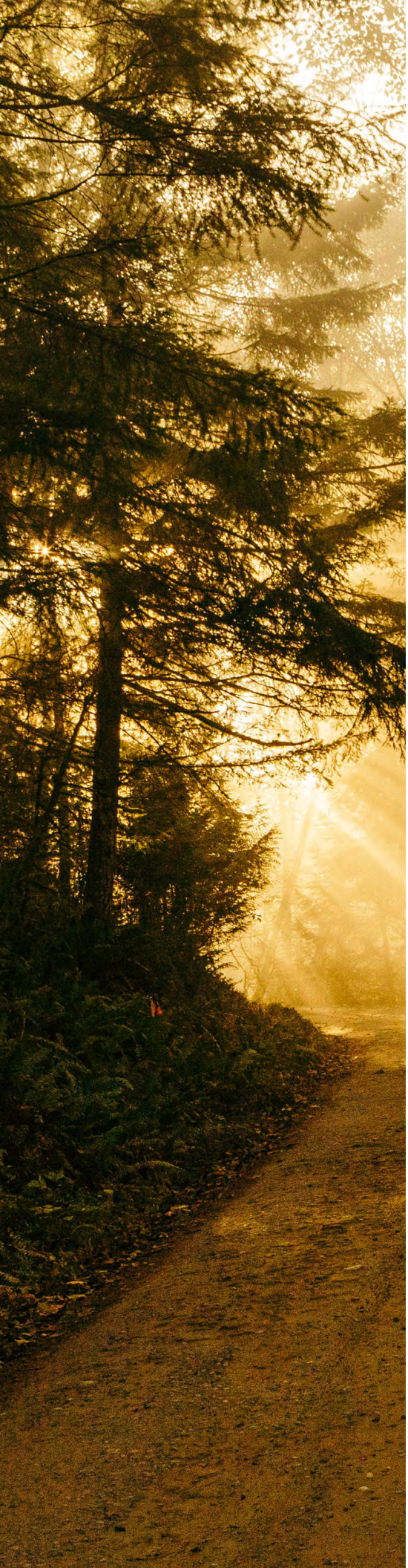


Conclusion

As we have seen in numerous examples throughout this book, decisions taken during the project implementation lifecycle can have dramatic impact on overall project success.

For example, the case study in Chapter 14, "Testing strategy," describes the negative business impact resulting from deprioritizing performance testing. What I love about this example is that it describes a situation we see all too often. Perhaps it's logical to think that performance testing is optional because the first deployment is a limited go live, but as the case study shows, it's important to prioritize performance testing before Phase 2 when more users are onboarded onto the system.

Every decision taken during a project implementation ultimately comes down to risk; the decision is either unknowingly adding risk or helping intentionally manage risks during delivery. What we have attempted to do with the Success by Design framework is provide you with a very clear set of guidelines and practices that will help you manage risk and ensure that you have approaches to mitigate all risks that show up as part of a solution implementation. These practices and the prescriptive approaches are based on years of experience gained by our team of over 200 Dynamics 365 solution architects, having overseen thousands of complex implementations over the last many years.



At Microsoft, our passion for building the most innovative products is rivaled only by our commitment to helping our customers and partners bring these products to life so we can *jointly* realize the opportunities presented by digital transformation. The Success by Design approach is a framework, and like all frameworks, it's far from perfect. We invite you to take this guidance and apply it to your projects—by making it your own and adding to it what we may have missed. In the spirit of jointly building the body of knowledge that drives our Implementation Guide, we invite you to engage with this content and let us know how it compares to your experiences in implementing Dynamics 365 solutions.

Your feedback is critical to helping us evolve this guidance, so please don't hesitate to reach out. You can email us directly at d365implementguide@microsoft.com.

We're immensely proud of the body of knowledge that our FastTrack for Dynamics 365 team has formalized through Success by Design, and even more excited to share this knowledge in the first iteration of this book. What you've just read is only the beginning—we will ensure that this guidance continues to reflect product truth and includes the latest best practices gleaned from the thousands more implementations that we will run in the coming years.

While we may never fully eliminate the nail-biting anticipation that accompanies most go lives, I hope that our unique take on the fundamentals of successful implementations will bring more certainty, predictability, and stability to your digital transformation journey. Thank you for partnering with us on your journey. I look forward to hearing from you as you put this guidance to practice.

Faisal Mohamood

General Manager, FastTrack for Dynamics 365

Microsoft

Acknowledgments

This book celebrates Microsoft's desire to share the collective thinking and experience of our FastTrack for Dynamics 365 team, which is currently made up of more than 140 solution architects who serve Microsoft Dynamics 365 customers and partners around the world.

For being the kind of leader who purposely carves out time in your busy schedule to think about, write about, and share your vision for Dynamics 365 and the Microsoft Power Platform, thank you, James Phillips. We hope that the way you've inspired us comes through on every page of this book.

For your tireless efforts in making Dynamics 365 a better product and for making us better solution architects, thank you, Muhammad Alam. We're also grateful for the time you gave us to round out the content in Chapter 1.

For encouraging us to embrace James Phillips's vision of making Dynamics 365 Success by Design for all, and for supporting us in bringing this book to life, we thank Faisal Mohamood, Swamy Narayana, and Jason Kim. Successful leaders prefer details over executive summaries—the reason we know this is because of you. Thank you.

Thank you to the book's authors, including Ajit Kotwal, Alejandra Cabrales Orozco, Alejandro Gutierrez, Alok Singh, Aman Nain, Amreek Singh, Dave Burman, Jeremy Freid, Jesper Livbjerg, Matthew Bogan, Pedro Sacramento, Rich Black, Richa Jain, Satish Panwar, Saurabh Kuchhal, Seth Kircher, Tak Sato, Veselina Eneva, and Vidyasagar Chitchula. At the start of this journey, you were given the task to write the truth into these chapters, to trust what you know, and to make sure your message is valuable to the reader. That's exactly what you all did, in addition to your day jobs. Now please go share this book you wrote with those closest to you.

No one who achieves success does so without acknowledging the help of others. The wise and the confident acknowledge this help with gratitude.

-Alfred North Whitehead

Also, a special thanks to Alok Singh and Pedro Sacramento for their insistence that the checklists at the end of most chapters be included in the first version of this book. You went above and beyond in your efforts, and we're certain that readers will find the checklists helpful and provocative.

To the book's many reviewers, including Ajay Kumar Singh, Dan Ogren, Faisal Mohamood, Gokul Ramesh, Gregg Barker, Jason Kim, Jayme Pechan, Matt Sheard, Paul Langowski, Praveen Kumar Srinivasan Rajendiran, Swamy Narayana, Timo Gossen, and Umran Hasan—you met us at the edge of exhaustion and pressed us to go on and to make a better book. Thank you.

To Denise Francis for your effective project management skills—and for keeping after us—throughout this effort. Thank you.

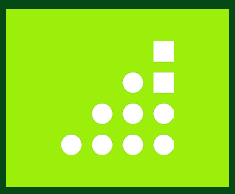
To Rishi Manocha for your critical eye and tireless assistance with all things related to the marketing of this book. Thank you.

To Carol Dillingham, Anna Tatistcheff, and the entire Steyer Content team for your highly competent and professional effort in advancing the book's content and design. Thank you.

Finally, a special thanks and debt of appreciation is due to all our current and former Dynamics 365 customers and partners from whom we learn and by whom we're inspired with each encounter. We marvel at what you do.

—Alejandro Gutierrez and Seth Kircher

Appendix



Phase Initiate

Regardless of which implementation methodology you use, here are the typical implementation deliverables, activities, and recommended roles that need to be performed by the project team during the Initiate phase of the project.

This table is for reference only; a missing deliverable, activity/task, or role might not be an indication that something is missing in your project.

Note that the names of deliverables, details regarding activities and tasks, and roles and job titles might change depending on your project.

When a deliverable or activity is application-specific, it is specified as either:

- F&O for Finance and Operations apps; or
- CE for Customer Engagements apps

Topic	Deliverables	Activities/Tasks	People
Program strategy	**Program strategy/ charter**	Define: ▪ Business vision ▪ Business goals/business drivers ▪ Business objectives ▪ Identify key success metrics or KPIs ▪ Project budget ▪ High-level project and solution scope	Customer project sponsor, program director Project managers Customer business stakeholders Customer IT director Customer IT architect Solution architect Functional consultant/ architect Change manager Project management office (PMO) team Resource manager
	Project plan	Define the project plan, scoping all activities across project phases.	
	Risk and issue register	Define the risk and issue register (with risks, severity, mitigation, consequence, issues, escalation process).	
	Project governance	Define the governance controls through establishment of a PMO and steering committee to manage the triple constraints (schedule, budget, resources), project risks, and change management/change control for the project.	
	Solution architecture management	Define architecture review board (ARB) or design authority (DA), which governs key architectural decisions made during implementations.	

Topic	Deliverables	Activities/Tasks	People
Program strategy *continued*	**Project technology and tools**	Install applications and tools that will be used to deliver the project. These could be: ▪ Lifecycle Services (for F&O projects) ▪ Azure DevOps ▪ SharePoint ▪ Microsoft Teams ▪ Conduct implementation team training on methodology and tools.	Customer project sponsor, program director Project managers Customer business stakeholders Customer IT director Customer IT architect Solution architect Functional consultant/ architect Change manager Project management office (PMO) team Resource manager
	Familiarization training	Conduct initial familiarization training on the Dynamics product(s) that will be implemented, as well as independent software vendors (ISVs), add-ons, and every other software element.	
	Train the trainer	Plan train-the-trainer strategy, which includes content preparation for training; identifying the customer trainers; number of training sessions; plan of execution of the training.	Functional consultant Technical writers Trainer SMEs Customer trainers Customer business consultant Customer IT team
	End-user training	Plan end-user training, which includes number of users, number of training sessions, and the plan for executing the training.	
	In-context help documentation	Plan in-context help document/manuals that Dynamics 365 application will provide for users.	

Topic	Deliverables	Activities/Tasks	People
Test strategy	**Test plan**	Plan the testing strategy. Define test activities such as unit testing, functional, user experience (UX), integration, system integration testing (SIT), performance, and time and motion study. Provide high-level definitions of test cases, scenarios, alternate scenarios, and expected results.	Test lead Test consultant Functional consultant Solution architect Customer IT team Customer business consultants UX consultant Performance test lead Performance test consultant
	Test execution/ metrics	Establish KPIs for measuring test execution outcomes.	
	Performance/ scalability goals	Plan performance benchmark testing to assess the scalability, concurrency loads, network configurations (latencies, bandwidths), and browser mix.	
	Usability	Elicit accessibility, compatibility, and ease-of-use requirements.	
	Testing automation tool	Confirm testing automation tool that will be used for testing, including regression testing.	
Business process strategy	**Process architecture**	Define process architecture, a commonly understood, shared view of all business processes that an organization performs to deliver a product or service to their customers.	Solution architect Functional consultant Customer business consultant(s) (SMEs) Customer IT architect

Topic	Deliverables	Activities/Tasks	People
Business process strategy *continued*	**Requirements traceability management**	Define requirements traceability matrix (RTM), a key deliverable and success measure. Every requirement needs to be linked to at least one process. It can also depict the to-be processes as requirements in the form of user stories/use cases. The RTM also includes nonfunctional and negative requirements.	Solution architect Functional consultant Customer business consultant(s) (SMEs) Customer IT architect
	Fit-gap assessment	Assess which requirement and respective process can be addressed by the system or requires some customization.	
	Gaps evaluation and estimation	Evaluate and estimate gaps with the minimum information of: ▪ Related process ▪ Related integration ▪ Criticality ▪ Complexity ▪ Dependencies	
	Organizational structure	Define organizational structure based on feedback from executives and senior managers from functional areas. Model how the business operates. It might consist of: ▪ Geographical structure ▪ Organizational groupings ▪ Line of business ▪ Reporting lines ▪ Segregation of duties	

Topic	Deliverables	Activities/Tasks	People
Business process strategy *continued*	**Country/region localization availability**	Validate country/region localization availability for each country/region in scope.	
Application strategy	**Solution architecture blueprint**	Defines high-level view of the solution architecture, which depicts how Dynamics 365 apps interact with other systems and users.	Solution architect Technical lead(s) Customer IT architect UX consultant Customer business consultant(s)
	ISV/add-ons	Define need for considering any ISV solutions to meet the business requirements. Include all considerations in the relevant strategies.	
	User experience (UX)	Prototype high-level wireframes from UX point of view to derive the new user experience.	
Data strategy	**Data migration plan and execution strategy**	Define data that needs to be migrated to Dynamics 365 apps, along with data volumes, approach, and tools to be used. Define high-level plan of data migration activities, such as data extraction, transformation, and load; migration runs; initial and incremental migrations; data validations and data verification; and production migration.	Customer IT architect Solution architect Functional consultant Data migration Lead/architect

Topic	Deliverables	Activities/Tasks	People
Data strategy *continued*	**Data management strategy**	Outline strategy, including: ▪ Scope ▪ Key requirements Approach, including: ▪ Governance ▪ Compliance ▪ Migration ▪ Master data ▪ Configuration ▪ Sharing and promotion Technology and tools	Customer IT architect Solution architect Functional consultant Data migration Lead/architect
	Configuration management (F&O)	Define plan and approach for how to manage: ▪ Configurations ▪ Shared data ▪ "Golden" configuration, including promotion ▪ "Golden" configuration environment	
Integration strategy	**System integration requirements**	Elicit the integration needs, such as: ▪ Identifying lines of business (LOBs) to be integrated. ▪ Inbound and outbound integration requirements from Dynamics 365 apps ▪ Integration types such as Sync, Async, Batch ▪ ACID (atomicity, consistency, integrity, and durability) requirements towards integration calls	Customer IT architect Solution architect Functional consultant Integration lead

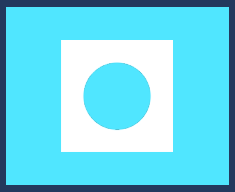
Topic	Deliverables	Activities/Tasks	People
Integration strategy *continued*	**Integration strategy**	Outline the strategy, including: ▪ Scope ▪ Integration approach ▪ Landscape ▪ Technology and tools	Customer IT architect Solution architect Functional consultant Integration lead
	Integration interfaces identification	Define the integration mapping between Dynamics 365 and other systems. Plan the dependencies.	
	Dual-write infrastructure	Consider using the dual-write infrastructure if customer has both Dynamics 365 CE apps and F&O apps.	
Intelligence strategy	**BI and reporting strategy**	Outline the strategy, including: ▪ Scope ▪ Reporting structure ▪ Key requirements ▪ Roles and responsibilities ▪ Approach ▪ Technology and tools	Customer business consultant Reporting lead Functional consultant Solution architect AI SME Customer IT architect
	Printing strategy	Outline the strategy, including: ▪ Scope ▪ Key requirements ▪ Approach ▪ Technology and tools	
	Personal, ad-hoc reports	Elicit the reports for personal or ad-hoc needs and data volumes	
	Operational reports	Elicit the reports for managers at various levels (e.g., team, business group)	

Topic	Deliverables	Activities/Tasks	People
Intelligence strategy *continued*	**Organizational and executive reporting**	Elicit the reports for broader scope, consolidated data across the enterprise systems, and KPIs for executives (CxOs).	Customer business consultant Reporting lead Functional consultant Solution architect AI SME Customer IT architect
	AI/ML needs	Elicit the requirements that are fulfilled through artificial intelligence, such as customer churn analysis, and customer lifetime value.	
	EDW (enterprise data warehouse)	Define the high-level view of Dynamics 365 data integration to customer's EDW.	
	Common data model (CDM) and Azure Data Lake Storage	Define the high-level considerations to use and structure data using CDM.	
Security strategy	**Security strategy**	Outline the strategy, including: ▪ Scope ▪ Key requirements ▪ Approach ▪ Technology and tools	Customer information security group (ISG) team Solution architect Identity SME Customer infrastructure SME Customer IT architect
	Federation, SSO	Define the needs for single sign-on and federation with other identity providers.	
	Security roles	Define all the roles required and assign these roles to business process activities so as to map them to business roles/ personas.	
	Identity integration	Elicit the requirements for integration with other identity providers.	

Topic	Deliverables	Activities/Tasks	People
Security strategy *continued*	**Azure Active Directory (AAD) access management**	Define the requirements for Active Directory or Azure Active Directory, identity integration needs, and cloud identities.	Customer information security group (ISG) team Solution architect Identity SME Customer infrastructure SME Customer IT architect
	Information security	Elicit security, privacy, and compliance requirements; regulatory needs; and audit events.	
	Mobile security	Define BYOD requirements, MDM, MAM, and/or conditional access policies.	
	Compliance management	Define the privacy, regulatory, compliance, and security procedures. Run risk assessment.	
ALM strategy	**ALM environments**	Define the required environments for ALM setup (e.g., dev, test, golden, pre-prod, prod).	DevOps consultant Customer IT architect Solution architect Technical lead Customer IT team
	Solution management	Define the solution segmentation strategy	
	Configuration management	Define the source control and branching strategy.	
	Build process	Define the processes for build management and automation.	
	Deployment process	Define the processes for deployment management and automation between the environments.	

Topic	Deliverables	Activities/Tasks	People
ALM strategy *continued*	**Release process**	Define the processes for release management of the solution.	DevOps consultant Customer IT architect Solution architect Technical lead Customer IT team
	ISV ALM	Define the process for managing the DevOps process for other components and solutions (such as Azure/non-Azure components) that make up the solution.	
Environment and capacity strategy	**Tenant, environment and geo strategy**	Define tenants/environment/ instance needs and the locations.	Customer IT architect Solution architect Technical lead(s) Customer support team Technical lead(s) PMO Customer information Security group team Customer infrastructure team
	Environment planning	Define required environments to be set up. Deploy required environments to enable the team to start working.	
	Platform release process and updates	Define the plan, maintenance schedules for the platform release and upgrades.	
	Regulation and compliance	Define the regulatory and compliance checks that must be adhered for the implementation.	
	Service-level operations	Plan the post-go-live operational procedures, such as application availability, backup, archive, audit, maintenance registers, supportability, and service updates.	

Topic	Deliverables	Activities/Tasks	People
Environment and capacity strategy *continued*	**API limits**	Define the capacity-planning requirements for integrations and user data access.	Customer IT architect Solution architect Technical lead(s) Customer support team Technical lead(s) PMO Customer information Security group team Customer infrastructure team
	Storage requirements	Define the capacity planning for storage.	
	Deployment planning	Plan the deployment strategy, locations, required hardware and software prerequisites, etc.	
	Rollout planning	Create the high-level plan for staggered rollouts.	
	Continuance plan	Create the continuous update architecture and update policy.	
	Regression automation tool	Confirm the regression automation tool that will be used.	



Phase
Implement

Regardless of which implementation methodology you use, here are the typical implementation deliverables, activities, and recommended roles that need to be performed by the project team during the Implement phase of the project.

This table is for reference only; a missing deliverable, activity/task, or role might not be an indication that something is missing in your project.

Note that the names of deliverables, details regarding activities and tasks, and roles and job titles might change depending on your project.

When a deliverable or activity is application-specific, it is specified as either:

- F&O for Finance and Operations apps; or
- CE for Customer Engagements apps

Topic	Deliverables	Activities/Tasks	People
Program strategy	**Project scope and requirements list signoff**	Following all related activities, the project scope and RTM to be signed off on.	Customer PMO Customer business stakeholders Project manager Solution architect Functional consultant/ architect
	Cutover strategy and plan	Complete this deliverable by the start of the build.	
	Solution design signoff	Once design is documented, it needs to be signed off on. All changes that are required follow review, approval, and version control.	
	Train the trainer	Prepare the training materials for customer training teams, such as functional admins, system admins, and identified functional users.	Functional consultant Tech writers
Test strategy	**Develop unit test cases**	Build unit test cases for the defined functionality to ensure TDD (test-driven development) practices are adhered to. These unit test cases should be executed as part of CI/CD pipelines.	Test lead Test consultant(s) DevOps consultant Technical consultant(s) Performance test consultant
	Build verification tests (BVT)	Conform to DevOps practices and build the BVT scripts that decide on the validity of the build for accepting to the test environment.	
	Execute test cases/test scenarios	Execute test cases manually or automatically.	

Topic	Deliverables	Activities/Tasks	People
Test strategy *continued*	**Automation of test cases**	Build automation scripts for high-impact processes. It is recommended to build automation scripts for processes that are critical and prone to changes.	Test lead Test consultant(s) DevOps consultant Technical consultant(s) Performance test consultant
	Performance testing	Develop and unit test performance test scripts and execute the scripts, per the load requirements.	
Business process strategy	**Financial structure**	Finalize financial structure design with the components: chart of accounts, dimensions, and account structure.	Solution architect Functional consultant Finance SMEs
	Test cases/ scenarios	Write test cases/scenarios supporting the functionalities to be tested for various identified test types, such as functional testing and SIT. Design the automation testing needs for performance testing, UI testing, and more.	Functional consultant Test consultants
	Business processes and personas	Define business processes (linear and cross-functional process flows) showing the inputs, outputs, and activities at a subprocess and process, including intended personas.	
	Gaps solution design	Design solution, update process flow, and confirm approach. Create development plan.	

Topic	Deliverables	Activities/Tasks	People
Business process strategy *continued*	**Low-level technical design (CE)**	For all gaps identified in the fit-gap assessment, it is recommended to have low-level technical design documentation (TDD) in place. Focus on how the requirements are achieved using the Dynamics 365 platform constructs. This must be detailed to the core. ▪ Define customizations, such as business rules, plug-ins, JavaScript, workflows, and Power Automate workflows. ▪ Depict a technical flow on how a functional requirement is fulfilled using the combination of client-side scripts and server-side rules.	Functional consultant Test consultants
Application strategy	**User experience (UX) design** **UX (e.g., forms, views) design**	Design the visualizations, such as UI form definitions, UI views, and custom controls.	Functional consultant UX consultant
	Solution design: API limits/service-protection limits planning	Consider the design and development impacts to API limits/service protection limits.	Technical lead(s) Solution architect Customer IT architect Technical consultant(s) Technical lead Functional consultant Technical consultant

continued on next page

Topic	Deliverables	Activities/Tasks	People
Application strategy *continued*	**ISV/add-ons**	Include the installation of the ISV/add-ons early in the solution design so that they are incorporated into the processes, data, security, and integration design.	Functional consultant Technical lead(s) Data migration lead Technical consultant(s) Customer IT team Customer data migration consultant(s)
	Solution design: storage capacity	Note that based on expected data migration activities and capacity planning of the solution, the design should include storage capacity needs.	
	Build solution development code	Develop the solution per the design documents.	
	Develop UX	Build UX constructs (e.g., forms, views, controls, web resources)	
Data strategy	**Data model**	Confirm data model is precise and complete. A clear definition of the following aspects of the data model should be available as part of defining the data model: ▪ Data types ▪ Data format ▪ Data length ▪ Rules (e.g., auditable, change tracking)	
	Data migration design	Often, it is possible to write custom tools or leverage ISV tools to execute data migration needs. Whatever the approach taken, it is important to design: ***continued on next page***	

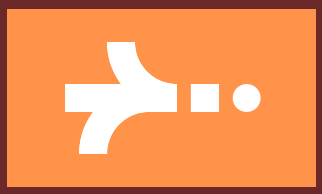
Topic	Deliverables	Activities/Tasks	People
Data strategy *continued*	**Data migration design** *(continued)*	▪ ETL (extract-transform-load) processes. ▪ Data dictionary: data migration mapping between incumbent and Dynamics 365 system. ▪ Data migration scripts design. ▪ Execution report. For projects with heavy data migration requirements, it is recommended to maintain a separate design document for Data Migration activities.	Functional consultant Technical lead(s) Data migration lead Technical consultant(s) Customer IT team Customer data migration consultant(s)
	Configuration management (F&O)	Create data packages and/or golden configuration environment.	
	Configuration management (F&O)	Store approved configuration and establish: ▪ Change management procedures. ▪ Transfer process to and from repository to the various development and test environments. ▪ Plan to move to production.	
	Historical data migration	Confirm requirements, scope, and approach. Evaluate migration in Dynamics vs. data warehouse storage.	

Topic	Deliverables	Activities/Tasks	People
Integration strategy	**Integration design**	It is recommended to maintain a separate integration design document if the integration needs are heavy. This should cover: ▪ Integration patterns to be considered ▪ Detailed integration interfaces and mapping of data attributes between the systems. ▪ Transformation rules ▪ Integration response processing ▪ Integration type (e.g., sync, async, batch). ▪ Integration technologies (e.g., logic apps, Power Automate, third-party ESBs)	Integration lead Technical consultants Azure architect/SMEs (if required) Solution architect
	Dual-write infrastructure	If dual-write is used, harmonize concepts such as company, business unit, security, etc., between CE and F&O. Determine the plan to architect the data properly between apps.	
Intelligence strategy	**BI design**	If the project has a high number of reporting requirements and the project needs custom reporting beyond what Dynamics 365 offers out of the box, it is recommended to build a detailed design document covering the following:	BI lead Technical consultants Functional consultant

Topic	Deliverables	Activities/Tasks	People
Intelligence strategy *continued*		▪ Sources for data beyond Dynamics 365 apps. ▪ Visualizations using reporting tools such as Power BI or SSRS. ▪ Reporting queries. ▪ Logic to make up the reporting interfaces.	
Security strategy	**Security design**	Security implementation is one of the critical pillars in Dynamics 365 implementations. Design should be thorough and requires extensive analysis of the current requirements before deciding on the security constructs suiting the Dynamics 365 apps. Example: For Dynamics 365 CE apps or Power Platform, design of the following is of utmost importance: ▪ Business units ▪ Security roles and access levels ▪ Ownership, access, and AAD teams ▪ Field-level security ▪ Custom security options, if any	Technical lead Solution architect Functional consultant Customer IT architect Customer information security group (ISG)

Topic	Deliverables	Activities/Tasks	People
Security strategy *continued*	**Security roles (F&O)**	When a custom role needs to be created, a good practice is to not modify the standard roles because it will impact the continuous updates from Microsoft. Apply strict change management control. Include security roles in the golden configuration.	Technical lead Solution architect Functional consultant Customer IT architect Customer information security group (ISG)
	Segregation of duties (SoD)	Once the security roles are assigned to process activities, identify SoD rules and baseline the security testing.	
	Compliance management	Track, assign, and verify your organization's regulatory compliance activities related to Microsoft Professional Services and Microsoft cloud services, such as Office 365, Dynamics 365, and Azure.	
ALM strategy	**Develop CI/CD pipelines**	For continuous integration and continuous development needs, it is recommended to build the routines for DevOps.	DevOps consultant Technical consultant(s)
Environment and capacity strategy	**Solution design: performance**	Design considerations for high-level customizations from the viewpoint of system performance, such as UI form loads, error handling, query performance, solution health checks, and more.	Technical lead(s) Technical consultant(s)

Topic	Deliverables	Activities/Tasks	People
Environment and capacity strategy *continued*	**Regression automation test**	Build regression automation tool and test scripts to be able to frequently uptake Microsoft service releases.	Technical lead(s) Technical consultant(s)



Phase
Prepare

Regardless of which implementation methodology you use, here are the typical implementation deliverables, activities, and recommended roles that need to be performed by the project team during the Prepare phase of the project.

This table is for reference only; a missing deliverable, activity/task, or role might not be an indication that something is missing in your project.

Note that the names of deliverables, details regarding activities and tasks, and roles and job titles might change depending on your project.

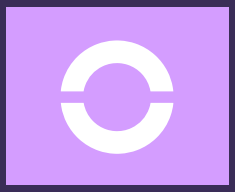
When a deliverable or activity is application-specific, it is specified as either:

- F&O for Finance and Operations apps; or
- CE for Customer Engagements apps

Topic	Deliverables	Activities/Tasks	People
Program strategy	**Signoff documents**	The customer/partner has gone through all internal approvals, information security reviews, penetration testing, and other steps to ensure production readiness of the system.	Customer information security group/team Customer IT architect Solution architect Functional consultant PMO
	Go-live readiness document/ cutover checklists	The cutover plan, go-live date, and go/no-go criteria are agreed upon with customer business stakeholders and the production deployment runbook is created with all tasks, owners, start time, duration, and rollback plan determined.	Customer project sponsor(s)
	Train the trainer	Complete training of the customer trainers for functional and administration activities.	Functional consultant Technical writers Customer training admins Customer business users
	End-user training manuals	Prepare and deliver the end-user training for effective use of Dynamics 365 Business Apps.	
	Online help guides	Revise online help guides based on feedback from UAT, if any.	
Testing strategy	**UAT test cases**	Refine UAT test cases used for validating the solution.	Customer business users (UAT test users) Test lead Functional consultant Test consultant(s)
	UAT test report	Prepare and present UAT test execution report that baselines the UAT signoffs.	

Topic	Deliverables	Activities/Tasks	People
Testing strategy *continued*	**Performance benchmark testing results**	These results help establish the confidence to go live for the anticipated concurrent loads, and transactions.	Performance test lead Solution architect Customer IT architect
	UAT signoff	Once the tests are completed, business signs off on user acceptance testing.	Program director Test lead
Business processes strategy	**Mock cutover approval**	Conduct mock cutover and validate and approve outputs. Sign off.	Tech lead Technical consultant(s) Solution architect Customer IT architect
Data strategy	**Data migration execution results/ signoff**	Team completes the last cutover migration of the data required following the shutdown of the incumbent systems before opening Dynamics 365 for go-live.	Tech lead Technical consultant(s) Solution architect Customer IT architect
ALM strategy	**Escrow**	Escrow all the project artifacts collected during the Initiate, Implement, and Prepare phases.	Tech lead Technical consultant(s) Solution architect Customer IT architect
	Knowledge transition documents	Register the documents and maintain all pre- and post-deployment activities that the support team should consider.	
	Maintenance register/ maintainability workshops	Maintainability workshops are set of trainings/documents that help the support team learn the operational procedures of the production solution.	

Topic	Deliverables	Activities/Tasks	People
Environment and capacity strategy	**Service-level agreement (SLA)**	Confirm and understand the SLA.	Tech lead Technical consultant(s) Solution architect Customer IT architect
	Production environment	Release and deploy production.	



Phase
Operate

Regardless of which implementation methodology you use, here are the typical implementation deliverables, activities, and recommended roles that need to be performed by the project team during the Operate phase of the project.

This table is for reference only; a missing deliverable, activity/task, or role might not be an indication that something is missing in your project.

Note that the names of deliverables, details regarding activities or tasks, and the roles or role titles in the People column might change depending on your project.

When a deliverable or activity is application-specific, it is specified as either:

- F&O for Finance and Operations apps; or
- CE for Customer Engagements apps

Topic	Deliverables	Activities/Tasks	People
Project postmortem		Post-go-live review	Customer support team Customer IT architect PMO Solution architect Functional consultant
Production operations (maintenance)	**Operations guides/checklists**	Refine the operations guides/ checklists that help maintain the production solution for updates, changes, and solution health.	
Platform release process and updates	**Operations guides/checklists**	Update the cloud solution. Test customizations supporting the new updates.	Customer Support Team Customer IT architect PMO Solution architect Functional consultant Technical consultant(s) Test consultant(s)
Maintenance	**Operations guides/checklists**	This is a very important practice to ensure necessary maintenance schedules are defined to undertake activities such as: ▪ Platform updates. ▪ Custom solution updates. ▪ Batch data updates. ▪ Archival activities. Periodical execution of archival and data retention.	Customer support team Customer IT architect PMO

Topic	Deliverables	Activities/Tasks	People
Change management	**Change request** **Change impact assessment** **Change control**	Documents that explain the change request (e.g., hotfix, feature enhancement) to be brought into the production solution and the impact assessment of the existing components, as well as the plan to implement and deploy the change.	Customer support team Customer IT architect PMO
User adoption/ usage	**User adoption document (user interviews, feedback loops)**	Document the solution usage patterns and user interviews, along with strategies to improve the adoption of the system.	Customer business users Customer IT architect PMO