Microsoft

Power Platform Enterprise Deployment Patterns and Skilling

Maturing Fusion Development Teams Across Power Platform Services

Written by: Brendan Hancock

Technical Contributors: Ken Auguillard, Jack Rowbatham, Nikita Polyakov, Kartik Kanakasabesan, Eric McChesney, Manuela Pichner, Sean Fiene, Marc Schweigert, Kent Weare, Ashvini Sharma

Published: May 2022

Version: 1.0

Microsoft

Executive Summary

This white paper is written for technical decision makers within organizations considering Power Platform services, as well as organizations that have begun leveraging Power Platform services and are seeking guidance on how to develop maturity across the suite of tools. Throughout the narrative of this paper, you will find themes and patterns about how organizations are skilling and scaling Power Platform into fusion development models. This paper will also convey an understanding of the benefits and challenges organizations are experiencing with various deployment models.

A group of blind men heard that a strange animal, called an elephant, had been brought to the town, but none of them were aware of its shape and form. Out of curiosity, they said: "We must inspect and know it by touch, of which we are capable". So, they sought it out, and when they found it, they groped about it. The first person, whose hand landed on the trunk, said, "This being is like a thick snake". For another one whose hand reached its ear, it seemed like a kind of fan. As for another person, whose hand was upon its leg, said, the elephant is a pillar like a tree-trunk. The blind man who placed his hand upon its side said the elephant, "is a wall". Another who felt its tail, described it as a rope. The last felt its tusk, stating the elephant is that which is hard, smooth and like a spear. –Blind men and an elephant - Wikipedia

Much like our elephant parable, the first contact with Microsoft Power Platform creates a different first impression depending on which service a user might first embrace. This sometimes causes confusion about all the tools that fall under this "platform" umbrella. Depending on the depth of technical knowledge of people in an

organization and their target business outcomes, Power Platform will take organizations on varying maturity paths. Some organizations are starting from a professional development posture, they already have deep knowledge of Microsoft Azure and already have technical resources embedded in their IT organization. These organizations with deep technical talent might be seeking to develop rapid application development acumen in their IT department or enable citizen development. On the opposite end of the spectrum some organizations might have a proliferation of citizen developed applications and are seeking guidance on how to build a fusion development practice. A fusion team is a team of professionals with varying technical skills that build Power Platform business applications and solutions, some of whom work daily with Power Platform resources. A successful fusion team will likely have executive sponsors, program managers, professional developers, solution architects, functional consultants, app makers, business analysts, citizen developers, and so on. According to Gartner, "at least 84% of companies and 59% of government entities have set up "fusion teams" — multidisciplinary teams that blend technology and other types of domain expertise and are often designed to deliver products rather than projects."

This construct of fusion development might seem like a lot to account for in terms of a talent matrix, particularly across a suite of services. When an organization lacks prior context on what Microsoft's Power Platform can do with data it's hard to rationalize how to scale out and skill up when data can be connected to anywhere. Organizations first encountering Power Platform are typically being introduced to a tool that uses data for one of the following purposes: reporting/analytics, cloud automation across APIs, robotic process automation, internal web apps built on Microsoft Dataverse, external pages built on Microsoft Dataverse, Power Apps built with APIs, bots, and artificial intelligence models.

Power Platform consists of these technologies to build solutions and each technology has a different learning curve. Depending on the complexity, format and location of the data, and the maturity required to build a solution, Power Platform development can range from extremely simple to immensely complex. Building out a deep competency in just one of these areas can become a full-time discipline. Orchestrating collaboration across all these competencies requires numerous people to work together in project teams and can evolve into a programmatic model where teams work across the full Microsoft platform. While there isn't a one size fits all model for any organization to successfully mature with Power Platform, there are themes and patterns which we will highlight throughout this paper. Culture and collaboration will be instrumental to success.

The remainder of this paper will introduce key ideas to help organizations find their current level of maturity and identify tools needed to advance their organization to their desired level of maturity.

Microsoft

Power Platform Enterprise Deployment Patterns and Practices

For organizations beginning their Power Platform journey, the first thing to understand about Power Platform maturity and deployment is that it is not a singular service or product. Rather, Power Platform is a suite of services, and each has an associated learning curve. If you look back at the various components of Power Platform you will find a 20-year history of services coming together under one umbrella. Each service has different practical applications for an enterprise organization and each service has a no-code, low-code, and pro-code approach.

Power Platform has almost infinite use cases but depending on business needs the use cases might be complex enough to merit full-time IT resources. In many circumstances the sprawl and sophistication of the data being leveraged will align to talent matrixes which are staffed by a combination of professional developers, stakeholders, administrators, IT project teams, advanced app makers, and citizen developers working in tandem. This paper highlights how some organizations align teams of people and deployment models to nurture Power Platform services at scale. Organizational outcomes with Power Platform tend to be unique and so the footprint of people required to build maturity frameworks is also unique in each organization.

To begin maturing with Power Platform, organizations need to understand the "data first" design approach of Power Platform services. Each service within the platform has a way of interacting with data and Power Platform can connect to data where it exists today. Power BI for example has an ecosystem of connectors designed to extract, transform, load, and then visualize large sets of data and publish data in charts and dashboards in a "read only" format. The connectors for Power Apps, Power Virtual Agents and Power Automate differ from the Power BI connectors. These connectors are designed to create, read, update, and delete smaller sets of data across integrated systems with APIs. Model-driven applications and Power Apps Pages are web applications built on Microsoft Dataverse, a relational database, which is hosted natively on the Power Platform. AI Builder is designed to work with structure or unstructured data to build AI models which run against Azure Cognitive Services APIs. Power Automate desktop is locally installed on-premises on a physical Windows machine or virtual machine and interacts with applications on-premises to emulate user keyboard and mouse actions. In order to implement any of these services it will require someone with the ability to navigate the schema of the underlying data or the API built on top of the data.

Environment scaling, ownership, and strategy

The Power Platform admin center is designed specifically for the creation and administration of Power Platform environments and segmentation of the platform's underlying services. Environments are the containers used in the Power Platform for

segmentation of users from app developers and each environment can have isolated components (apps, flows, virtual agents, etc.). At scale, environments become critical for security segmentation as well as application lifecycle management (ALM). For business users an environment strategy might not be apparent because they might build components in a default environment which is shared across all Office 365 users. However, for IT centric Power Platform projects, environments require an ownership matrix which maps uniquely to how a business wants to deploy components to users. Some organizations segment users by line of business, others by geography or language. This presents a preliminary consideration for every organization delving into fusion or professional development with Power Platform. Power Platform environments scale outward, and organizations can have as many or as few environments as desired. If there is not currently a named Power Platform owner, or if there is a sprawl of environments and connections which have been created without oversight, then there is a need for ownership alignment. An organization can have many environments where data can be stored in an Azure Active Directory tenant. The more environments created, the more oversight is required, and resources scaled to manage them. For environments in which multiple connectors have been enabled for integration, the more API connections enabled, the more thought it takes to govern data loss prevention policies. How an organization chooses to scale out a multiple environment strategy is a people and process consideration that is unique to each organization and has a myriad of considerations. Technical owners of Power Platform should consider reading Microsoft's documentation on defining an environment strategy. Successful organizations that want to do more with Power Platform will need more people to orchestrate the desired outcomes.

In addition to researching environment strategies, technical owners and administrators should also read the latest whitepaper regarding Power Platform governance and administration.

Organizations seeking to build out fusion teams or Center of Excellence teams need to embrace ownership of an environment topology. A baseline governance around the Power Platform default environment usually requires some direction from an executive sponsor who nominates an individual or team of people to take on ownership and management of the Power Platform administration center. Beyond the default environment, an organization needs to develop an internal strategy for who takes on ownership and management of each of the environments and the developers, users, and security in those isolated environments.

As organizations plan an environment ownership strategy, they should consider the following:

- Who is the owner of your organization's Power Platform environment administration and data loss prevention policies across all environments?

- Who are the administrators or owners that are responsible for managing individual environments, their users and security roles of individuals in those environments?
- Where does data reside, and how is it secured within the context of Power Platform architecture and connectors?
- Who can navigate the schema and API of data which is connected to via Power Platform services?

Nurturing citizen development maturity in default environment

With a citizen development team in the default environment, there is generally not much of a "center" in terms of support and alignment to an IT staffed implementation. Some would call this a decentralized model, best fostered by creating enthusiasm, fostering Communities of Interest, then developing deeper competency. It's important to caveat, some organizations without large or centralized IT departments can thrive in this manner without a formal Center of Excellence. We will discuss both Communities of Interest and Center of Excellence later in this paper.

Clever business users tend to build elegant solutions with the tools they are given. However, they might run into challenges when trying to scale them out organizationally. There can be business models which are organizationally designed to be siloed across multiple Azure Active Directory tenants. Those organizations tend to start their low-code journey in pockets of experimentation without a centralized IT team or identity model, and these also tend to have decentralized approaches towards Power Platform.

In a model which is trying to grow from citizen development to pro development we see a maturity model that grows from an initial experimentation to a centralized model which eventually becomes supported by IT. Organizations sometimes find that citizen developers have created solutions which are valuable to scale across multiple lines of business or even an entire enterprise. This is a common trigger to start fusion development. This is a maturity pattern where we start with business users that gradually develop skills on the Power Platform, then subsequently require assistance from an IT-staffed team to scale out organizational solutions.

At minimum, organizations need to understand that there is a default environment in the Office 365/Azure Active Directory tenant in which all users are enabled as makers to create Power Automate flows and Power Apps. This is generally considered an environment space designed for productivity and collaboration within the context of Office 365. Users cannot be disabled from building apps or flows in this environment. However, they can have most connectors restricted within the Power Platform administrative center. Citizen development thrives in this default environment, but

citizen developers are expected to build things for themselves, and their teammates, then return to their normal jobs.

At the time this paper was written, there were also certain connectors under the Office 365 suite which could not be explicitly blocked in a default environment. Most organizations want to have a baseline DLP policy applied to even this default environment and it is someone's responsibility to make sure that policies are created and enforced. Therefore, even organizations just beginning with a decentralized citizen development Power Platform methodology have an initial ownership task that is generally IT supported: Define an owner(s) responsible for oversight of DLP policy for the default environment.

Nurturing professional development maturity with solutions across dedicated environments

Consider a traditional Application Lifecycle Management (ALM) topology with dedicated environments for development, test and production in which an organization has a critical suite of business applications. These environments which have a formal integration, testing and staging process are tightly aligned to solutions. Some triggers which should indicate the need for creating fusion teams or Center of Excellence teams is the requirement for ALM, continuous integrations continuous development (CI/CD). It is very typical to see a three-environment topology (or more) which consists of multiple environments working with automated deployment processes. In almost any scenario where software development lifecycle (SDLC) is a requirement, it will blend into a fusion or professional development aligned staffing matrix. The topic of SDLC is deeply integrated into Power Platform solutions topology and an understanding of solutions. How they are managed and integrated with Azure DevOps for CI/CD orchestration and GitHub for source code management is a technically deep consideration.

Professional grade software development lifecycles can be implemented and managed within a multi-environment strategy and the scale of people involved can grow into a multi-developer team. Organizations seeking this type of Power Platform maturity tend to have dedicated professionals that work in-depth on specific Power Platform tools and for this type of skilling individuals on the team have extensive depth on the specific technologies they work with.

This is more akin to a centralized team that works on complex business applications and is staffed with full-time IT professionals. This model is typical in organizations deploying multiple Power Platform tools at enterprise scale and very typical for organizations implementing Dynamics 365 Customer Engagement applications. These types of resources tend to move from one Power Platform project to another project and this constitutes full time work, this is their job.

Microsoft

In a professional development framework, Power Platform talent alignment falls into an IT supported role and responsibility framework which requires orchestration and ownership by service. In the diagram below you can see that their dedicated roles for the various Power Platform services and the headcount in those roles can scale based on the number of Power Platform projects an organization is implementing.

IT								
IT Monitoring Group	IT Developers (Pro)	Power Platform Service Owner	Power Platform Environment Owner	IT Security Group	Service Catalog Owner	Service Management Team	App Portfolio Group	Licensing Group (Finance)

Business Units				
Environment Owner	Citizen Developers	App Users	App Portfolio Owner	Financial Management Group (chargeback)

External Resources		
Microsoft	Partners	Consultants, etc.

In the typical talent pool, there are dedicated professionals that specialize in one or two Power Platform services (service owners), in most enterprise organizations there aren't implementation resources that try to master everything. This document highlights how organizations are developing maturity in each service throughout this paper.

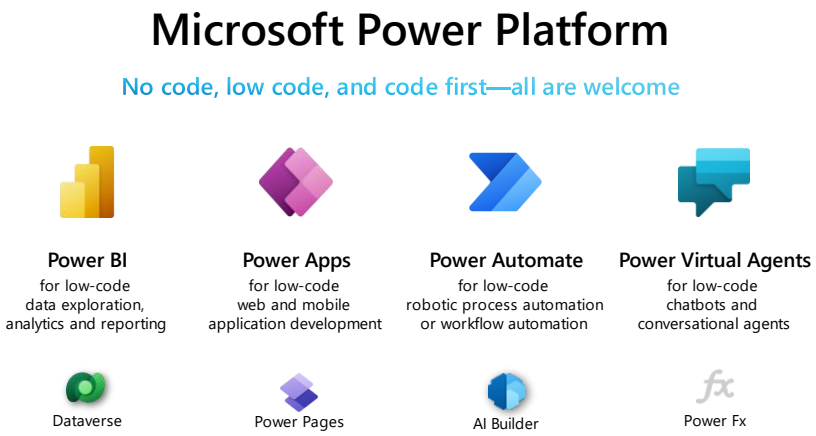
Each organization developing deep skills on Power Platform also needs to consider that each service is under constant development with Microsoft's product teams and that these services evolve at a rapid pace. Individuals seeking depth in specific Power Platform services must keep up with the constant changes to keep up with any one technology.

Although there might be some exceptions for generalist knowledge such as solutions architects, it is difficult for any one person to keep up to speed with all Power Platform services as a professional development resource all at once. Therefore, teams of specialists exist both within Microsoft as well as Microsoft's partner ecosystem and these specialists collaborate to architect solutions across the Power Platform services.

Power Platform Services

Power Platform is a suite of products and services to create line of business applications, automations, analytics, and bots. With Power Platform you can empower everyone at your organization with an intuitive, collaborative, and

extensible platform of tools that makes it easy to create efficient and flexible solutions. With each of these services, there is a no-code/low-code entry point and pro-code extensibility.



Deployment models tend to depend on the complexity of the data, technical requirements and security controls that protect data. Citizen development talent matrixes might start with individual business end-users building simple solutions for themselves which then scale up to teams of admins, advanced app makers, and professional developers building enterprise grade applications. Whereas professional developer talent matrixes could start with development of a Center of Excellence which branches off into great depth by each Power Platform service and later fuses together with citizen developer teams.

Power Apps

Power Apps consist of three types of services which are becoming increasingly coupled/converged with one another. Model-driven apps are web forms designed for internal employee use and they are configured and played through a browser, these web app forms are built exclusively on the Dataverse data service in Power Platform. Pages apps are web forms designed for external customer or partner use, they are surfaced through a web portal and are also exclusively designed on the Dataverse database. Canvas apps are designed for both external and internal users and primarily for being used on touch interfaces such as tablets and mobile phones. Yet canvas applications are interacting with data via connectors and the APIs they are built on which means data can reside in multiple locations.

Canvas Applications

Canvas applications are primarily designed as mobile/tablet for touch interface applications built on top of Power Apps connectors. They can also be embedded in

Microsoft

Microsoft Teams or browsers for mouse/keyboard interaction. Connectors act as a proxy wrapper around an API. There are hundreds of pre-built connectors for existing Microsoft services as well as 3rd party services such as Adobe. When a connector does not exist for on-premises systems or 3rd party service an organization can develop custom connectors on APIs.

Organizations thematically tend to have first perceptions about Power Apps because they have been introduced to canvas applications from SharePoint, One Drive, Microsoft Excel, or one of our workshops like App in a Day. Citizen development maturity tends to begin with building a simple mobile application user interface on SharePoint or Excel (usually a single list/table of data in structured columns). Citizen developers tend to come up with simple solutions to problems they encounter in their day-to-day work. These citizen-developed apps aren't expected to scale, integrate or develop complex security around their solutions. Citizen developers go back to their day job when they have finished building a solution to their own challenge. When those solutions need to be deployed across an organization that is a trigger for IT involvement whose job it is to manage solutions full time and at scale. This citizen developer experience of building a user interface with a drag and drop configuration much like PowerPoint is a frictionless entry point for those with no coding experience. As resources mature on canvas applications, they gain depth on the formula language which drives canvas application behavior known as Power Fx. From a talent alignment perspective, the Excel savvy business users and SharePoint users can pick up this type of development with relative ease compared to full stack development, because the data source they are interacting with tend to be simple, such as a single SharePoint list (flat table). Citizen development thrives with these low-complexity data structures and maturity can be fostered with communities of interest without significant need to involve information technology teams, aside from establishing governance. Often organizations establish root data loss prevention policies and simply enable citizen developers to build canvas applications on these existing tools they already know inside the "Default Environment".

However, most organizations tend to have more robust data structures and repositories and almost all organizations have data distributed across multiple cloud services. As Canvas App maturity grows in an organization, developers that understand how to navigate connectors and APIs becomes a critical skill set. Each connector has an underlying API with varying complexity and capability; learning how to build applications across multiple connectors can become a full-time discipline particularly for complex systems such as SAP.

At this degree of maturity, data is sprawled across systems with disparate security residing in each location. A baseline involvement of information technology teams should be included to administer Power Platform and to manage data loss prevention. Technical team resources should be included in post-citizen development to have a baseline governance and environment strategy assuming the data has an

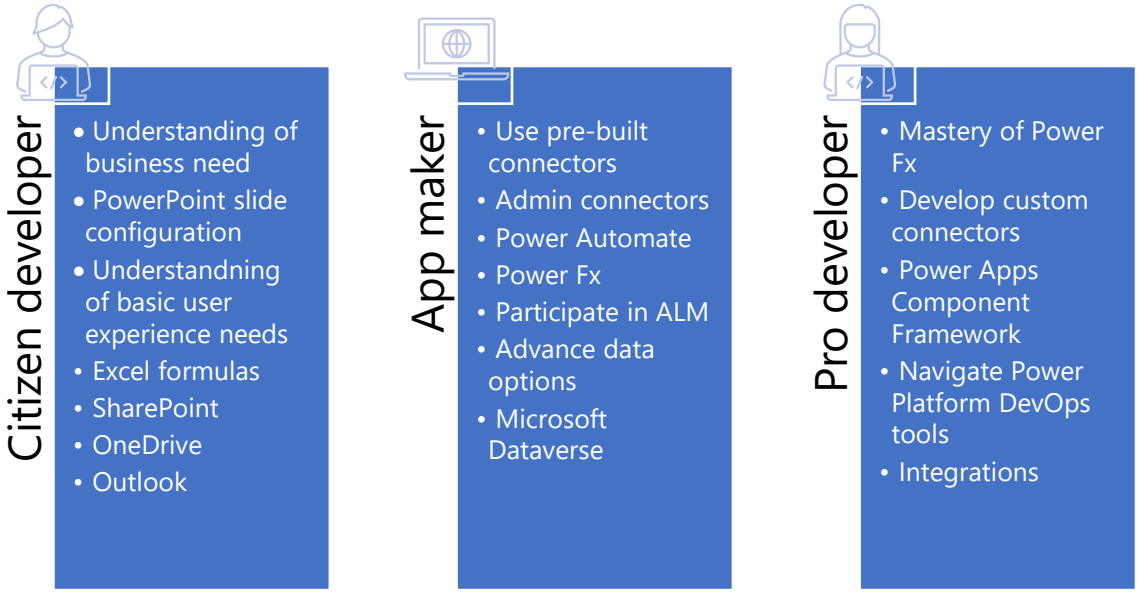
API it can be connected to, and canvas applications can be built on top of the data via the connectors.

Imagine a scenario where an organization is leveraging some of the most popular connectors such as SQL Server a first party Microsoft connector, and two of the most popular external connectors for Power Platform, the Salesforce connector and the ServiceNow connector. This type of data proliferation is extremely common in enterprise organizations and requires the canvas application developer to have the wherewithal to interact with the relational structures within those underlying data stores.
Power App makers can leverage canvas applications to build user interfaces on these data sources extremely rapidly, but those Excel savvy business end users that lack an understanding of relational database concepts may not readily pick up this type of design.

This talent alignment is drifting into the type of talent pool that requires service administrator or developer acumen; particularly for navigating the API and custom data schemas which have been developed in SQL, Salesforce, ServiceNow. This represents the middle tier complexity of canvas apps makers, where being able to navigate APIs and data structures is critical, however still does not necessarily require full stack professional development experience. This is low-code, but not no-code.
As for the professional coder resources in an organization, the tooling has been gradually evolving within canvas applications to be designed for rapid application development. If organizations are seeking reusable building blocks for canvas applications, then professional developers can get started with the component libraries for canvas applications.

If application lifecycle management or continuous development/continuous integration needs to be applied to Power Platform then a broader understanding of Power Platform solutions needs to be studied, understood and aligned to an organization's application development methodology.

Depending on developer preferred tools, Power Platform solutions have had DevOps build tools for solution orchestration for some time and more recently debuted GitHub Actions which have similar capabilities. If these enterprise grade capabilities are sought after to deploy large/complex applications then organizations need to consider staffing professional development teams to design to their business outcomes, this deployment sophistication is becoming professional development.
Different members of the canvas app maker team need different skills. Below is a non-exhaustive list of some of the skills by maker roles. A more detailed list, and links to learning resources can be found at the end of this paper.



Model-driven Apps

Model-driven Power Apps are built on Dataverse which evolved from the Dynamics 365 Customer Engagement backend (sometimes called XRM) and presents a web application user experience with over a 20-year history of no-code relational database development, including workflow, business rules, business process flows. Additionally, these model-driven apps have professional developer extensibility via plugin, web services and customized steps and actions. Model-driven applications are relational database applications with a Dataverse backend hosted in each Power Platform environment.

Model-driven Power Apps have native low-code configurability within the Power Platform maker studio for those non-developers. For pro-developers there are also multiple ways to extend them via APIs as well as various professional developer approaches shared in the Dataverse developer documentation.

One consideration when it comes to development of model-driven apps is whether to build multiple applications in a single environment with a single database, or to create multiple environments with multiple databases. If an organization chooses to build on a single environment and single database, then that generally runs into complexity when trying to build security models and logic to segregate users in different organizational business units (a security construct of Dataverse). Alternatively, when an organization wants to create multiple databases, then each database creates integration complexity when trying to integrate to other critical line of business applications. Either choice requires alignment to unique organizational

structures and a clearly defined environment strategy. Regardless of which choice is made "single or multiple databases," either path runs into security and integration complexity that the platform is designed to handle.
The more model-driven applications you want to deploy to within an organization, the more people it takes to analyze, implement, train/adopt, and administer them. This is another example of scaling talent based on the proliferation of multiple applications. The more complex an individual model-driven application is in its data schema, security, logic and integration the greater depth of skill it takes across people to implement that application.

Model-driven application creators are building out a database schema consisting of tables, columns, forms, system views, charts and relationships which are then packaged into a web application. Developers of model-driven apps tend to have a fundamental understanding of relational database constructs and some exposure to low-code relational database tools. Perhaps a savvy Excel user or a prior Access Database maker is a good first person for model-driven app maker, particularly if seeking a low-code approach to building applications. However, those familiar with Microsoft's partner ecosystem are aware that building model-driven Power Apps can become a full-time consulting discipline and many who have exposure to this discipline originated from a Dynamics CRM background.

Citizen development approaches to model-driven applications often initiate with technology professionals migrating away from prior low-code technologies such as Access, Excel, InfoPath, Lotus Notes, SharePoint lists, etc.

Lots of documentation about patterns for these types of low-code migrations exists but learning Dataverse and model-driven application is not a 1:1 mapped discipline with any of these prior low code tools like Access and Excel. There are even tools specifically designed to help with these data migration/consolidation projects to Power Platform, but still a Dataverse learning curve post-migration. From a purely citizen development perspective there are themes and patterns between relational database models and spreadsheets but not directly mapped features or design approaches.

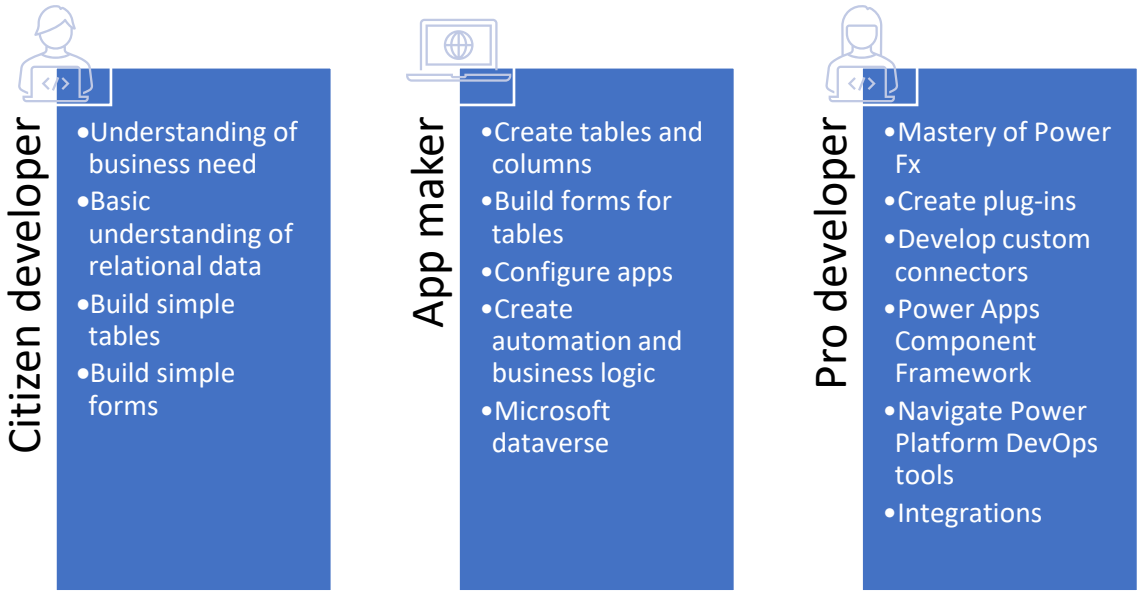
Model-driven apps evolved from Dynamics 365, the depth of model-driven applications can be immense and heavily intertwined with professional development tools and various professional code extensible approaches. There are full time professional developers that specialize in building Dynamics 365 and model-driven applications. Some of the most complex model-driven Power Apps are built with iterative implementations in mind so implementations can gradually incorporate more lines of business over months/years. Microsoft's own Dynamics 365 Sales, Customer Service and Field Service apps are examples of first party Model Driven applications which have had significant professional development extensibility built for complex enterprise business application orchestration.

Microsoft

Building model-driven Power Apps at enterprise scale is a discipline that is implemented by professionals that often constitute teams of business stakeholders, business analysts, project managers, developers, administrators, and end users. Either Dynamics 365 applications or complex model-driven Power App merit alignment to resources that have a deep understanding of the backend of Dataverse. Most organizations that want to deploy model-driven applications at enterprise scale either hire professional services organizations to build the applications to specification or build internal teams in a COE (Center of Excellence) to implement and administer the applications. Microsoft publishes a Center of Excellence Starter Kit as a baseline set of templates to begin fostering organizational maturity regarding standards, consistency and governance. This starter kit template consists of tools and these tools require people to learn how to use them and build a process around them that meets their organizational needs.

Before going down the path of trying to build any model-driven application, organizations should perform due diligence to see what Microsoft and partners have already built as offerings. There are many pre-built schemas for model-driven applications that are developed by Microsoft as well as Microsoft's partners readily available on Microsoft App Source.

Different members of the model-driven app maker team need different skills. Below is a non-exhaustive list of some of the skills by maker roles. A more detailed list, and links to learning resources can be found at the end of this paper.



Microsoft

Power Pages

Pages is a new low code experience built on top of the foundational architecture which evolved from Dynamics 365 Portals. At the outset of a Pages project an organization needs to first understand that Pages are designed to expose data to a public facing website, or a website with a logged in user. Pages therefore can have quite a few different types of external users that will consume content and data from Pages. Customers, partners, or anonymous browsers are personas that might interact with a Page and its underlying data.

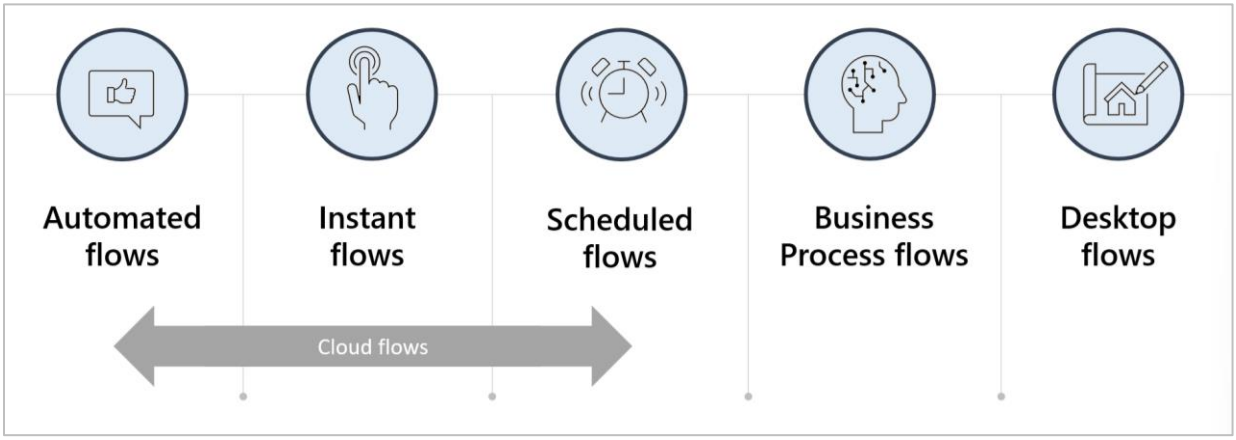
If an organization is doing anything more than building an anonymous content driven website with open access to data, then they will have to design an architecture which deals with authentication and authorization of logged in users. In order to build authorization to underlying data sets, Page developers will have to familiarize themselves with the underlying data schema of Pages built on Dataverse and understand relational database security constructs. This includes designing how pages are built on forms and tables which are then exposed to web roles and authorized logged in users.

In order to build authentication with an external facing website, there are a myriad of design choices and integration capabilities with multiple authentication tools. There is usually administrative overhead to working with external customers (website navigation, password resets, ticket/case management).

Documentation for Pages helps curate how to build citizen development on the new Pages experience with basic forms on pages. The documentation content also curates how to build advanced forms on pages. Historically speaking enterprise Page deployment evolved from Dynamics CRM Portal deployment and was staffed as a professional developer role. There is significant programmatic extensible capability to extend Pages with VS Code, the Web Portal API and Liquid. Perhaps the most important consideration of any Pages project is building the knowledge required to navigate the backend of Dataverse and deploy security around it.

Power Automate

Power Automate currently consists of two workflow automation services. Cloud Flows deal with workflow orchestration across APIs within Office 365, 3rd party service, as well as through gateways connecting to on-premises data services. Desktop Flows consist of a robotic process automation (RPA) tool which is designed to emulate user clicks and mouse interactions that is running on a local Windows device or virtual machine. In combination these two automation tools work as a formidable suite which enable organizations to build automations across data almost anywhere it resides. Business Process Flows are sometimes included in the Power Automate conversation. We will discuss this more later in the paper where we discuss Dataverse in greater detail.



Cloud Flows

Power Automate (formerly Microsoft Flow), is built on top of Microsoft Azure Logic Apps and is available to organizations as a SaaS based cross API workflow application. At a high-level Power Automate provides users the ability to configure automations across different systems using triggers, conditions and actions (integrated workflow).

There are hundreds of pre-built connectors within Power Automate, there is also the ability to access on-premises services via gateway connectors, and finally the ability for professional developers to create custom connectors for their internet facing applications.

Power Automate flows can be very simple and most of the citizen development with cloud flows starts when end users deploys one of the hundreds of templates in the Power Automate service. Some of the Power Automate cloud flow functionalities are built into various Office 365 plans and there are adjacent services within Office 365 such as SharePoint, Microsoft Teams, OneDrive, or Outlook which are some of the most common services on which to build citizen-developed automations.

Getting users started on these Office 365 services for automation is relatively simple, because users already understand how to use them. Business end users are remarkably savvy at identifying their own redundant data entry tasks across Office 365 services and those tasks are ripe for automation. These scenarios tend to be low-hanging fruit and often can be built by end users without any IT intervention and the security backbone for these scenarios is all unified under the Azure Active Directory tenant umbrella.

Power Automate flows begin to get more complex when introduced to data sources on-premises, outside of the Azure Active Directory tenant or in 3rd party connectors which are also outside of your tenant. Cloud flow automations have some formidable return on investment when they can act on underlying on-premises sources of data such as SQL on-premises. In order to establish this type of connection, organizations need to install local gateways that can proxy into these on-premises services on infrastructure behind a customer's firewall, virtual machines or personal PCs.

One theme is scaling out automations on 100+ SQL on-premises servers. This can be a large infrastructure project to setup behind a firewall, particularly because enterprise scale usually uses more than single VM per SQL instance, opting to spin up a cluster of machines and configure them for optimized network load balance and high availability scenarios. The type of team assembled to tackle this large automation project also consisted of the SQL Database Admins and SQL Developers that build all of these custom SQL database schemas. This type of Power Automate project requires heavily sponsorship from an IT department and is usually implemented by professional developers, and supported by a network team to keep all the virtual machines optimized, gateways updated, and keep downtime to a minimum. This thematic type of transformation has significant ROI but also significant IT investment.

Power Automate flows also tend to get more complex when connecting to 3rd party cloud services. Having the ability to connect external services is straightforward, but these 3rd party services have their own identity models and data models. For example, orchestrating automations across Salesforce and ServiceNow would involve aligning to technical resources that have a fundamental understanding of both the security model and the data available in the APIs of these applications. In order to successfully build automations across these 3rd party connectors, you generally need to include in people that can design a security model across these services as well as resources that can navigate their underlying data structures in the API of those 3rd party services. Consider that with hundreds of connectors available, this type of automation becomes a micro-services architecture and needs to be staffed with talent that understands each service and orchestrate across services.

Power Automate flows tend to need professional development resources when development of custom connectors is underway. If an application does not already have an external facing API, the development of that API would be one of the first steps required to build a custom connector. Once an external facing API has been developed, the person building automations on top of it is going to require an understanding of the underlying API to orchestrate workflows across it (or at least be able to explore/navigate an API).

Power Automate Desktop Flows (RPA)

Power Automate Desktop has two deployment models, attended and unattended. What this naming convention translates to is RPA for individual use on Windows 10/11 while a user is logged into their own device vs RPA scaled out on virtual machine infrastructure to deal with automations which are always running without any user needing to be logged in. Attended scenarios tend to lend themselves well to simple automations built by business users, whereas unattended scenarios tend to require a larger IT supported infrastructure project with dedicated RPA developers.

The Power Automate Desktop (RPA) solution was previously called WinAutomation and it was acquired with Softomotive by Microsoft in 2020 to address local on-premises robotic process automation. This RPA tool had been under development for 15 years at the time it was acquired. While this application is included with Windows 11 for users to record simple processes and automate them, at enterprise scale RPA can become a massive project that customers are encouraged to build a Center of Excellence around.

Before diving in-depth on how to think about skilling and scaling with Power Automate Desktop, organizations should understand that RPA is considered a secondary option for automation. Microsoft recommends only using RPA where an API cannot be used, and an API cannot be developed. The cost of scaling out infrastructure for RPA can be significantly higher than writing an API. It is Microsoft's recommendation that any time there is an opportunity to build an API automation with Power Automate cloud flows, Microsoft customers are encouraged to do so. Enterprise grade deployment of Power Automate Desktop usually entails deploying infrastructure, such as multiple virtual machines either in Azure or behind a customer's local network firewall. Along with this infrastructure build out there are considerations for creating redundant clusters of virtual machines as well as performance tuning considerations for optimal automation. Infrastructure resources need to be part of a collaborative project to implement RPA at scale successfully. There is also an administrative responsibility with RPA to update virtual machines and keep the RPA software up to date as well, so consider long-term maintenance part of an implementation plan.

Some of the core functional capabilities of RPA tend to be quite technical at the outset. Inputs, outputs, loops, variables tend to be concepts adopted by people with a fundamental understanding of software development. At a slightly deeper level, understanding how to work with the front-end of systems that RPA interacts with is also a discipline which usually requires a technical understanding of the application (Active Directory, Exchange, databases, Windows). Then perhaps at its deepest functionalities, RPA can interact with XML, cryptography, FTP, and cognitive services across multiple cloud providers. The latest Power Automate RPA Developer exam includes the following recommended experience:

"Candidates should have experience with JSON, cloud flows and desktop flows, integrating solutions with REST and SOAP services, analyzing data by using Microsoft Excel, VBScript, Visual Basic for Applications (VBA), HTML, JavaScript, one or more programming languages, and the Microsoft Power Platform suite of tools (AI Builder, Power Apps, Dataverse, and Power Virtual Agents)."

Whereas some Power Platform components were designed for citizen development then gradually became more aligned to professional development resource, RPA seems to have been developed for professional developers and gradually became more accessible to citizen users. Organizations that have developed a Center of Excellence around RPA solutions tend to have full-time automation developers whose sole responsibility it is to automate on-premises workloads. Organizations seeking to scale RPA across an enterprise that don't have internal automation developers are encouraged to seek out engagement with partners or staff internally.

Power Virtual Agents

Power Virtual Agents are accessible to citizen developers that are building out a dialogue which interacts with internal employees, particularly if they are building on a single data source they can already navigate. The backbone of integration with the virtual agent interface is Power Automate and so should be thought of as a user facing interaction layer of an automation practice. Virtual agents came into fashion because of deeper complexity challenges with the Azure Bot Framework upon which virtual agents are built. In essence virtual agents were created to make a simpler, configurable, friendlier SaaS based application for bot dialogue which could be embedded in multi-lingual and multi-channel frameworks. Virtual agents can take advantage of some of the more complex capabilities in the bot framework such as skills and currently can be exported into the Bot Framework for custom development extensibility. The Azure Bot Framework Composer is included for free in the Power Virtual Agent experience all hosted as a SaaS. Developers get the opportunity of having two authoring canvases for the price of one based on their preference.

Power Virtual Agents hosts multiple AI models and AI capabilities on a single service, the core of which is a transformer-based natural language understanding (NLU) model. Different from the traditional approach, Power Virtual Agents language understanding model uses the example-based approach powered by a deep neural model. Such large-scale model only needs to be trained once with massive amounts of data using AI supercomputing, and can be later used for specific tasks with just a few examples and no further training. This is part of Microsoft's AI at scale initiative—which basically changes the way AI is developed. It provides an intuitive way for bot makers to work on the bot content simultaneously and confidently, without having to involve AI experts.

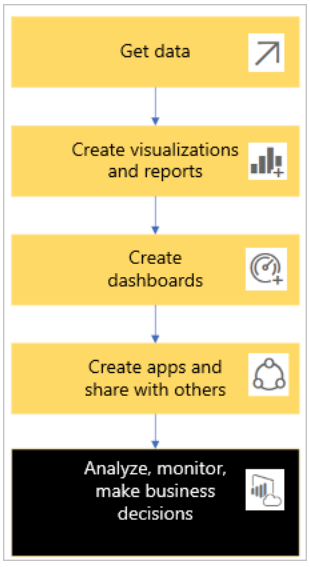
As far as talent alignment is concerned, the skills needed to build Power Virtual Agents at scale is essentially the same as Power Automate cloud flows with a flavor

for writing customer/employee conversation interaction design. Virtual agents usually start simple and scale out to multiple connectors in the Power Platform ecosystem. Citizen developers can get far with virtual agents and pre-built connector as long as they understand what they are connecting to (Exchange, SharePoint, One Drive, etc.)

Where professional developers tend to get integrated into Power Virtual Agents is when customer facing virtual agents are created, or custom connectors need to be created, or where citizen developers don't know how to interact with the data in connectors, or where connectivity to on-premises systems is a requirement (including RPA.) Because virtual agent can leverage Power Automate for workflow integration, the skill sets are essentially the same as Power Automate.

Power BI

Power BI was the first service to be branded under the Power Platform. Over the course of its development many Power BI data sources have been created for the purposes of developing reporting, dashboards, and analytics. The connectors for Power BI are different from the connectors for Power Apps, Power Virtual Agents, and Power Automate. Additionally, the administration of Power Bi is currently separated from the other Power Platform components, working with a security construct called a "Workspace" which is similar to an "Environment" in Power Platform. For on-premises data Power BI is designed to work in orchestration with the Power BI on-premises data gateway. The Power BI stack consists of Power BI Desktop and the Power BI Service, that work in orchestration to perform the following functions.

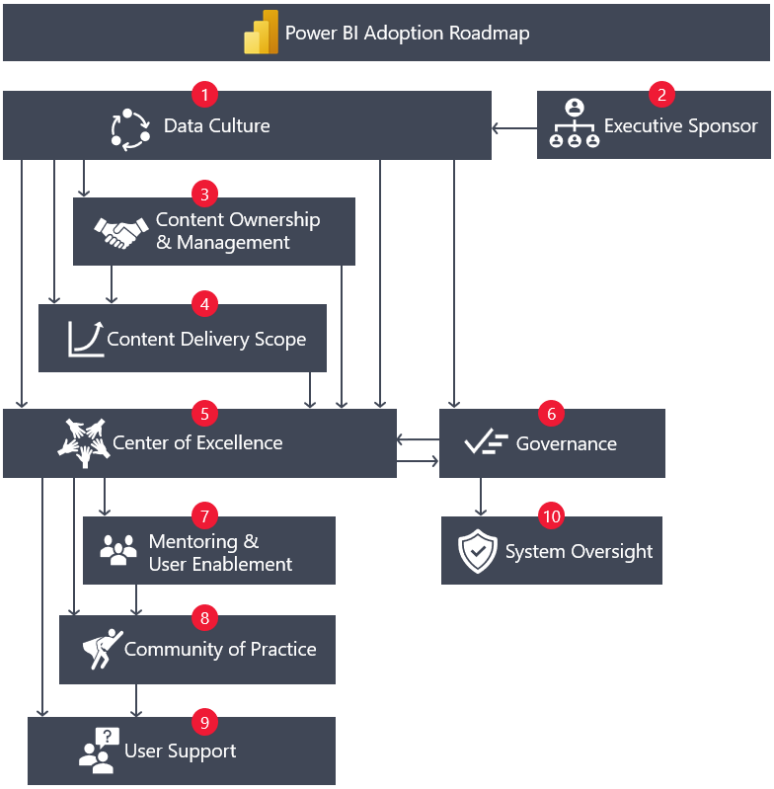


There is already tailored documentation for Power BI for business users as well as Power BI Administration. For developers there is guidance on Power BI APIs for

developers and developing custom visuals. Additionally, there is guidance available for individual report creation Power BI Desktop for individuals as well as Enterprise Report Creation. We have included links to this documentation at the end of this paper.

Power BI as a service does an amazing job with users familiar with slicing data in Excel. A good analogy is to think Excel pivot tables and visualizations for citizen development or individual report creation. Power BI often scales from individual report writers to small teams without significant IT support. Generally, users have a depth of understanding of the data they interact with and can wrap their understanding of data into the Data Analysis Expression (DAX) language.
There are a few areas/triggers which elicit building out an IT supported talent framework. When complex Azure ETL, or Data Flow ETL operations must be completed on data sets it usually requires a data analyst specific resource. When large on-premises implementations require setting up virtual machine gateway infrastructure, it tends to elicit infrastructure and administrative resources. When extensibility is required to develop custom AI or machine learning capabilities Python or R Models the project often requires developers or even professional data scientists. When significant segmentation of Workspaces or row-level security is required, BI projects tend to incorporate information security and administrative resources.

Power BI also has varying pricing and tiers of premium capacity available for deployment. Managing administration, security and capacity are all constructs that can become deeper considerations for organizations deploying Power BI at enterprise scale. There are 12 additional Power BI white papers to explore if you are a technical decision maker considering Power BI deployment. Additionally Microsoft curates more in depth Power BI adoption documentation as well as documentation outlining how organization are deploying adoption and maturity roadmaps outlined in the diagrams below.



1. **Data culture** Data culture refers to a set of behaviors and norms in the organization that encourages a data-driven culture. Building a data culture is closely related to adopting Power BI, and it is often a key aspect of an organization's digital transformation.
2. **Executive sponsor** An executive sponsor is someone with credibility, influence, and authority throughout the organization. They advocate for building a data culture and adopting Power BI.
3. **Content ownership and management** There are three primary strategies for how business intelligence (BI) content is owned and managed: business-led self-service BI, managed self-service BI, and enterprise BI. These strategies have a significant influence on adoption, governance, and the Center of Excellence (COE) operating model.
4. **Content delivery scope** There are four primary strategies for content delivery including personal BI, team BI, departmental BI, and enterprise BI. These strategies have a significant influence on adoption, governance, and the COE operating model.
5. **Center of Excellence** A Power BI COE is an internal team of technical and business experts. These experts actively assist others who are working with data within the organization. The COE forms the nucleus of the broader

community to advance adoption goals that are aligned with the data culture vision.

6. **Governance** Data governance is a set of policies and procedures that define the ways in which an organization wants data to be used. When adopting Power BI, the goal of governance is to empower the internal user community to the greatest extent possible, while adhering to industry, governmental, and contractual requirements and regulations.
7. **Mentoring and user enablement** A critical objective for adoption efforts is to enable users to accomplish as much as they can within the guardrails established by governance guidelines and policies. The act of mentoring users is one of the most important responsibilities of the COE. It has a direct influence on adoption efforts.
8. **Community of practice** A community of practice comprises a group of people with a common interest, who interact with and help each other on a voluntary basis. An active community is an indicator of a healthy data culture. It can significantly advance adoption efforts.
9. **User support** User support includes both informally organized, and formally organized, methods of resolving issues and answering questions. Both formal and informal support methods are critical for adoption.
10. **System oversight** System oversight includes the day-to-day administration responsibilities to support the internal processes, tools, and people.

Additional Power Platform components

AI Builder is a Power Platform capability which provides AI models to streamline business processes. The AI models can be leveraged within both Power Apps as well as Power Automate. The AI builder tool is designed to make AI models accessible to non-developers that would otherwise require a development background. These AI models are all working with the more developer centric Azure Cognitive Services APIs. The capabilities that AI Builder enables can help with automating a manual process such as form processing and perhaps the best part is that these powerful capabilities are aligned to beginners. The learning documentation for using AI builder is primarily geared towards beginners that does not require any coding experience whatsoever.

Microsoft Dataverse is the evolution of 20+ years of low-code relational database development and it has historical origins in the Dynamics CRM/365 CE ecosystem. It has previously been called the Common Data Service as well as XRM. Dataverse is a SaaS based service that enables organization to create enterprise grade applications with the following capabilities.

- Azure Active Directory security

- No code relational data modeling
- No code role-based, row/field level security
- Ability to model hierarchical security models
- Offline sync framework
- No code data import with Power Query
- Metadata driven UI from data model (model-driven Power Apps)
    - Forms, grids, data filtering, charts, dashboards
    - Dataverse security which drives UI security
    - Ad-hoc query tool over all data in Dataverse
    - UI extensibility with code
- Web apps and mobile apps
- No code business rules and business process flow framework
- No code synchronous and asynchronous business logic
- Business logic extensible to .NET code
- Web services layer over data model

A handful of Power Platform services explicitly require Dataverse for the backend including model-driven Power Apps, AI Builder and the Power Pages. Other Power Platform services such as canvas Power Apps, Power Automate, Power BI do not explicitly require Dataverse. While not explicitly required for some Power Platform services, there are certain capabilities such as ALM which require Dataverse solutions which enable moving Power Platform components from one environment to another environment.

Dataverse presents an ecosystem of configurable integrations to the broader platform of Office 365 including SharePoint, Microsoft Teams, Exchange, Project, Excel Online. Dataverse is also the engine which drives Microsoft very own first party business applications Dynamics 365 Customer Engagement applications as well as many of the industry accelerators developed by Microsoft and Microsoft partners. The actual architecture of Dataverse is hosted across SQL for relational data storage and BLOB storage for unstructured data. Organizations don't manage infrastructure in the SaaS model, but purely from an ownership and talent alignment perspective talent frameworks tend to work well with organizations that have had database administration skills or Dynamics CRM skills. When Dataverse databases proliferate across an organization there is generally an IT sponsored Center of Excellence responsible for establishing governance across databases.

Simple Dataverse applications tend to begin when organizations are running into scalability issues with other backends for canvas Power Apps such as SharePoint lists

or Excel. Dataverse applications can be very straightforward, perhaps with only two or three tables which can be built with relatively low complexity by citizen development resources (some of these start out as Power Apps in Microsoft Teams). Middle complexity Dataverse application might look more like Access databases and there are even tools for organizations migrating Access sprawl into the centrally managed Power Platform on Dataverse. Yet there is deep capability though with Dataverse to build enterprise applications with complex integrations and programmatic extensibility and these types of projects have historically been deployed by Microsoft partners, particularly ones with a Dynamics 365 background. It might be a cliché or a catchall phrase when it comes to Dataverse project and talent alignment, but "it really depends on what an organization wants to do". Organizations are building everything from very simple "Access-like" applications for small teams of users, all the way to globally scaled enterprise Dynamics 365 applications. Simple applications might consist of a table and a few columns and be presented in Microsoft Teams built by someone with no programmatic experience. Complex applications can consist of swaths of integrations with upstream, downstream, and bi-directional integrations coupled with custom code extensibility and full stack development talent.

Power Platform is made powerful by its ability to leverage data across many platforms. To do this, components of Microsoft Power Platform use connectors. You can think of connectors as a bridge from your data source to your app or workflow which allows information to be conveyed back and forth. Connectors allow you to extend your business solutions across platforms and add functionality for your users. Connectors enable you to connect apps, data, and devices in the cloud. There are hundreds of connectors for Microsoft Power Platform, enabling all of your data and actions to connect cohesively. Examples of popular connectors include Salesforce, Office 365, Twitter, Dropbox, Google services, and more. In addition to the many published connectors organizations can make their own custom connectors using available APIs for virtually any data source. Organizations which are maturing on Power Platform have implemented effective DLP policies to allow these connectors to be used while still protecting important data.

Culture and Deployment Models

"Knowledge is powerful and so it tends to be hoarded. Experts in any field rarely want people to understand what they do, and generally enjoy putting people down." –Ted Nelson's Computer Lib, 1974

Effective fusion deployment models require knowledge to be shared between multiple groups and to incentivize those with the most knowledge to share with the people with less knowledge. Technical lines of business within an enterprise

organization have historically hoarded knowledge into what has been described as "the computer priesthood." Frustration ensues within business units that leverage technology because they are never given enough knowledge to build or use systems built for them, the frustration often leads business to create tools they understand without IT interventions (Email, Excel, Access). This posture wherein IT organizations horde knowledge causes two cultural problems; a small group of knowledgeable resources which is a technology bottleneck and a culture of shadow IT. These two cultural challenges are the greatest obstacle to overcome when fusing a Center of Excellence and a Citizen Development model.

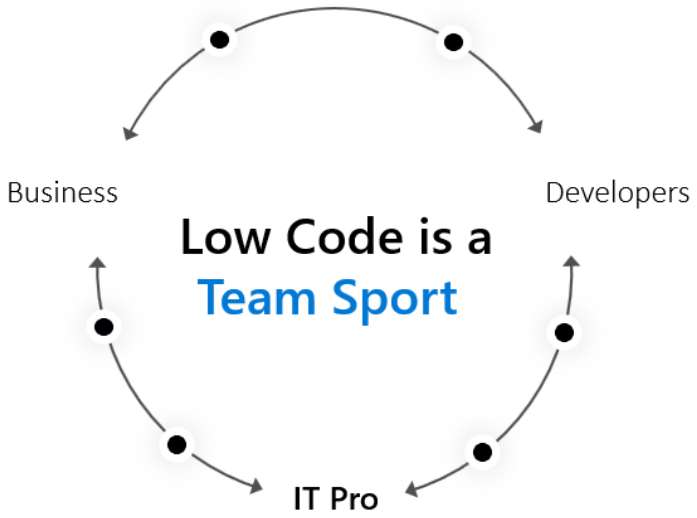
Regulated industries and organizations with extensive governance, risk and compliance practices tend to build a Center of Excellence first approach to Power Platform. Alignment to regulatory compliance is a technical requirement which usually starts with "locking everything down" at the outset of a new technology implementation. Due to underlying compliance such as HIPPA, FERPA, GDPR, it is uncommon to see a citizen development deployment model to meet these regulatory requirements and generally organizations with these types of considerations first deploy IT sponsored and staffed Power Platform implementations.

Before an organization chooses a deployment model it must evaluate the cultural viability of any deployment model. Some organizations have multiple siloed lines of business supported by a centralized IT department; these organizations tend to struggle with fusion development deployment models. Some organizations have multiple tenants and no central IT department; these organizations struggle with centralized deployment models. Some organizations have little or no IT department; these organizations struggle with both fusion and centralized deployment models but can often do decentralized citizen development.

The sophistication required to build a "Fusion" or a "Center of Excellence" power platform models is often a people and process challenge as opposed to a technical challenge. When Power Platform deployments fail, it's often because of poor orchestration of resources as opposed to technical deficiency of the services. The three delivery models outlined in the remainder of this document are outlines of the most popular deployment models organizations are executing in enterprise organizations.

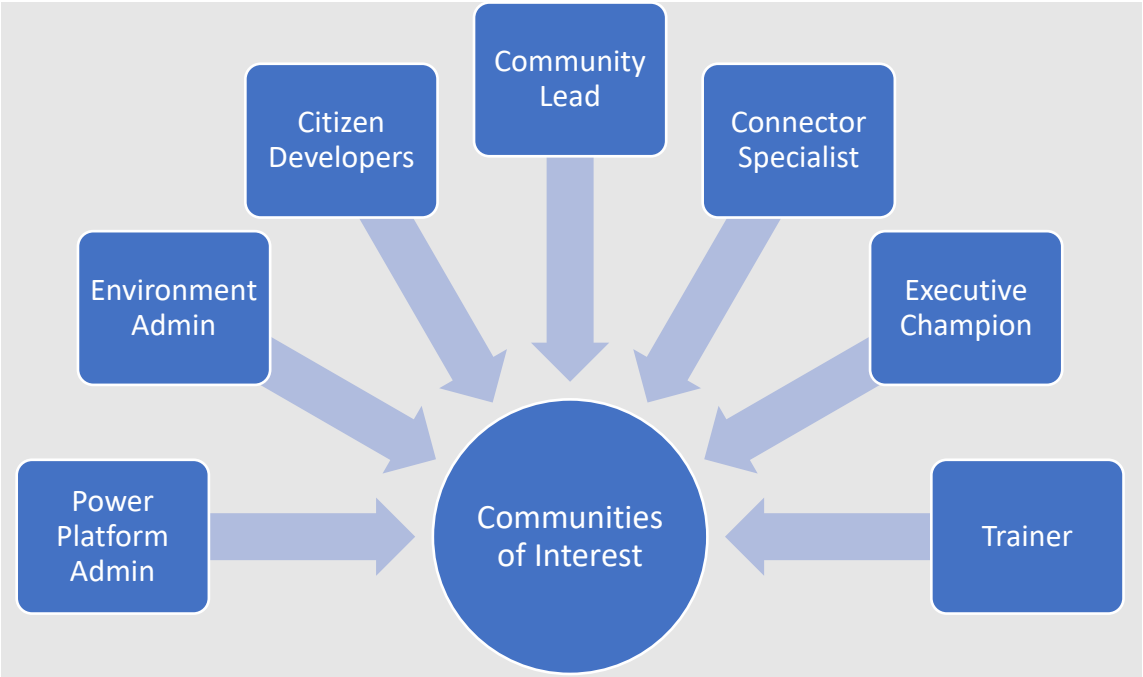
All these delivery models outlined below require collaboration, process orchestration and various talent matrixes. All these models benefit by having executive sponsorship and an underlying acknowledgement that nurturing maturity takes time, talent and investment. There are benefits and challenges to executing on any deployment model, but creating teams of talented individuals and having a baseline project charter is often a preliminary step to begin deploying any Power Platform service.

Effective deployments require people and a company culture that invests in the success of people using the services. Building successful Power Platform cultures often requires nurturing investment with trainings, hackathons, adoption, certifications, time for hands on prototyping and surrounding that nurturing with process, administration, and program management. If these types of cultural nurturing models aren't built and executive sponsorship is lacking, then Power Platform tends to have low adoption/utilization regardless of whether the technologies can meet an organizational use case technically.



Communities of Interest: A Citizen Developer deployment model

This methodology involves establishing governance around the default environment and enabling citizen developers to build productive applications with as little IT involvement and ownership as possible. Existing Office 365 services are enabled and users within the Office 365 tenant are given the ability to automate the tasks they would normally be doing across Outlook, Teams, SharePoint, One Drive, as well as any other connectors that the Power Platform administers determine are valuable to the business. Aside from baseline IT governance of users and connectors, deeper administration of a multi-environment scale is not required unless migrating to another methodology. In this approach users are encouraged to talk to one another, build a process to share information across business groups.



This approach does not require significant investment in internal staffing or consulting partners or augmented staffing. This methodology thrives best when communities of interest are fostered culturally, and organizations develop internal training programs to ramp users up to speed.

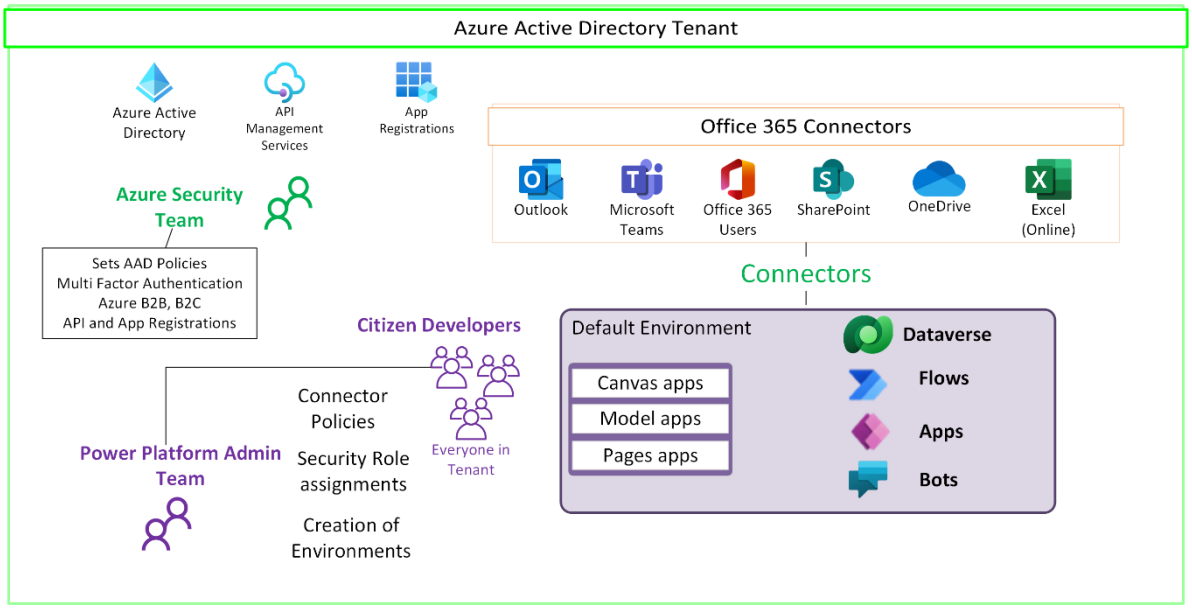
Benefits

- Business users enabled by citizen development are not constrained by the need for IT resources
- Lines of business work with relative autonomy to solve challenges they already understand
- Users autonomously gain more experience and technical depth
- Business scales out faster by itself and usually at lower cost
- Affords productivity and collaboration solutions for organizations with decentralized business models (multi-tenancy, orgs with highly siloed business units)
- No significant increase in IT Budget

Challenges

- Less mitigation of "Shadow IT"
- Unclear ownership when technology requires extensibility/support
- Lack of administrative center

- Sprawl of applications that can be orphaned when people leave or change roles
- Most difficult to regain control of application sprawl
- More challenging for IT-spend visibility and establishing charge back models



Center of Excellence: An information-security and IT resourced deployment model

This methodology is the most restrictive model that affords the most control. Essentially nothing is enabled in the default environment unless approved by an IT staffed Center of Excellence Team or explicitly enabled by Microsoft. More complex projects that are identified are mapped to a Dev, Test, Production environment topology which is created for one or more business units. This model is staffed by professional full-time resources that specialize in Power Platform services and usually staffed by an IT department.
Establishing this model requires either significant internal staffing or hiring consulting partners and augmented staffing. Complex business challenges are solved faster than conventional custom development where professional developers have learned how to use Power Platform services in depth. Professional development is leveraged where configurable development cannot meet business requirements. This methodology thrives best when service aligned groups in the COE develop depth on the Power Platform tools and collaborate with one another.

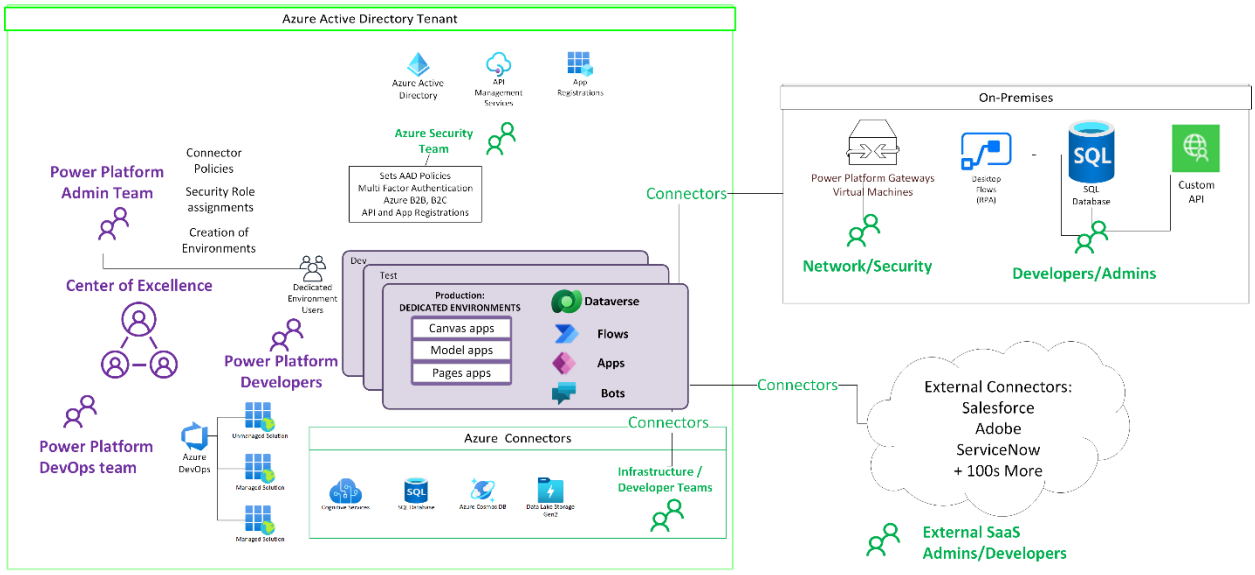
Benefits

- Comprehensive monitoring, detection, alert of Power Platform Components.

- Governance of environment and application proliferation.
- Greatest mitigation of “Shadow IT”.
- Information Security First approval of all Power Platform components
- Strong development resources building applications faster
- Robust tooling and process maturity which prevents orphaned apps
- Tightest control of application sprawl
- Less challenging visibility for IT-spend and establishing charge-back models

Challenges

- Development Bottleneck
- Significant IT resource alignment in terms of time and talent
- Onerous request/approval process for business enablement
- Not always possible due to organizational structure (consider multi-tenancy, franchise models, or organizations constantly merging/acquiring companies and tenants)
- Slows or disables Citizen Development
- Segmented or isolated Center Team that is unaware of business needs, or too busy to address them (a major problem citizen development is supposed to alleviate)
- Business can only scale out as fast as IT resources can support and usually at higher cost
- Possibly significantly increased IT Budget



Microsoft

Fusion Development: Professional Developer + Citizen Developer Collaboration

Fusion Development embraces the concepts of Citizen Development and Professional Development in a process paradigm where Citizen Development is allowed to thrive. There is also a monitoring process paradigm where critical line of business applications are identified and they are migrated into professional development solutions and environments. Fusion development requires a Center of Excellence staffed team to consistently collaborate and monitor Citizen Development solutions to identify high value tools for enterprise deployment. Consider this methodology one that combines the benefits of both models, but also combined investment and often a change in operating models.

Establishing this model would require the highest degree of internal staffing which may or may not be supported by partners or augmented staffing. Simple and complex problems are addressed in parallel and aligned to the best resources for rapid application development with no-code, low-code and pro-code. Fusion development enables the most people in an organization to solve problems they are equipped to solve but is the most challenging methodology to build. It requires change management, as well as the greatest investment of nurturing resources: time and talent.

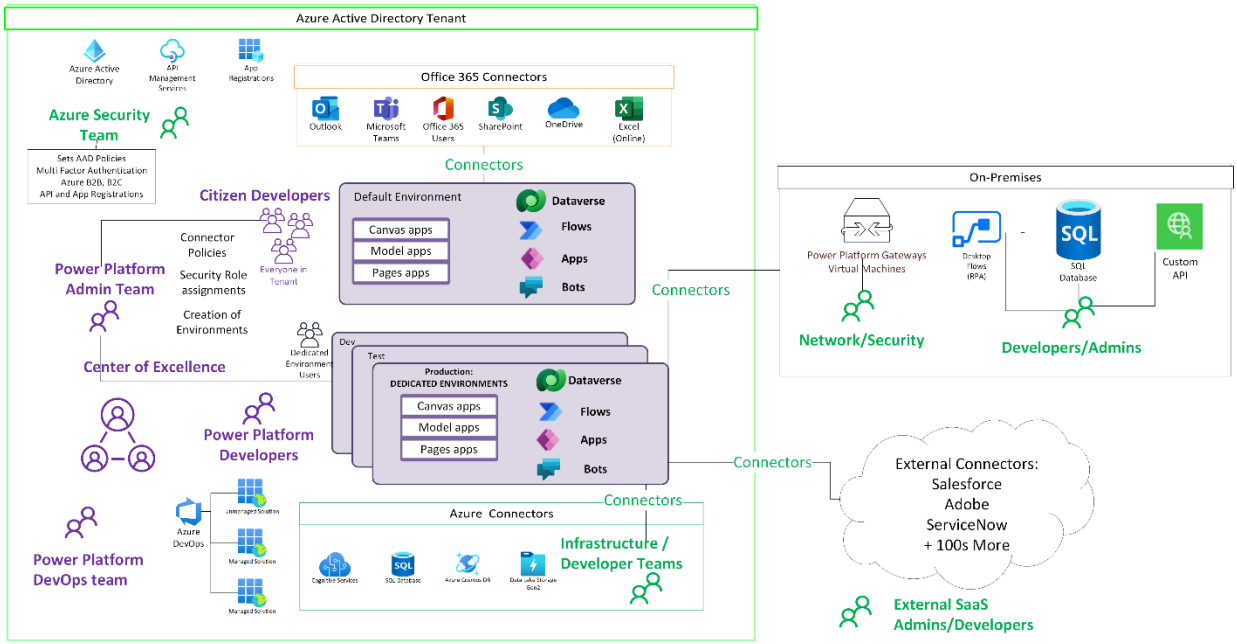
Benefits

- Citizen development thrives but with support and guard rails
- Business is enabled to solve problems for itself
- IT supports the most critical line of business applications
- Collectively more people are solving more problems just at different levels of complexity
- Enterprise level visibility of all low-code and pro-code application development
- Organically breaks down siloed business units: Business units becomes more IT savvy, IT becomes more business savvy
- Done well and done at scale is measurably the largest Return on Investment strategy for low code platforms
- Ownership is well defined across organization

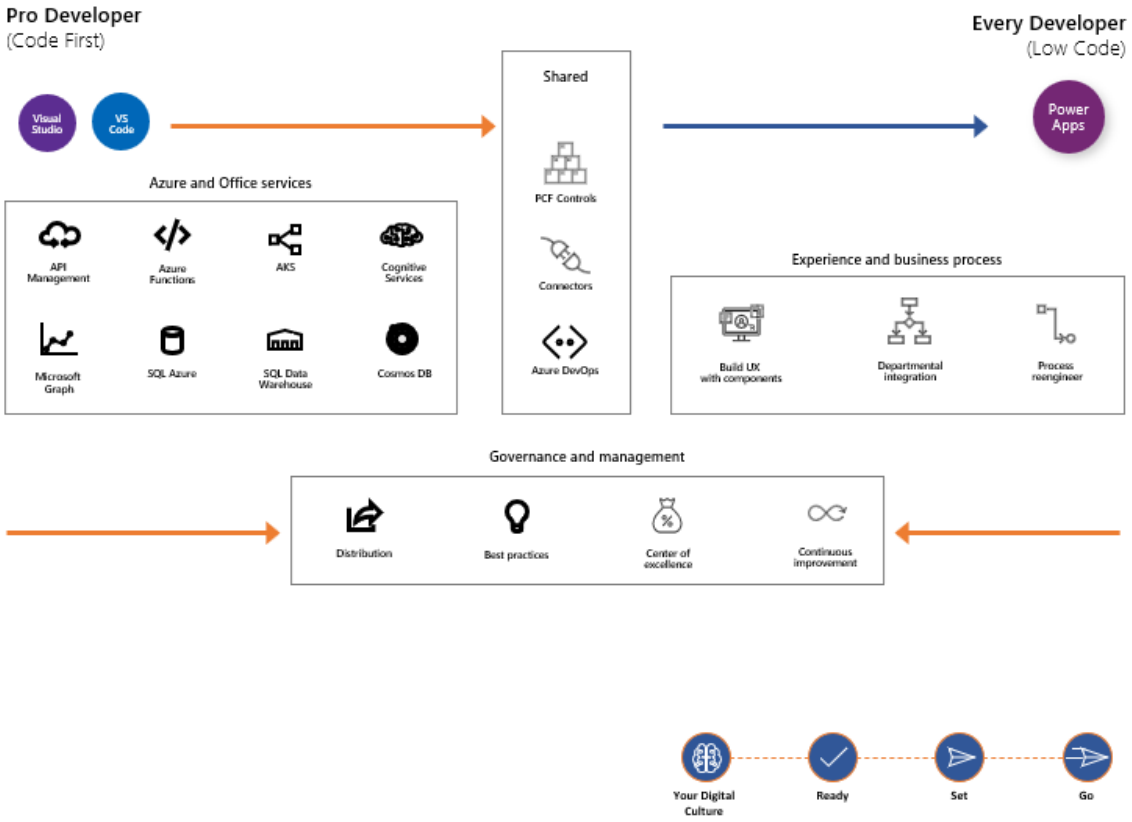
Challenges

- Orchestration and collaboration complexity between IT and Business (particularly for siloed organizations)
- Administrative overhead monitoring, alerting, protecting of applications

- Requires acumen/methodology to bridge the gap between business and IT to act as one
- Requires most training and adoption to foster competency across entire organization
- High level of IT investment
- Identifying "critical apps" requires technical resources to allocate time, more business analysis
- Often a large change management initiative to change behaviors
- Complex responsibility assignment RACI matrix



Power Platform solutions can be applied to business challenges of varying levels of complexity. When complexity merits IT involvement, it is usually when data complexity merits IT involvement. Fortunately, the Microsoft Power Platform resides in the Microsoft stack with deep synergies with extensible Azure, Office 365 and Dynamics 365 services. Where low-code development ends professional development begins and when the professional developers pick up low-code tools, productivity ensues.



Get started with Power Platform

This document has outlined the various considerations that technical decision makers should be evaluating when deploying Power Platform services within an enterprise organization. Power Platform services can be developed at every level of technical maturity, but at scale they can become a full-time technical commitment requiring significant IT sponsorship. Organizations endeavoring to scale Power Platform should develop a baseline environment strategy, DLP policy and choose a deployment model which aligns best to organizational structure, culture, and appetite to invest time and talent.

Where scaling sophistication or data complexity are beyond citizen development skills, the Power Platform is a profound set of rapid development services that can be extended by full-time IT professionals. Sign up to try Power Apps, Power Automate, Power BI, or Power Virtual Agents for free.

Microsoft

The remainder of this document is a vast set of resources organized by Power Platform service to help organizations mature on specific deployment models and skill across services.

Start your Center of Excellence (CoE)

- What is a Center of Excellence - Power Platform | Microsoft Docs
- Find your Executive Sponsor and your Champions
- Establishing an Environment Strategy
- Establishing a DLP strategy
- Educate your COE how to use Solution concepts
- Ramp your team up with tailored Power Platform Learning Paths
- Learn more about Power Platform Certifications and Exams
- Attend Power Platform - Events
- Read the latest Power Platform Release Notes
- Read how organizations are Transforming their business with Fusion Development

Power Apps learning paths

Model-Driven Power Apps Skills

- Database Modeling
- Configuring Dataverse (tables, columns, forms, fields, views, charts, relationships)
- Power Platform Environment Administration (for multiple environments)
- Managing Solutions
- User Security Management
- Environment Management (for individual environments)
- Programmatic Extensibility
- Model Driven Application Development Experience
- Programmatic experience with Dynamics 365 Customer Engagement (Sales, Service, Field Service)
- Model Driven App API development
- Leveraging Power Platform DevOps tools

- Source Control with Solution Files
- Leveraging GitHub
- Prior Experience with XRM Toolbox

Power Pages Skills

Power Apps Pages

- Get Started with Power Apps portals - Learn | Microsoft Docs
- Work with Power Apps portals - Learn | Microsoft Docs
- Administer Power Apps portals - Learn | Microsoft Docs

Portals security + authentication

- Power Apps portal: Security 101 - Microsoft Dynamics CRM Community
- Overview of authentication in Power Apps portals - Power Apps | Microsoft Docs
- Create web roles for portals - Power Apps | Microsoft Docs
- Manage page permissions - Power Apps | Microsoft Docs
- Configure security using table permissions - Power Apps | Microsoft Docs

Portal Migration

- Migrate portal configuration - Power Apps | Microsoft Docs

Power Portal Professional Development Skills: Liquid + web development

- Work with Liquid template language in Power Apps portals - Learn | Microsoft Docs
- Understand Liquid operators - Power Apps | Microsoft Docs
- Build custom Power Apps portals web templates - Learn | Microsoft Docs
- Extend Power Apps portals - Learn | Microsoft Docs

Power Automate Learning Paths

- Learning Portal

- Power Automate documentation - Power Automate | Microsoft Docs
- Working with Custom Connectors
- Power Automate for enterprise developers, ISVs, and partners
- Let customers test drive your flows on AppSource - Power Automate | Microsoft Docs
- Power Automate documentation - Power Automate | Microsoft Docs

Power Automate Skills

- Intro to Power Automate
- How to build an automated solution
- Connection references
- Power Automate limits
- List of actions
- Using wait conditions
- Data operations
- Error handling
- Approval flows
- Data Loss Prevention (DLP) policies

Power Virtual Agent Learning links

Beginner Learning Paths

- Introduction to Power Virtual Agents
- How to build a basic chatbot
- Enhance Power Virtual Agents bots
- Get started with Power Virtual Agents bots
- Manage topics in Power Virtual Agents
- Create apps, chatbots, flows, and more with Microsoft Dataverse and Teams
- Create a chatbot with Power Virtual Agents and Dataverse for Teams

Microsoft

Intermediate

- Create bots with Power Virtual Agents
- Manage Power Virtual Agents
- Proactive Messaging in Power Automate RPA and Power Virtual Agents processes
- Implementing a Power Virtual Agents Fallback Topic
- Using Global Variables in Power Virtual Agents
- Logging Salesforce Cases Using Power Virtual Agents and Power Automate
- Parsing Dates with Power Virtual Agents and Power Automate
- Connecting Power Virtual Agents to ServiceNow using Power Automate

Advanced

- Solution Architect series: Power Platform architecture
- Solution Architect: Design Microsoft Power Platform solutions
- Solution architect series: Explore Power Virtual Agents
- Solution Architect series: Testing and go-live
- Solution Architect series: Evaluate Power Platform analytics and AI
- Custom Validation in Power Virtual Agents using Regular Expressions
- Adding Thumbnail Cards to Power Virtual Agents/Bot Composer Framework Chatbots

Power Virtual Agent Best Practices

- Power Virtual Agents quick start guide | Ebook
- Power Virtual Agents whitepaper | Download
- Power Virtual Agents best practices | Download | Blog
- Building responsible bots | Guidance
- Community posts | Murali Kumanduri | Dhina Gajavarathan

Microsoft

Power BI Skills

- Data Sources
- on-premises data gateway
- Power BI Desktop
- Power BI Service
- Power BI for business users Power BI Administration Power BI APIs for Developers,
- Developing custom visuals.
- Power BI Desktop for individuals
- Enterprise Report Creation
- Data Analysis Expression (DAX)
- Azure ETL
- Data Flow ETL
- Virtual Machine Gateway Infrastructure,
- AI or Machine Learning capabilities,
- Python
- R Models Workspaces
- Row-level security (RLS)
- Power BI adoption documentation Power BI Maturity
- Power BI white papers
- Power BI adoption documentation
- Adoption and maturity roadmaps

Microsoft

Glossary of terms

Citizen developer- A citizen developer is a professional in anything other than traditional code or app-making. They are empowered by low code/no code platforms, such as Power Platform, to solve business problems through custom apps, automation, bots and analytics. A citizen developer continues their current job while dabbling in solving business problems using these low code/no code platforms. Sometimes citizen developers become full-time app and automation specialists known as makers.

Maker- A maker is someone who often works in business or IT and solves business problems using low code/no code tools such as Power Platform. Their explicit job is using these tools to build apps, automation, bots and analytics to be consumed by stakeholders in their organization.

Professional developer- A professional developer traditionally solves business problem by creating custom line of business apps using high code tools such as C# code. In low code/no code platforms such as Power Platform, a professional developer still uses custom code but in tandem with no code/low code options.

Fusion development- Fusion development is the creation of line of business apps, automation, bots, and analytics by a team comprised of citizen developers, makers and professional code developers. A mature Fusion team is successful in supporting all these roles while maintaining the needed data security/governance using platform tools such as DLP policies, Azure Active Directory and proper training.