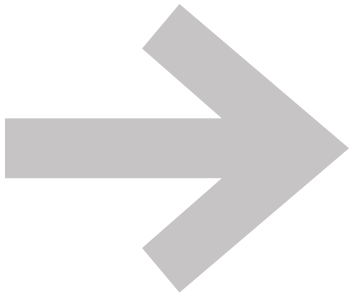


Understanding the Impact of Low-Code Development with Power Apps and Power Automate

A Guide for IT Infrastructure Leaders





Adopting Power Apps and Power Automate can bring many benefits. But realizing those benefits requires IT infrastructure leaders to effectively run, manage, and support these low-code applications. This guide explains the fundamentals of what you need to do help your organization succeed with these technologies.

01 /

Understanding low-code development

- 05 How low-code development benefits your organization
- 08 A brief introduction to Power Apps and Power Automate

02 /

Succeeding with low-code development: The questions IT leaders must answer

- 14 How can IT monitor and control low-code applications?
- 18 How can IT control access to data?
- 21 How can IT do change management?
- 22 How should IT provide support?

03 /

What to do now

04 /

For further reading

Understanding low-code development

Can your IT group create new applications fast enough to meet demand? For most organizations today, the answer is no. In an era of digital transformation, a world where technology does more and more, the demand for new applications outstrips what professional software developers can supply.

Low-code development lets nonprofessional developers create powerful applications.

An effective way to address this challenge is to let people working in business units who aren't professional developers build applications on their own. These citizen developers can rely on easy-to-use tools for creating powerful applications. This low-code approach can help your organization create the software it needs in less time and for less money.

Yet as an IT infrastructure leader, you know the risks of what's known as rogue or shadow IT. Letting business people create applications on their own, building software that IT has no access to or control over, can be a perilous path. You also know that when things go wrong, the blame is likely to fall on you. What you'd like is a way to get the best of both worlds: citizen developers in the business building applications largely on their own, with central IT providing the necessary guardrails.

Microsoft provides Power Apps and Power Automate to enable low-code development.

To make this possible, Microsoft provides Power Apps and Power Automate, two related technologies that support low-code development. Both are cloud services—they run on Microsoft Azure—and both are available with Microsoft Office 365, Microsoft Dynamics 365, and on their own. Microsoft also provides the tools you and your team need to manage this low-code environment.

Embracing low-code development can provide real benefits for your organization. Yet for IT leaders, this approach raises some obvious concerns, including these:

- How can you manage an organization where IT isn't creating all applications? It's entirely possible that your firm is already running solutions built with Power Apps and Power Automate, even if you don't know about them. This isn't necessarily a bad thing—the technologies are designed to allow this—but you're likely to want your IT organization to have some say in how these applications are managed. How can you do this?
- How can you ensure that your organization's data remains secure? What stops an employee from building an app that, say, posts all your customer data through Twitter?
- How can you support this potentially decentralized approach to building new software? Once people start creating these low-code applications, they're bound to ask for help. How should you provide this?

As an IT leader, you need to answer these questions. This guide addresses all of them and reading it will help you make the right decisions for your organization.

How low-code development benefits your organization

Low-code applications can be used in several different scenarios.

Using Power Apps and Power Automate lets your organization more easily create the applications it needs, helping to eliminate your application backlog. These technologies can also help ensure that scarce IT resources are focused on projects that genuinely require professional developers, letting citizen developers create whatever applications are suitable for low-code development. The two groups can also work together, with pro developers building on work done by citizen developers and vice versa.

Low-code applications can be used in a variety of different scenarios, including the following:

- **Building departmental applications:** Common examples here include applications that help with data capture, provide custom user interfaces over existing data, and automate internal business processes that might otherwise be done with email or even pen and paper. While software created with Power Apps and Power Automate certainly can be tier-one, mission-critical applications, a majority of low-code development today starts with less critical solutions.
- **Extending Office 365 solutions:** One example of this is replacing the older Power Apps Workflow Designer and InfoPath with the modern cloud-based solution provided by Power Apps and Power Automate.
- **Extending Dynamics 365 applications:** As your organization adopts Microsoft Software as a Service (SaaS) solutions, Power Apps and Power Automate are how you'll tailor and extend those apps to meet your specific needs.
- **Extending legacy applications:** For example, suppose you'd like to add functionality to an existing Oracle-based custom application. Rather than modifying the legacy system directly, you could instead use Power Apps and Power Automate to build new functionality around what already exists, such as adding mobile access.

Adopting low-code development enables bottom-up innovation.

Because business people can now build apps on their own, your organization can benefit from bottom-up innovation. Rather than jumping through whatever hoops are required to get approval and funding for traditional application development, Power Apps and Power Automate let creative people in any part of your company experiment with creating new applications.

Yet as described in more detail later, these applications always run within the constraints defined by IT; you're still in control.

The truth is this: your users want the ability to create their own applications—the adoption of low-code development is an industry wide trend—and if you don't provide a supported way to do it, they might well buy a low-code solution on their own. But since Power Apps and Power Automate are available by default to almost everybody on an Office 365 or Dynamics 365 plan, wouldn't you rather have them use the low-code platform you already have, running on a cloud you already trust? Your responsibility is to help make them successful in doing this.

For more on business benefits, read [Forrester's report on their economic impact](#)

In fact, many organizations around the world are using Power Apps and Power Automate today to better meet their goals. Here are some examples:

- SNCF, the French national railway, has used these technologies to create automated solutions for guiding train repair, reporting incidents, booking vehicles, and more.
- The American Red Cross created an app to automate the process of ordering supplies for training classes. They've also used Power Apps and Power Automate for other solutions, such as tracking assets and managing volunteers during disasters.
- Standard Bank has created apps for things such as helping ATM inspectors record faults, onboarding new customers, and more

Power Apps and Power Automate are mature technologies, used successfully by thousands of organizations today.

A brief introduction to Power Apps and Power Automate

Low-code development is just a simpler way to create applications. In fact, low-code apps built with Power Apps and Power Automate are structured much like traditional applications. Figure 1 shows their main components.

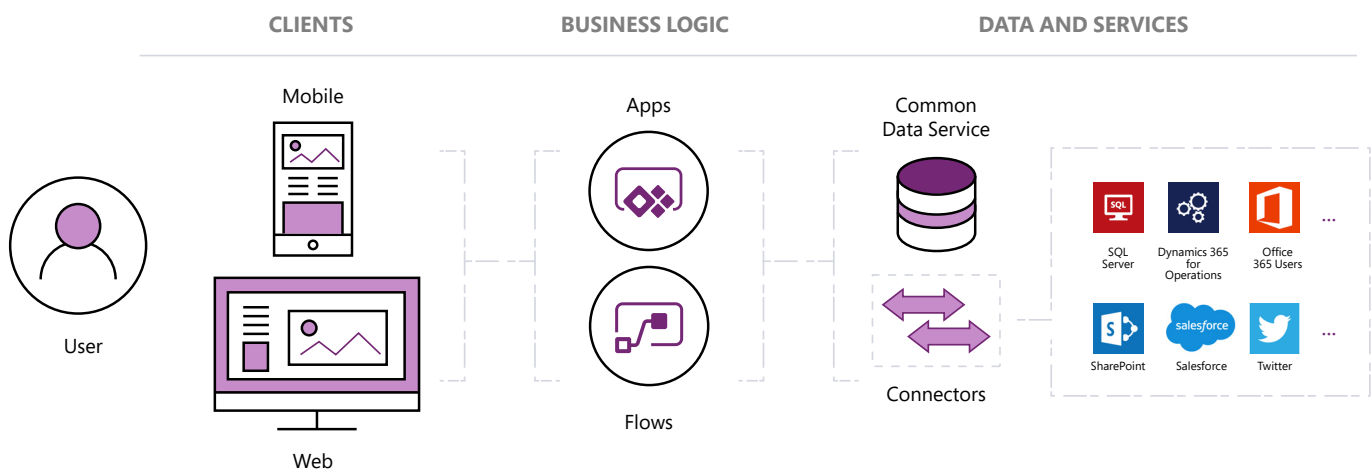


Figure 1: Low-code applications replace traditional development technologies with simpler alternatives.

Low-code applications have the same basic structure as traditional applications.

As the figure shows, low-code apps can be thought of in three distinct parts—clients, business logic, and data—just like traditional applications. The simplest way to understand the low-code way is to look at each of these parts.

- **Clients:** Power Apps and Power Automate support both web browser clients and mobile clients running on iOS and Android. For example, Power Apps provides a standard mobile client for low-code apps. An app's creator can customize this client's user interface with a simple PowerPoint-like graphical tool.

- **Business logic:** Unlike a traditional application, where logic is created by writing code in C#, Java, or some other programming language, a low-code app can be built without writing code. In Power Apps, logic is created by writing Excel-like formulas, while Power Automate allows graphically specifying processes. It's also possible, when necessary, to call out to code written in traditional programming languages. This makes it easier to extend low-code apps with logic created in other ways. And because both Power Apps and Power Automate run on Microsoft Azure, apps and flows benefit from the security, reliability, and certifications this cloud platform provides.
- **Data and services:** Traditional applications commonly rely on some kind of database management system (DBMS). Low-code apps built with Power Apps and Power Automate can instead use a built-in data management system called the Common Data Service. Along with storing data, this service also lets people define business rules and validations on that data. Another option for low-code applications is to rely on a large number of connectors that make it easy to access other data sources such as SQL Server, Oracle, SharePoint, Dynamics 365, Salesforce, Dropbox, and many others. Some connectors also allow access to functionality provided by cloud services, such as the ability to send tweets on Twitter. And although it's not shown in Figure 1, Power Apps and Power Automate also provide an on-premises data gateway to let your cloud-based low-code applications access on-premises data.

Power Apps and Power Automate make creating logic easier.

While all three parts of an application are important, the heart of what makes low-code development different is how logic is created. To get a more concrete sense of how Power Apps and Power Automate simplify this, it's useful to look at a simple example. Figure 2 shows a screenshot of a flow created with Power Automate.

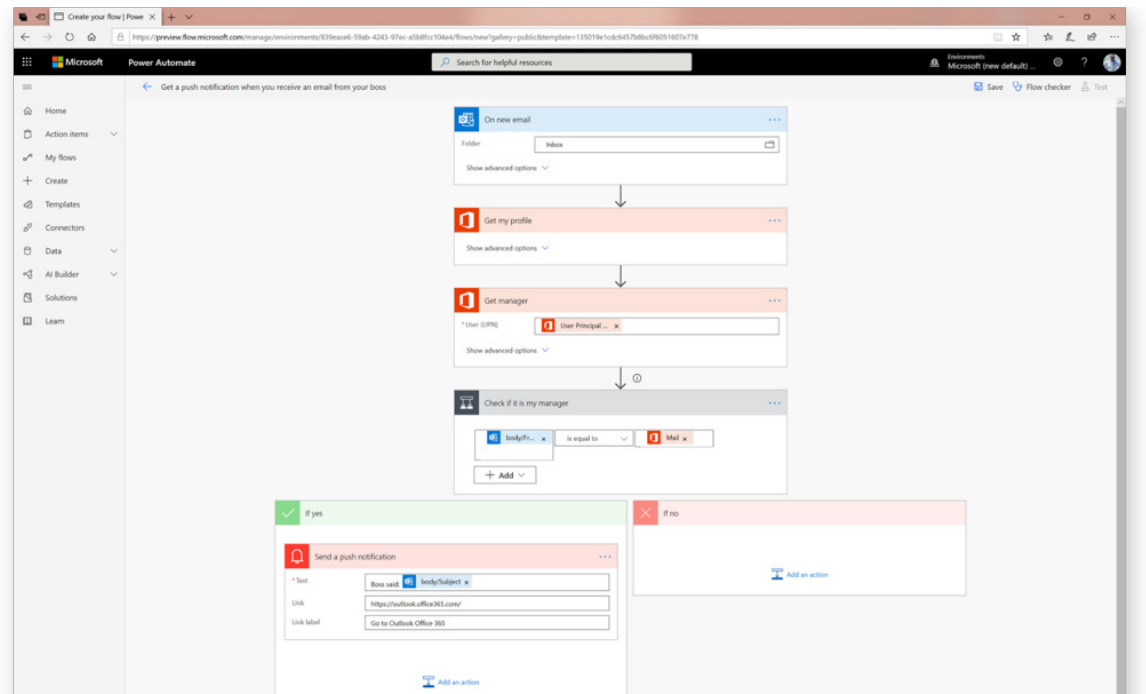


Figure 2: Flows are created graphically, providing a simple way to automate business processes.

Creating low-code applications requires much less technical knowledge.

As this example shows, Power Automate provides a tool for graphically defining processes. Rather than using a more complex professional developer tool, such as Visual Studio, this low-code development environment runs in a web browser.

Creating traditional applications requires a substantial amount of specialized knowledge, which is why they're built by professional developers. Creating low-code applications requires understanding the business problems to be solved, but there's much less need for detailed technical knowledge. Instead, they're designed to be created by less technical business people, i.e., by citizen developers. Because Power Apps and Power Automate provide extensibility, however, extending an app with code created by a professional developer in .NET or another language is straightforward.

Can professional developers use low-code tools?

Absolutely. Power Apps and Power Automate can help pro developers create applications faster than using traditional technologies such as C#. Any IT organization that wants to improve how it builds software should consider these low-code options. Many organizations are doing this today, including Microsoft: our own internal IT groups now builds a large share of its applications using Power Apps and Power Automate.

Adopting low-code technologies in a professional dev environment is much like adopting any other new development technology; it requires training, support, and more. This all happens within the IT department, however, so it's outside the scope of this guide. Still, it's important to realize that low-code solutions can bring real value to application development within IT as well as in other parts of your organization.

Succeeding with low-code development: The questions IT leaders must answer

Business people create low-code applications; IT monitors, controls, and supports those applications.

For your organization to get the most value from Power Apps and Power Automate, two big things need to happen:

1. Citizen developers—people outside of IT—need to create, use and manage low-code applications that improve how the organization works. They also need to understand the data privacy guidelines of your organization and perhaps learn how to work with apps throughout their lifecycle.
2. IT needs to monitor, control, and support these applications and the infrastructure they depend on. This is how your organization provides the guardrails required to make citizen development successful.

Figure 3 illustrates this idea.

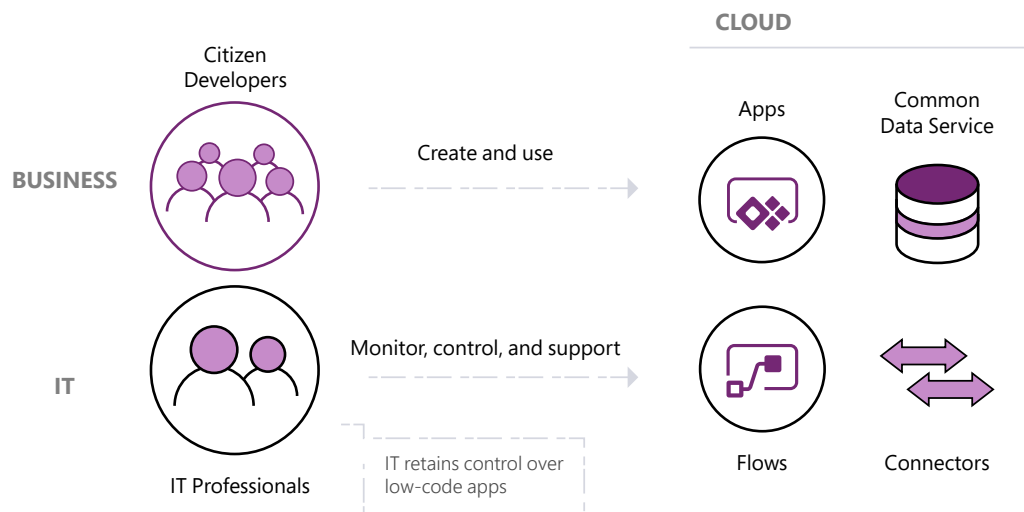


Figure 3: While citizen developers create and use low-code solutions, IT professionals monitor, control, and support these solutions.

Different organizations address these needs in different ways. Still, to succeed with low-code development, IT leaders must answer four main questions:

1. How can IT monitor and control low-code applications?
2. How can IT control access to data?
3. How can IT provide change management?
4. How should IT provide support?

What follows looks at each of these questions.

How can IT monitor and control low-code applications?

While citizen developers can create Power Apps and Power Automate solutions on their own, IT has total control over these apps and flows. You can monitor them, control what types of data they can access, and even shut them down if you wish. This combination of freedom and control relies on an abstraction called an environment. Each environment can run one or more apps/flows, and each environment is logically isolated from all others. Figure 4 shows how this looks.

All apps and flows run in specific environments.

CLOUD TENANT:

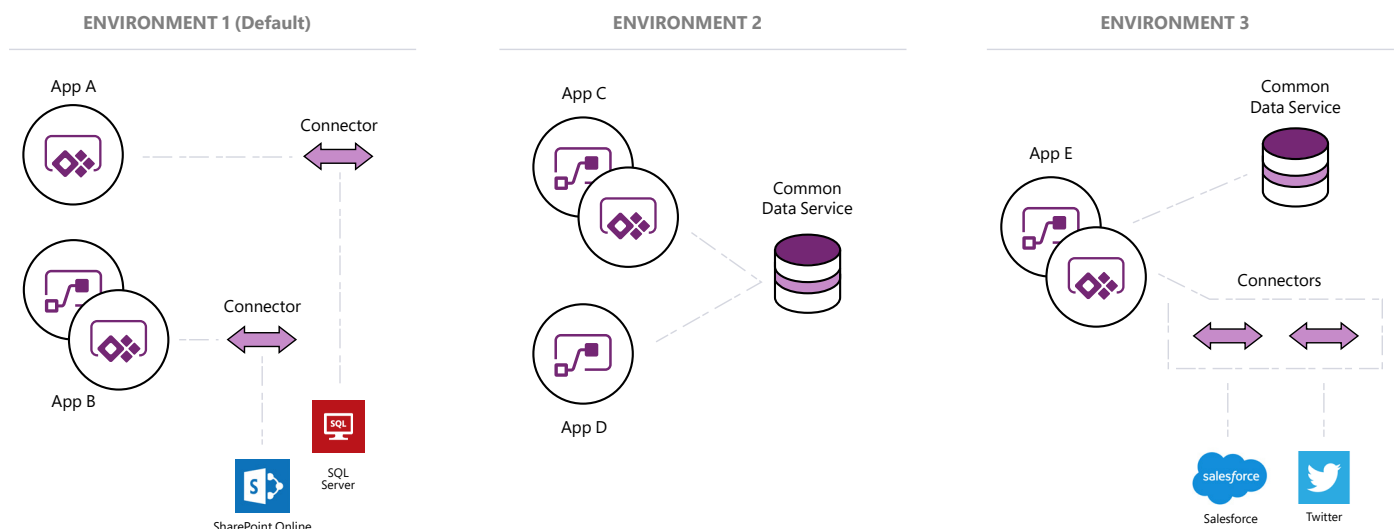


Figure 4: Each cloud tenant contains one or more environments, and each environment can contain one or more apps/flows.

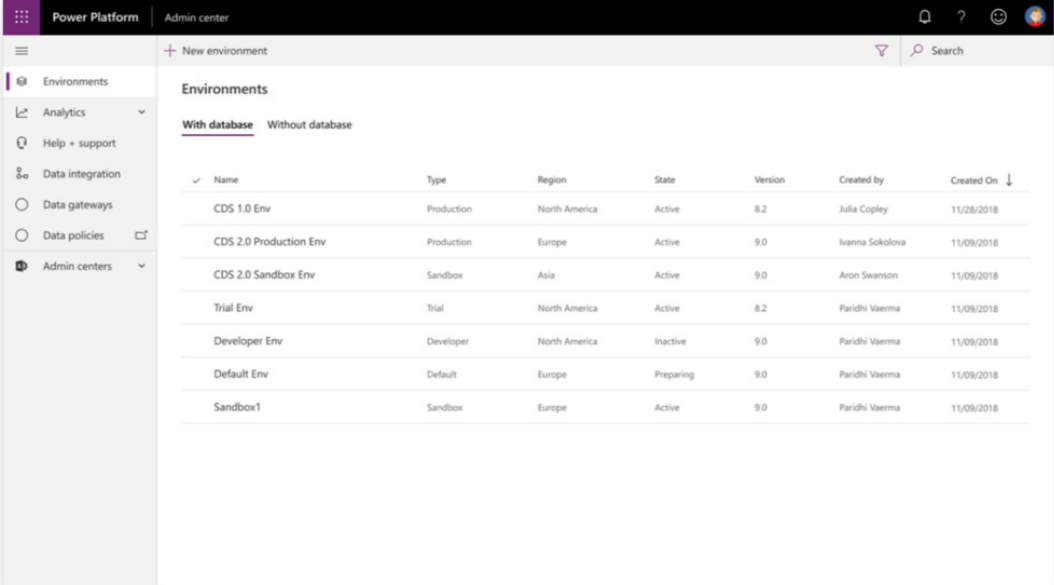
As the figure shows, environments exist inside your cloud tenant. This example shows three environments, each running different things:

- *Environment 1* runs App A, built solely with Power Apps, which accesses SQL Server through the SQL Server connector. This environment also runs App B, built using both Power Apps and Power Automate, which accesses SharePoint Online through the SharePoint connector.
- *Environment 2* also runs two apps, both of which access data in the Common Data Service. App C is built using Power Apps and Power Automate, while App D uses only Power Automate.
- *Environment 3* runs just App E, built using Power Apps and Power Automate. This app works with data from several sources, including the Common Data Service, Salesforce, and Twitter. These last two data sources are both accessed via connectors.

The default environment is always available; it can't be shut down.

Note that Environment 1 is marked as the default environment. This environment is created when your organization starts using Office 365 or another plan that includes Power Apps and Power Automate. All the other environments must be explicitly created by your organization. Different organizations do this in different ways, and you need to determine what approach is best for you. It's important to note, however, that you can't shut down the default environment—it's always available to citizen developers. IT can shut down any apps and flows running in this environment, but it can't shut down the environment itself.

So how does IT maintain control? The answer is that you can monitor what's happening in every environment, including the default environment, using the Power Platform administrative tools. You can then take whatever action you think is appropriate, including shutting down any application created with Power Apps and Power Automate. For example, Figure 5 shows how you can see a list of the environments running in your organization.



The screenshot shows the Power Platform Admin center interface. On the left is a navigation pane with options: Environments, Analytics, Help + support, Data integration, Data gateways, Data policies, and Admin centers. The main area is titled 'Environments' and has a '+ New environment' button. Below the title are two tabs: 'With database' (selected) and 'Without database'. A table lists the environments with columns: Name, Type, Region, State, Version, Created by, and Created On. The table contains eight rows of environment data.

Name	Type	Region	State	Version	Created by	Created On
CDS 1.0 Env	Production	North America	Active	8.2	Julia Copley	11/28/2018
CDS 2.0 Production Env	Production	Europe	Active	9.0	Ivanna Sokolova	11/09/2018
CDS 2.0 Sandbox Env	Sandbox	Asia	Active	9.0	Aron Swanson	11/09/2018
Trial Env	Trial	North America	Active	8.2	Paridhi Vaerma	11/09/2018
Developer Env	Developer	North America	Inactive	9.0	Paridhi Vaerma	11/09/2018
Default Env	Default	Europe	Preparing	9.0	Paridhi Vaerma	11/09/2018
Sandbox1	Sandbox	Europe	Active	9.0	Paridhi Vaerma	11/09/2018

Figure 5: The Power Platform Admin center lets you see what environments are running, along with information about each one.

This screen shows the basics for each environment, such as whether it's the default environment, an environment used for production, an environment used for testing (this is what "sandbox" means) or something else. It also shows who created each one, when it was created, and more. From this screen, you can get to more detail about any of these environments.

For example, Figure 6 shows more information about this tenant's sandbox environment.

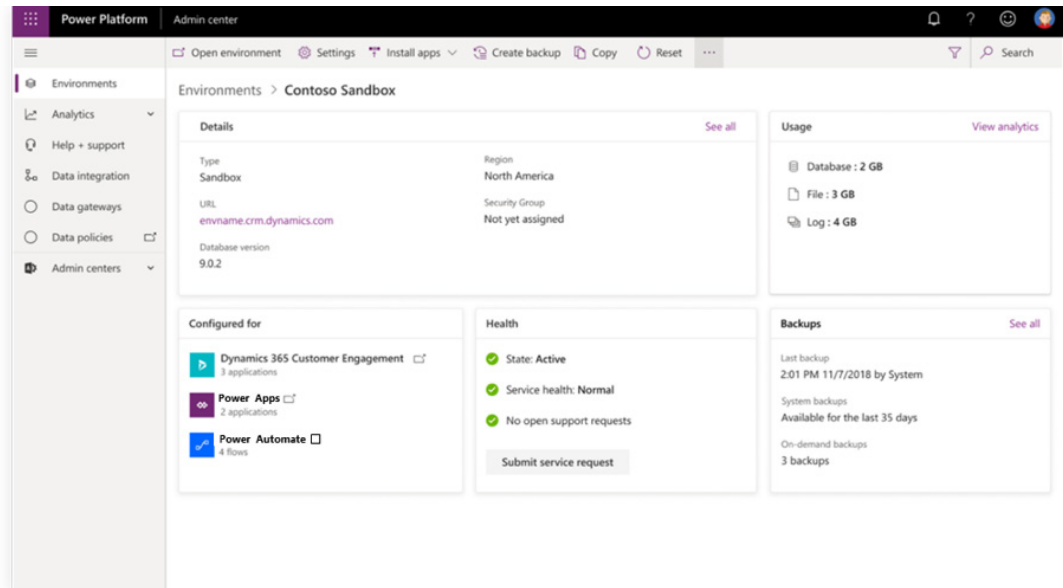


Figure 6: The Power Platform Admin center also lets you get more detail about an environment, as well as control the apps and flows running inside it.

As this screen shows, you can see exactly what apps and flows are running in this environment. If necessary, you can examine each of these in more detail, then take whatever action is necessary.

You can also create custom management tools using Power Apps, Power Automate, and PowerShell.

Power Platform Admin center and other administrative tools allow much more than this. For example, you can see how many users are using each app/flow, how often they use it, and where this usage is happening. You can also see failures in flows, then drill into them to find specific errors.

If you need more than what these tools provide, you're free to create your own custom management tools. These tools can be created using apps and flows, or you can build them in PowerShell. Microsoft provides connectors that let you interact with the management APIs exposed by Power Apps and Power Automate,

giving you the ability to do many different things. For instance, you can create a custom dashboard just for your organization that shows things such as which apps are new, which have been shared with more than twenty people, which connectors each app uses, and more. Your custom dashboard might also provide notifications, such as sending you an email or text message when an app gets shared with more than twenty people.

Power Apps and Power Automate bring a new organizational model of how applications are created. Rather than waiting for their central IT group, business people can now create their own applications. Yet IT still retains full control over the execution of those applications. This change can bring substantial business value, which is why it's so important that you and your IT organization understand the part you play in making it possible.

How can IT control access to data?

Power Apps and Power Automate let users create new applications on their own. And while IT has complete control over those applications, there's another critical question to ask: What about the data? How can IT prevent citizen developers from using your organization's data in destructive, non-compliant, or even illegal ways?

Using Power Apps and Power Automate doesn't elevate a user's permissions in any way

The first thing to understand is that using Power Apps and Power Automate doesn't change a user's data access permissions in any way. If a citizen developer creates a flow to make some task more efficient, that flow can access only the data its user already has permission to access. If an app uses the SharePoint connector to work with SharePoint Online data, for example, a user of the app will be allowed access only to the SharePoint lists and other data she already has the right to access. Using this connector doesn't mean that she can now access any SharePoint list she wants. The foundation of access control remains the permissions your organization already has in place; Power Apps and Power Automate don't change this.

You can also control which connectors can be used together in the same app or flow. You do this by creating one or more data loss prevention (DLP) policies. Each DLP policy specifies (and enforces) rules about which connectors can be used in the same app or flow. To do this, each DLP policy classifies connectors into two disjoint groups: *Business data only* or *No business data allowed*.

Figure 7 illustrates this idea.

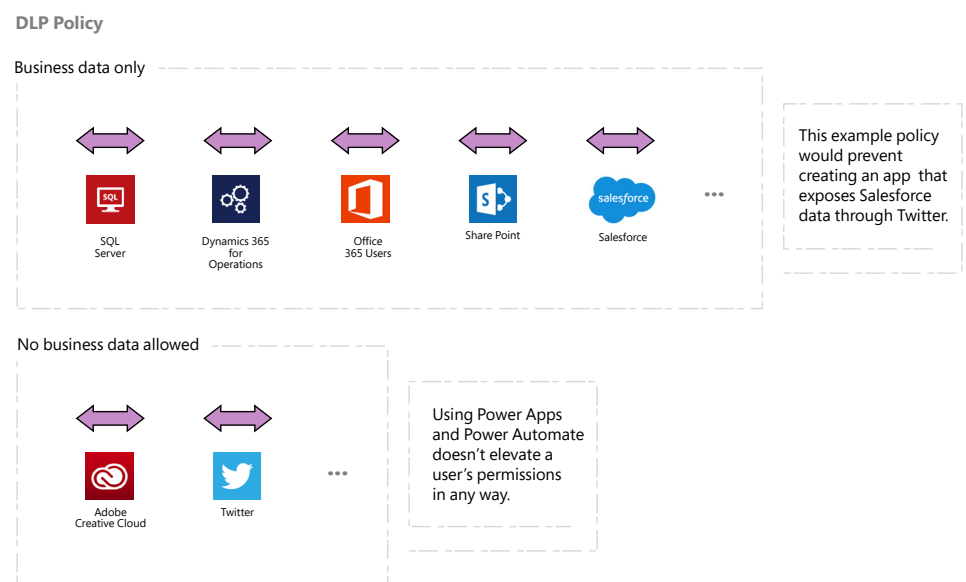


Figure 7: A DLP policy controls which connectors can be used together in a single app or flow.

A DLP policy prevents connectors in different categories from being used together.

Each DLP policy enforces a simple rule: a specific app or flow can use connectors only from one of these two groups. With the DLP policy shown in Figure 7, for example, a particular app/flow could use any of the connectors in the top row, those in the Business data only category, but it would then be prohibited from using any in the No business data allowed category. Another app or flow subject to this same DLP policy could use any of the connectors in the bottom row, the ones in the No business data allowed category, but it couldn't use any of the Business data only connectors.

To see why this is useful, suppose somebody in your organization tries to build an app that reads your CRM data from Salesforce, then tweets each customer's information to the world. With the DLP policy shown here, creating this app wouldn't be allowed—adding both the Salesforce connector and the Twitter connector would raise an error.

By default, all connectors are considered part of the No business data allowed category. This means that every app and flow can use any of these connectors together (something you probably want to change).

DLP policies can be set for all environments or for a specific environment. How you define these policies depends on how your organization uses Power Apps and Power Automate. For example, a tenant-level setting might have a very restrictive DLP policy that applies to all environments except one: an enterprise-wide production environment in which applications must be approved by IT before they run. This would let citizen developers build and run apps as they saw fit in their own environments, with no IT involvement, as long as these developers used connectors in a safe way. If a citizen developer wanted to do something risky, however, such as use the CRM connector and the Twitter connector in a single app, this app would have to run in the enterprise-wide production environment, which means the app's creator would need to get IT's approval to run it.

Power Apps and Power Automate provide a range of options for managing change.

How can IT do change management?

Managing change is a fundamental part of any application environment. This includes low-code platforms such as Power Apps and Power Automate. Accordingly, these technologies provide a broad set of options for change management.

An important part of this is making sure that only authorized people make authorized changes. Doing this starts with authenticating users, i.e., making sure that everyone is who they say they are. Since Power Apps and Power Automate run on the Microsoft Azure platform, the solution for doing this is obvious: every user of these technologies, including both citizen developers and people who use the apps these developers create, is authenticated using Azure Active Directory.

Once this happens, access to resources depends on other things. For example, as just described, data access relies on the authorization mechanisms built into each data source; Power Apps and Power Automate don't change this.

But how can you control who is allowed to create apps and flows, or who's allowed to create new environments? The answer is that this low-code environment provides a range of different roles. There's a role that allows creating new apps and flows, for example, and while everybody belongs to this role for the default environment, this need not be true for other environments. There are other roles that grant various levels of administrative control over the environments in your tenant. Assigning these administrative roles to individuals can be done by business units or by IT, giving you the flexibility your organization needs.

Effective change management requires addressing other questions as well, including these:

- Since good development practice calls for separate domains for development, testing, and production, how can Power Apps and Power Automate provide these? The answer is straightforward: the people creating apps and flows can use different environments for development, testing, and production, just as in traditional software development.
- What happens when someone creates an app or a flow, then leaves the company? The answer is that someone else can be assigned as the new owner of this app or flow. Because all identities on this platform are managed by Azure Active Directory, making this change is straightforward. An organization could even create a flow that recognizes when an app owner leaves the company, then sends email to an administrator requesting that a new owner be assigned.
- What happens when a new version of an app or a flow is deployed? The solution here is simple: each time someone deploys a new version, it replaces the existing version. Older versions are retained, however, making it easy to roll back to the previous release if necessary.

How should IT provide support?

As citizen developers start to use low-code development, they'll expect support from IT.

Whether you're aware of it or not, your organization probably already has at least a default environment running for Power Apps and Power Automate. Your business users can easily find these technologies—both appear in the Office 365 waffle, for example—and so citizen developers are bound to start using them. And once this happens, those people will expect support from IT.

The level of support you choose to provide depends on how your organization decides to use Power Apps and Power Automate. There's no single approach that's right for everybody. One way to think about it is to view your options along a continuum like the one Figure 8 shows.

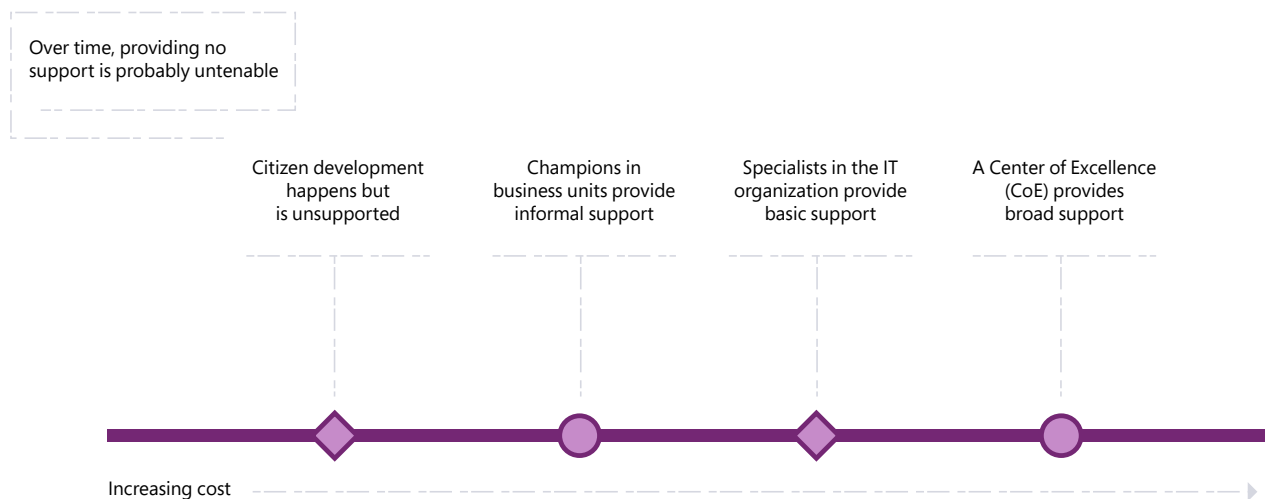


Figure 8: There's a continuum of options for supporting low-code development in your organization, with the cost increasing as you provide more.

It's worth taking a closer look at each of the four options shown in the figure:

Providing no support is possible, but it's probably not a good long-term option.

1. **Citizen development happens but is unsupported:**

This approach is, at least initially, the simplest and the cheapest. People in your organization will still create apps and flows, but when these citizen developers call IT support for help, they'll be told that Power Apps and Power Automate are unsupported. This approach lets you avoid the cost of training support staff, and it also lets you avoid using your support staff's time helping with low-code development. But it's also likely to make IT look bad. After all, since your users have access to Power Apps and Power Automate, they're bound to assume that IT supports these technologies. And if the low-code approach spreads throughout your organization, providing no support at all is likely to become untenable over time.

2. Champions in business units provide informal support:

If the IT organization doesn't offer support for low-code development, it's likely that a few successful citizen developers will emerge within the business. They'll then act as technology champions, offering informal support to less experienced users of Power Apps and Power Automate. IT might even help these champions get the training they need to do this, such as an understanding of data privacy and application management. This approach can work, but it's not stable. What happens when a champion's manager decides that he's spending too much time supporting others, something that's probably not an official part of his job? Or what if the champion changes jobs or leaves the company, and this support disappears with no warning? In the absence of formal support from IT, business unit champions are better than nothing. But relying on enthusiastic technology fans isn't typically the best long-term approach.

3. Specialists in the IT organization provide basic support:

A third approach is for IT to invest in providing some support for citizen development. This support might focus on infrastructure issues, such as creating environments, establishing DLP policies, and assigning security roles. Citizen developers can do all these things by themselves—IT doesn't have to step in—but especially as the use of Power Apps and Power Automate grows in your organization, you're likely to be more successful if trained, specialized people take on these responsibilities. To do this, you might define policies, such as requiring that any low-code application with more than 20 users must work with central IT. You might even create a flow that looks for applications with this number of users, then notifies their owners of this policy. Perhaps the policy also requires that people who create these more popular apps go through approved training. There are many other issues that might also make an app need approval, such as personal data usage, employee monitoring, and more.

A Center of Excellence provides the most complete support for low-code development.

4. A Center of Excellence (CoE) provides broad support:

Since Power Apps and Power Automate can bring substantial value to an organization, many organizations make a substantial investment in supporting these technologies. Providing a CoE with trained low-code architects and developers, along with templates and samples for citizen developers, can significantly increase an organization's ability to create high-quality applications. (How many citizen developers are up to speed on effective testing practices, for example?) A CoE might also impose more rigorous policies, such as defining standards that low-code applications must meet. For example, perhaps any app or flow with more than 50 users or that works with personal data or meets some other criteria must run in a production environment controlled by IT and so must be reviewed and approved by IT. The goal is to make citizen developers as independent as possible while still making sure that any widely used applications they create are of high quality.

Some organizations start from the right side of the continuum in Figure 8, deciding up front that low-code development will bring them substantial benefits. They then decide to invest in a CoE from the beginning, an approach that requires clear support from upper-level management. Others start from the left side, building their support capabilities as needed. If their organization fully embraces low-code development, they'll eventually wind up on the right side of the line, with a fully staffed CoE, but starting here certainly isn't required. The right answer for you depends on your organization's approach to adopting low-code development.

One more challenge that organizations face in adopting low-code development is avoiding duplication of effort. Especially in big companies, people in different business units often want similar applications. The decentralized development that Power Apps and Power Automate provide make it easier

for each group to build its own version of an application, which is probably a waste of resources. To avoid this, some organizations have created a shared catalog of Power Apps and Power Automate solutions, making apps and flows available throughout an organization. Doing this requires some centralized support—it's something a CoE might take on, for example—but it can help reduce spending on redundant applications.

Different organizations will choose different approaches to support.

How you choose to support low-code development is up to you; there's no single end state that's right for every organization. What you shouldn't do, however, is ignore it. Whether you know it or not, you probably have citizen developers building apps and flows in your organization right now. You must decide how best to support their efforts. For examples of what organizations are doing today, see the For Further Reading section below.

What to do now

Low-code development lets IT stop being a bottleneck for creating new applications.

Low-code development is a revolution in how organizations create applications. IT need no longer be the bottleneck for building solutions needed by the business. But this new approach fundamentally changes IT's role. Your organization can retain control over data, over how (and even whether) applications run, and other things that IT has long been responsible for. But you can give up control to the business for creating many of those applications. This speeds up your organization's ability to meet the needs of the business, while also letting more technical IT development resources remain focused where they're needed most.

Power Apps and Power Automate can provide the best of both worlds: decentralized development with centralized control. To do this, central IT must play its part. At a high level, you need to do four main things:

1. *Set up monitoring of your low-code environments.*
Doing this is simple, and it gives you complete control over all your organization's apps and flows. If you're not already doing this, expect to find a number of low-code applications in use once you start looking.
2. *Create appropriate DLP policies for your environments and your tenant.* Without this, anyone creating apps and flows can use any combination of connectors in their apps and flows, something you probably don't want to allow.
3. *Assign appropriate roles.* By default, citizen developers have a great deal of control, both in the default environment and in environments they create. You can't change all of this—this low-code platform is designed to empower business users—but you might find it useful to implement more control through roles.

4. *Determine the level of support you want to provide.* Doing nothing is an option, but your support desk will still get calls. A better option is to get ahead of demand by determining what kind of low-code support makes the most sense for your organization, then begin putting this support in place.

IT leaders can't ignore low-code development.

Power Apps and Power Automate offer a huge opportunity to generate business value. And while giving citizen developers more autonomy might seem risky, missing this opportunity is also risky. A new application model is coming, and IT has the chance to get ahead of it.

The truth is clear: IT leaders need to come to grips with the low-code revolution as soon as possible. Playing your part well is how you will create the most value for your organization.

For further reading

To learn more about Power Apps and Power Automate, go to <https://powerapps.microsoft.com/en-us/blog/microsoft-powerapps-learning-resources/>

To see how other organizations, including Virgin Atlantic, Heathrow Airport, the American Red Cross, Standard Bank, and SNCF Railway have succeeded with Power Apps and Power Automate, go to <https://powerapps.microsoft.com/en-us/blog/powerapps-champions/>

To read Forrester's view on the economic impact of these technologies, go to <https://go.microsoft.com/fwlink/?LinkId=2005303&clid=0x409>

For a more detailed look at this topic, see Administering a Power Apps Enterprise Deployment, available at <https://aka.ms/powerappsadminwhitepaper>

© 2020 Microsoft Corporation. All rights reserved. This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples are for illustration only and are fictitious. No real association is intended or inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes. You may modify this document for your internal, reference purposes.

Some information relates to pre-released product which may be substantially modified before it's commercially released. Microsoft makes no warranties, express or implied, with respect to the information provided here.