Microsoft 365

ALM

Application Lifecycle Management Best Practices in Power Platform and Dynamics 365

Effective strategies for managing software development lifecycles

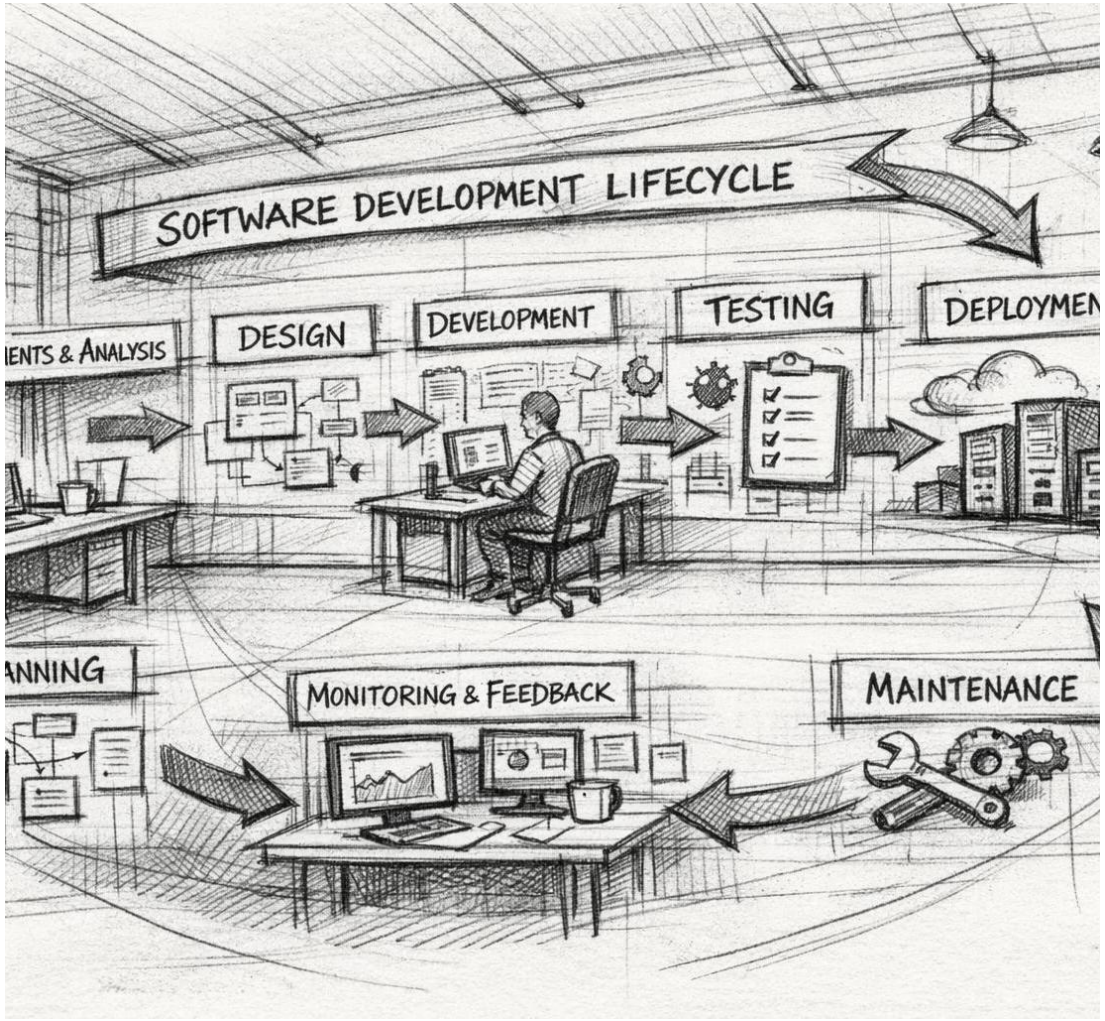
Presenter:
Sabrina Di Bartolomeo – Sr FastTrack Solutions Architect

Today's Agenda Overview

- Foundations of ALM
- Benefits of ALM
- Environment Strategy and Governance
- Solution Strategy and Best Practices
- Source Code Integration + Demo
- Deployment Tools
- Q&A

Foundations of Application Lifecycle Management in Microsoft Environments

Key concepts and principles of ALM



Application Lifecycle Management (ALM)

- spans the entire project lifecycle, not just development and deployment
- integrates processes and tools to manage software from conception to retirement effectively
- must be embedded in the overall project methodology, not only as a technical activity
- is a continuous, adaptable process that should evolve with business and project needs

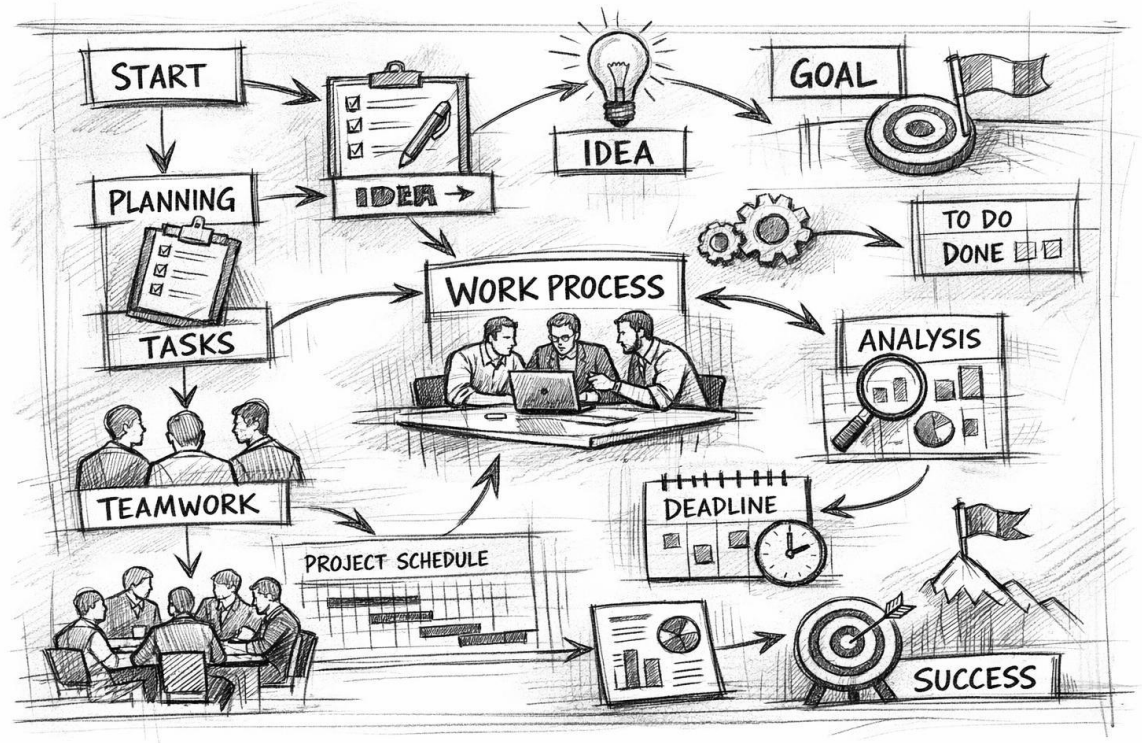
The right ALM approach depends on various factors, such as:

- *Environment Strategy*: number of dev/test environments, shared vs isolated development
- *App Landscape*: single app or multiple apps (Sales, Customer Service, Field Service, Finance, etc.)
- *Team & Skills*: team size, DevOps maturity, customer readiness to manage the platform
- *Release Strategy*: sprint-based releases, hotfixes vs feature releases

There is no one-size-fits-all ALM strategy

Benefits of ALM

Why is ALM important?



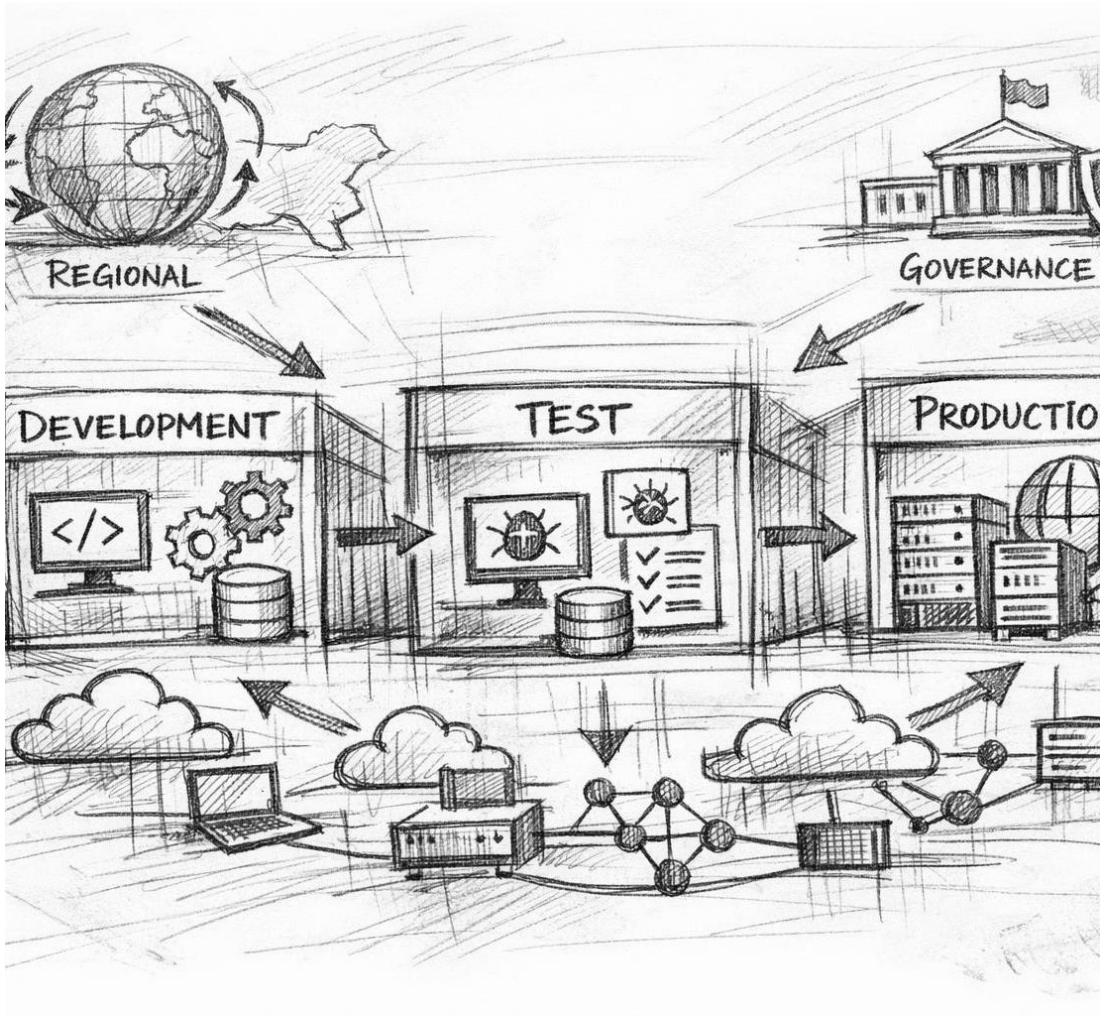
Common challenges solved by following a healthy ALM:

- "It was working in TEST, why it is not working in PROD?"
- "Something was deployed by mistake. How can it be rolled back?"
- "Someone changed something directly in PROD and now it's not working anymore, what has been changed?"
- "I need to delete unused fields and tables, and I don't want to do it manually in all environments"
- "I need to recreate my development environment in a specific point in time"

https://docs.microsoft.com/en-us/power-platform/alm

Environment Strategy and Governance

Choosing the right Environment Strategy



Environment Types

Identify the number of environments needed such as development, test, training, data migration, and production.

Regional Deployment

Determine the geographic regions to host your environments for optimal performance and compliance.

Environment Governance

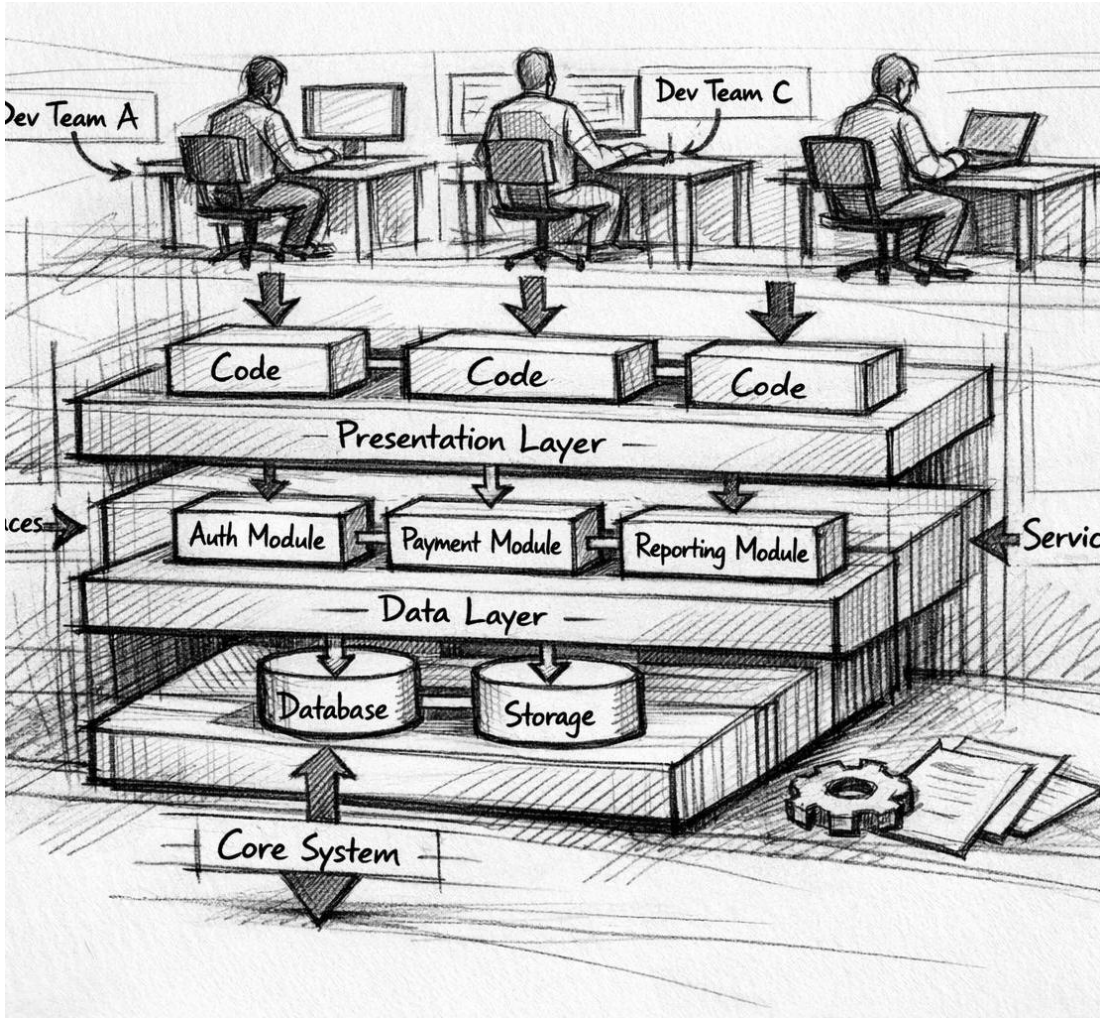
Define rules and policies to govern environment usage, access, and lifecycle management.

Within Managed Environments:

- *Environment Routing*: automatically directs makers/developers to the correct development environment instead going to the default environment.

Solution Strategy and Best Practices

Solution Strategy



Plan before creating solutions

- Solution design should start before development, not at deployment time
- Early planning avoids dependency issues, deployment risks, and rework later in the lifecycle

Single Solution Strategy

All customizations are grouped into one unmanaged solution during development, which is later exported as a single managed solution for deployment.

Recommended for:

- Small-medium scale implementations
- Scenarios where future modularization is unlikely

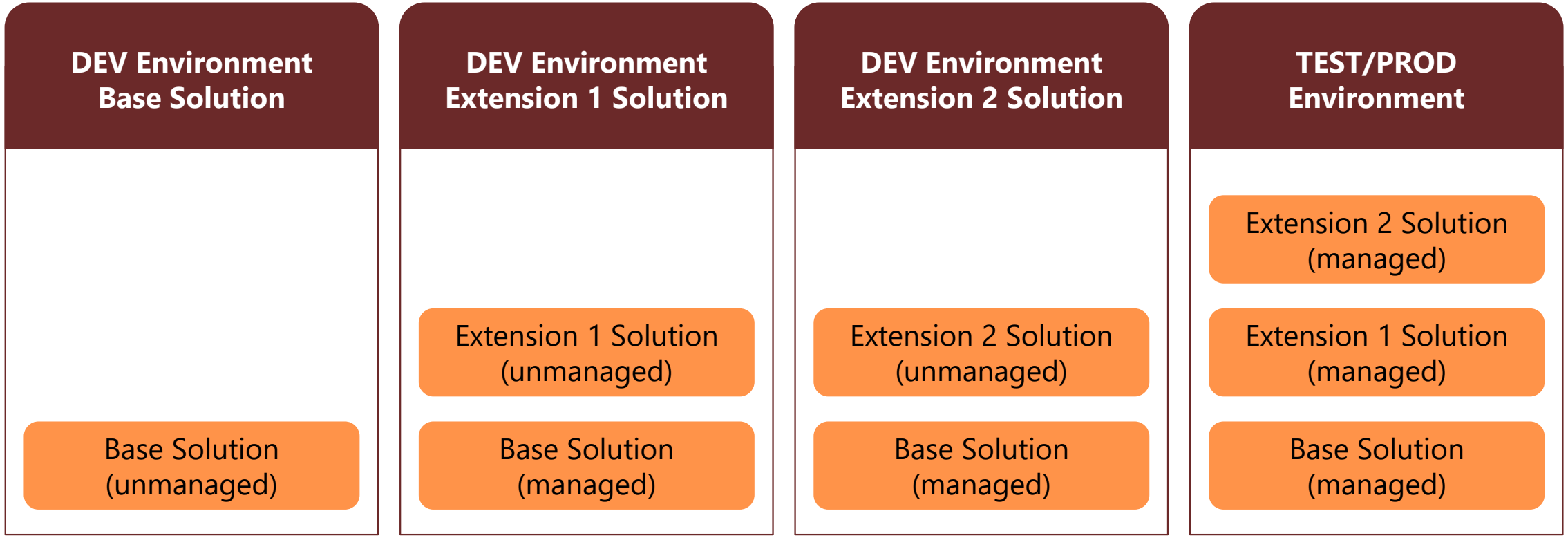
Multiple solutions with dedicated development environments

This strategy is commonly used in modular architectures where, for example, different applications—such as Sales, Customer Service, or Field Service—are built and maintained independently. A base solution containing common components (for example, account and contact tables) is created and deployed as a managed solution into each app-specific development environment. Each app then has its own unmanaged solution, layered on top of the base managed solution, allowing teams to extend functionality without altering the base foundation.

Recommended for:

- Large-scale enterprise projects
- Teams with multiple developers or partners
- Scenarios requiring strict governance and CI/CD pipelines

Multiple solutions with dedicated development environments



PUBLISHER

Can be associated with **multiple solutions across dev** environments

Defines the **customizations prefix** and **base option** set values

When importing new assets through a managed solution, the associated publisher **OWNS** those assets

Several **default publishers** are created with a new environment. **DO NOT USE OR MODIFY THEM**

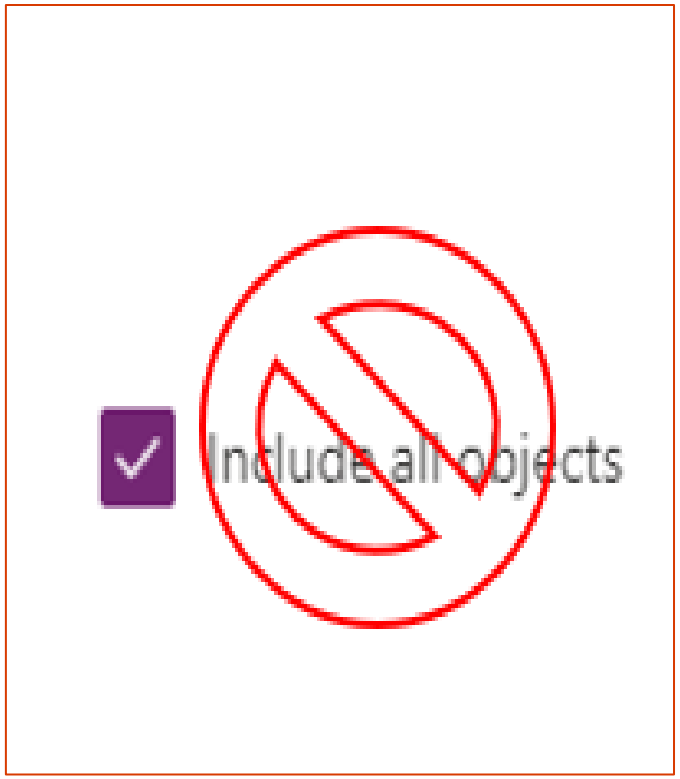
Create a **new custom publisher** for the Project

Use the **same publisher across all dev** environments, **unless there is a clear reason not to**

https://docs.microsoft.com/en-us/power-platform/alm/solution-concepts-alm#solution-publisher

https://learn.microsoft.com/en-us/power-platform/alm/managed-properties-alm

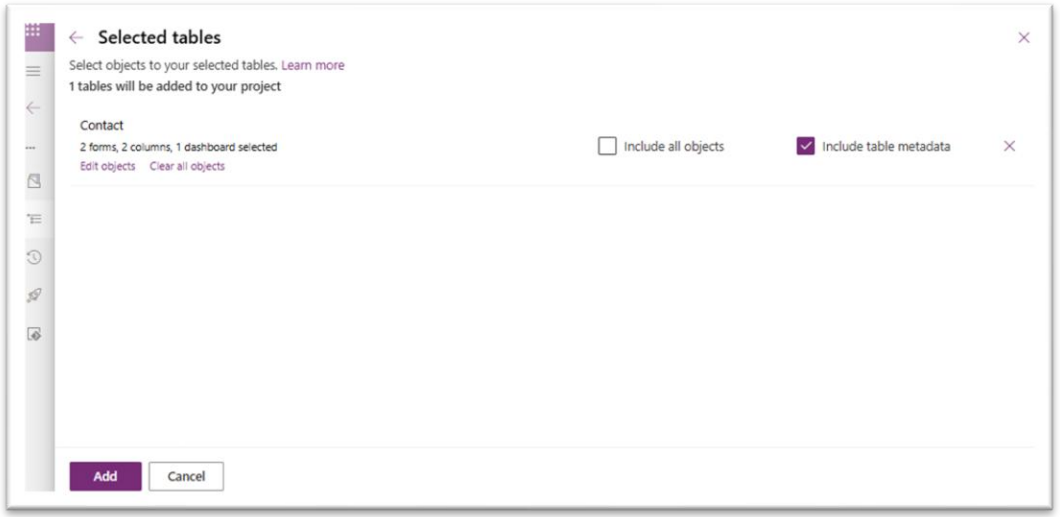
TABLE SEGMENTATION



- **Include** in a solution ONLY **components** being **modified** or newly **created**.
- AVOID **Include all objects** when adding Microsoft tables or managed tables for further customizations.
- When adding a table, note that the system now automatically preselects unmanaged components and components with unmanaged customizations. Use **Select objects** to review and adjust the preselection before choosing Add to complete the process.
- The **Preferred Solution** feature helps to ensure that only created or changed components are added by default.

https://learn.microsoft.com/en-us/power-platform/alm/segmented-solutions-alm

INCLUDE TABLE METADATA



“Include table metadata” will add the table settings section to the solution

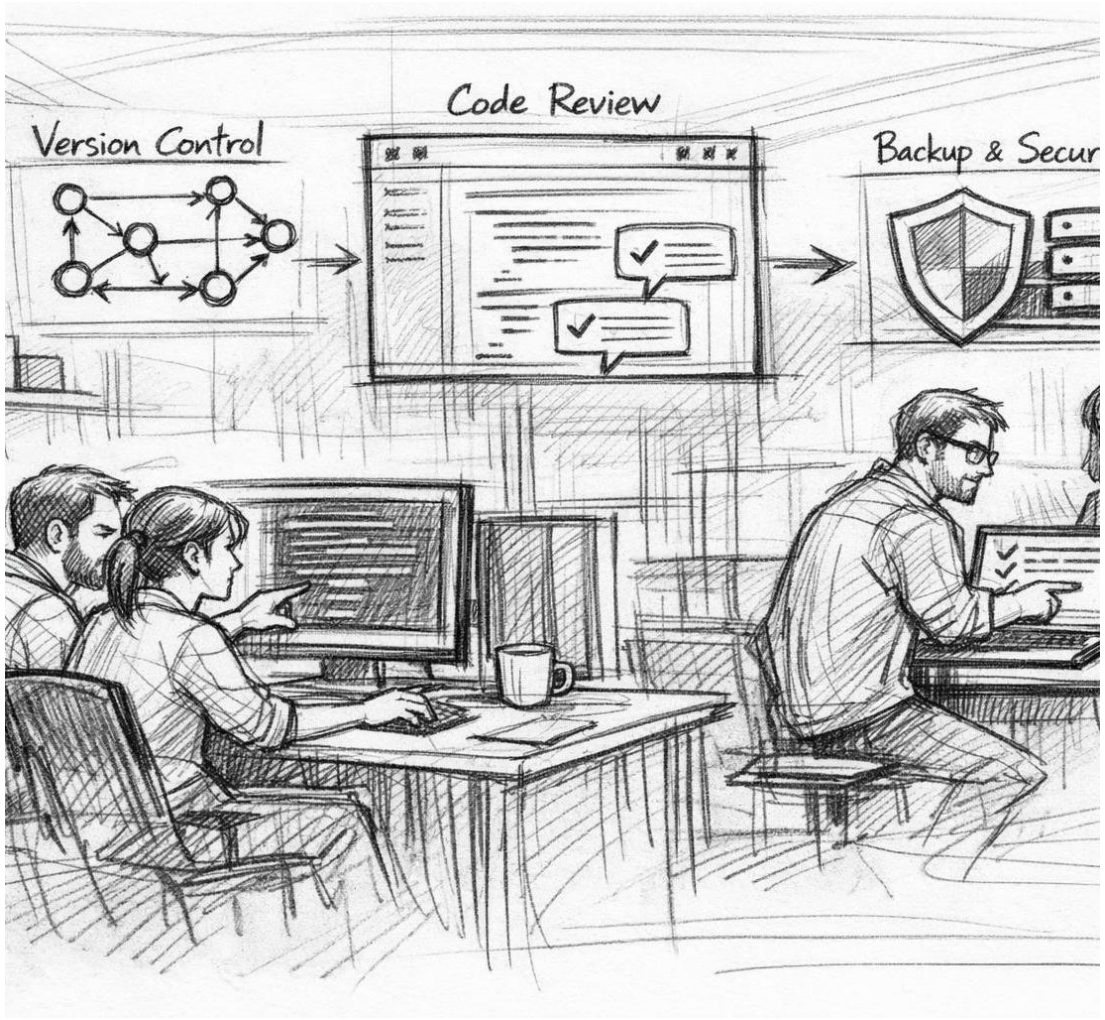
```
<Name LocalizedName="Work Order" OriginalName="Work Order">msdyn_workorder</Name>
<EntityInfo>
<entity Name="msdyn_workorder">
<LocalizedNames>
...</LocalizedNames>
<LocalizedCollectionNames>
...</LocalizedCollectionNames>
<Descriptions>
...</Descriptions>
<attributes>
...</attributes>
<EntitySetName>msdyn_workorders</EntitySetName>
<IsDuplicateCheckSupported>1</IsDuplicateCheckSupported>
<IsBusinessProcessEnabled>1</IsBusinessProcessEnabled>
<IsRequiredOffline>0</IsRequiredOffline>
<IsInteractionCentricEnabled>0</IsInteractionCentricEnabled>
<IsCollaboration>1</IsCollaboration>
<AutoRouteToOwnerQueue>0</AutoRouteToOwnerQueue>
<IsConnectionsEnabled>1</IsConnectionsEnabled>
<EntityColor>#660033</EntityColor>
<IsDocumentManagementEnabled>1</IsDocumentManagementEnabled>
<AutoCreateAccessTeams>0</AutoCreateAccessTeams>
<IsOneNoteIntegrationEnabled>0</IsOneNoteIntegrationEnabled>
<IsKnowledgeManagementEnabled>1</IsKnowledgeManagementEnabled>
<IsSLAEnabled>1</IsSLAEnabled>
<IsDocumentRecommendationsEnabled>0</IsDocumentRecommendationsEnabled>
<IsBPFEntity>0</IsBPFEntity>
<OwnershipTypeMask>UserOwned</OwnershipTypeMask>
<EntityMask>ActivityPointer</EntityMask>
<IsAuditEnabled>1</IsAuditEnabled>
<IsRetrieveAuditEnabled>0</IsRetrieveAuditEnabled>
<IsRetrieveMultipleAuditEnabled>0</IsRetrieveMultipleAuditEnabled>
<IsActivity>0</IsActivity>
<ActivityTypeMask>CommunicationActivity</ActivityTypeMask>
<IsActivityParty>0</IsActivityParty>
<IsReplicated>0</IsReplicated>
<IsReplicationUserFiltered>0</IsReplicationUserFiltered>
<IsMailMergeEnabled>1</IsMailMergeEnabled>
<IsVisibleInMobile>0</IsVisibleInMobile>
<IsVisibleInMobileClient>1</IsVisibleInMobileClient>
<IsReadOnlyInMobileClient>0</IsReadOnlyInMobileClient>
<IsOfflineInMobileClient>1</IsOfflineInMobileClient>
<DaysSinceRecordLastModified>0</DaysSinceRecordLastModified>
<MobileOfflineFilters><fetch version="1.0" output-format="xml-platform" mapping="logical" distinct="false"><entity name="msdyn_workorder"></entity></fetch></MobileOfflineFilters>
<IsMapiGridEnabled>1</IsMapiGridEnabled>
<IsReadingPaneEnabled>0</IsReadingPaneEnabled>
<IsQuickCreateEnabled>1</IsQuickCreateEnabled>
<SyncToExternalSearchIndex>1</SyncToExternalSearchIndex>
<IntroducedVersion>3.1.0.3</IntroducedVersion>
<IconMediumName>msdyn_/Icons/Entity/WorkOrder_32/_imgs/NavBar/Invisible.gif</IconMediumName>
<IconSmallName>msdyn_/Icons/Entity/WorkOrder_16.png</IconSmallName>
<IconVectorName>msdyn_/Icons/SVG/WorkOrders.svg</IconVectorName>
<EnforceStateTransitions>0</EnforceStateTransitions>
<EntityHelpUrlEnabled>1</EntityHelpUrlEnabled>
<EntityHelpUrl>http://go.microsoft.com/fwlink/?LinkId=773453</EntityHelpUrl>
<ChangeTrackingEnabled>1</ChangeTrackingEnabled>
<IsEnabledForExternalChannels>0</IsEnabledForExternalChannels>
<IsMSTeamsIntegrationEnabled>0</IsMSTeamsIntegrationEnabled>
<IsSolutionAware>0</IsSolutionAware>
<HasRelatedNotes>True</HasRelatedNotes>
<HasRelatedActivities>True</HasRelatedActivities>
</entity>
</EntityInfo>
```

Do’s and Don’ts

- **Avoid** creating a **separate solution for every small change**, such as each sprint. Where possible, keep a **single solution**.
- Dedicate a Development environment per unmanaged solutions and **avoid multiple unmanaged solutions in the same environment if they have dependencies**.
- Establish a new (and **only 1**) **custom publisher** for the project. Don’t use the default ones.
- All development should happen **inside solutions**. Components created outside solutions should be avoided. For testing or experimentation, use a dedicated playground environment. Using the **Preferred Solution** helps keep development consistent.
- Avoid adding unchanged Microsoft managed components. Only include **components** that are being **changed**. Avoid **Include all objects** when adding Microsoft tables to solutions.
- Use **unmanaged** solutions only in **development** environments.
- **Patches** are **not recommended** anymore.
- Utilize **solution checker** to validate solutions.

Source Code Integration and Branching Strategy

Benefits of Source Code Integration



Version History and Traceability

Track every change to easily identify who made modifications and revert to previous versions when needed.

Collaboration Among Developers

Allows multiple developers to work simultaneously with streamlined management of updates and conflict resolution.

Code Quality and Review

Supports pull requests and code reviews to improve quality and catch issues before integration.

Backup and Disaster Recovery

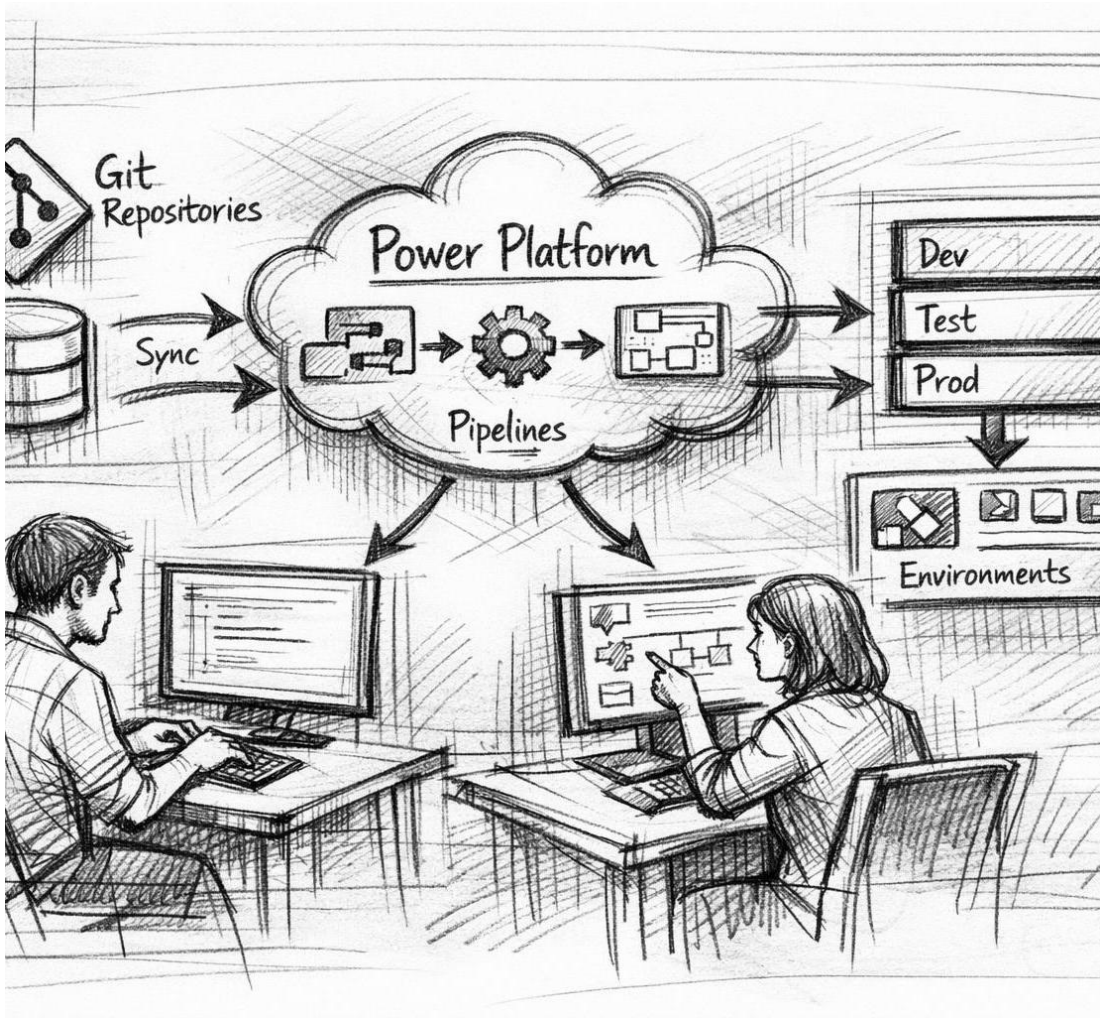
Ensures safe backup of solutions protecting against accidental data loss or corruption.

Audit Trail and Compliance

Maintaining a history of changes is crucial for auditing and compliance. Source control provides a clear record of all modifications, which is especially important in regulated environments.

Keeping Power Platform solutions in source control is a core best practice for professional teams.

Native Source Control Integration



Native Git Integration

Connect to Git repositories in seconds without extra tools, enabling fusion development across environments. Solutions stored in source control come from unmanaged solutions in a DEV environment.

Platform and Custom Hosts

Platform Host is tenant-wide and automatic; Custom Host supports enterprise governance and multiple pipelines.

Connecting to Git

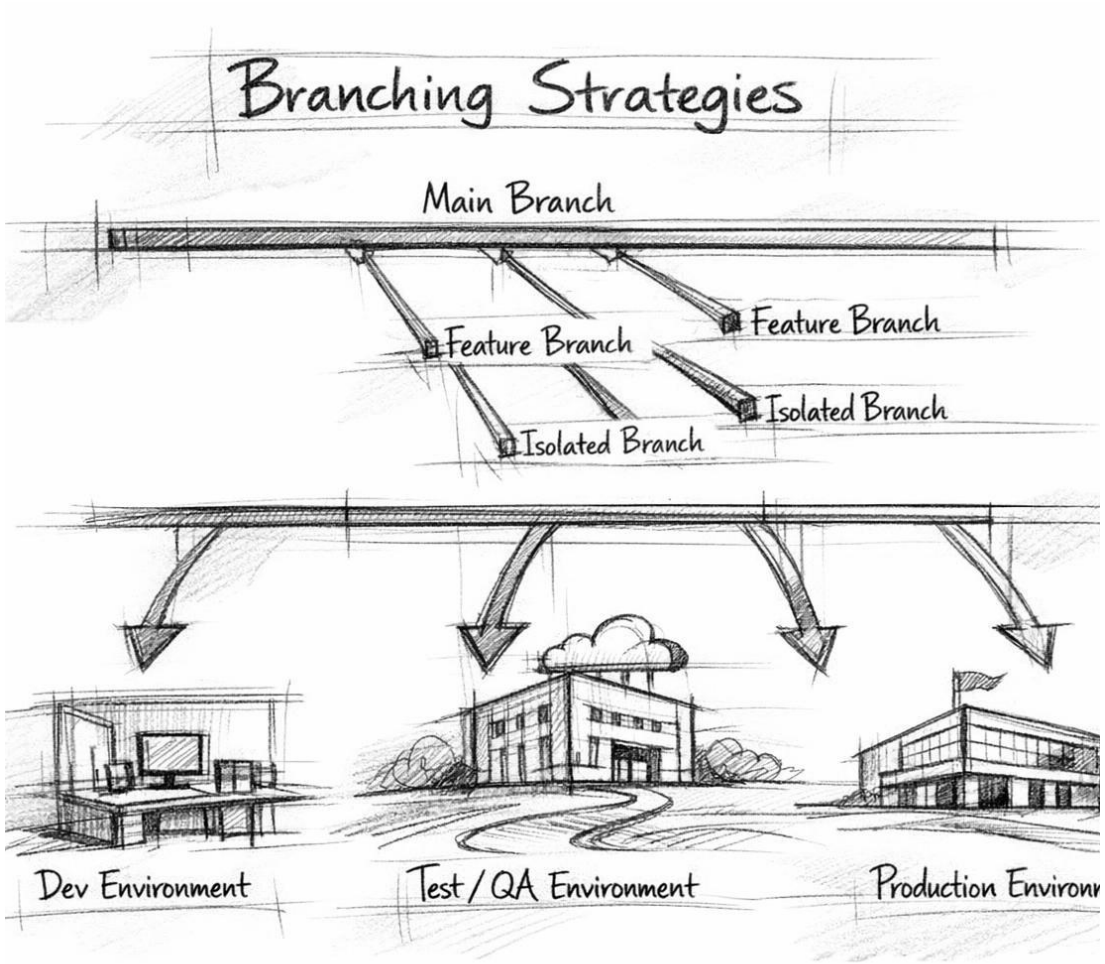
Users connect by selecting solutions, then choosing Environment or Solution mode to enter Git details.

File formatting for solution objects

After committing and pushing changes to source control, the solution objects are stored in a YAML format. This format is easy to read and understand and can be used to track and merge changes to the solution objects.

https://learn.microsoft.com/en-us/power-platform/alm/git-integration/overview

Branching Strategies for ALM



Single Branch Strategy

A simple approach with no additional branches, syncing source code directly in a single development environment.

Feature Branch Strategy

Uses a stable main branch and separate feature branches for isolated development and merging after completion.

Multiple Developer Branches

Developers work in isolated branches and environments, frequently syncing with main to avoid conflicts.

Hotfix Developer Branches

A dedicated separate Hotfix branch can be used for isolated bug fixing.

ProjectA

Overview
Boards
Repos
Files
Commits
Pushes
Branches
Tags
Pull requests
Advanced Security
Pipelines
Test Plans
Artifacts

Branches

New branch

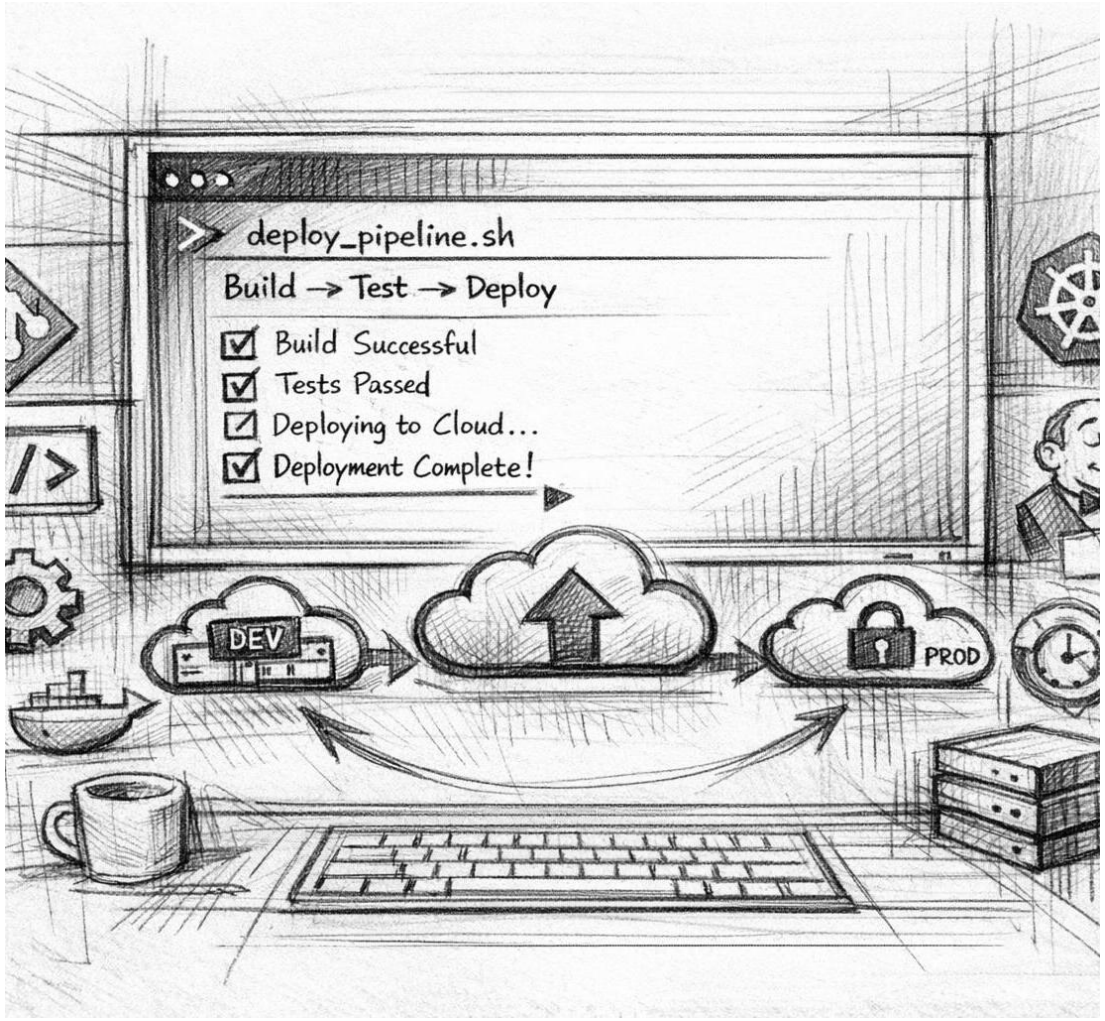
Mine All Stale

Search branch name

Branch	Commit	Author	Authored Date	Behind | Ahead	Status	Pull Request
main Default Compare	e424f953	Sabrina Di Bartolomeo	1h ago			

Deployment Tools

Overview of Deployment Tools



Pipelines in Power Platform

This is the native deployment tool built directly into Power Platform. Pipelines significantly reduce the effort and domain knowledge previously required to adopt healthy, automated ALM processes.

Power Platform CLI

Is a command-line tool used to export, unpack, pack, and import solutions. The Power Platform CLI performs various operations related to Environment lifecycle, Authentication, Solution packages and more.

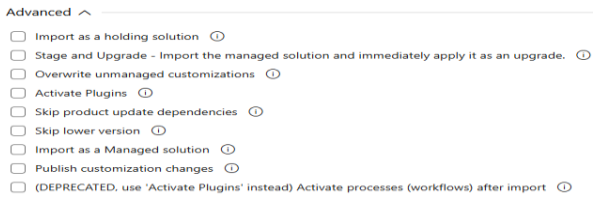
Build Tools for DevOps/GitHub Actions

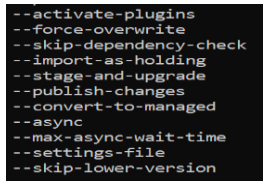
Build Tools Tasks and GitHub Actions are built on top of the Power Platform CLI. These tools are intended as building blocks to configure a custom ALM process to meet an organization's needs.

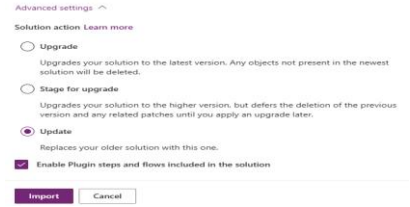
Custom Pipelines

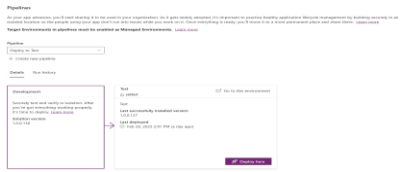
Custom scripts, custom yaml pipelines, or extended pipelines built on top of CLI, APIs. It allows full customization for complex requirements such as advanced branching strategies, environment routing, tenant-specific logic, or integration with legacy systems.

Solution import options





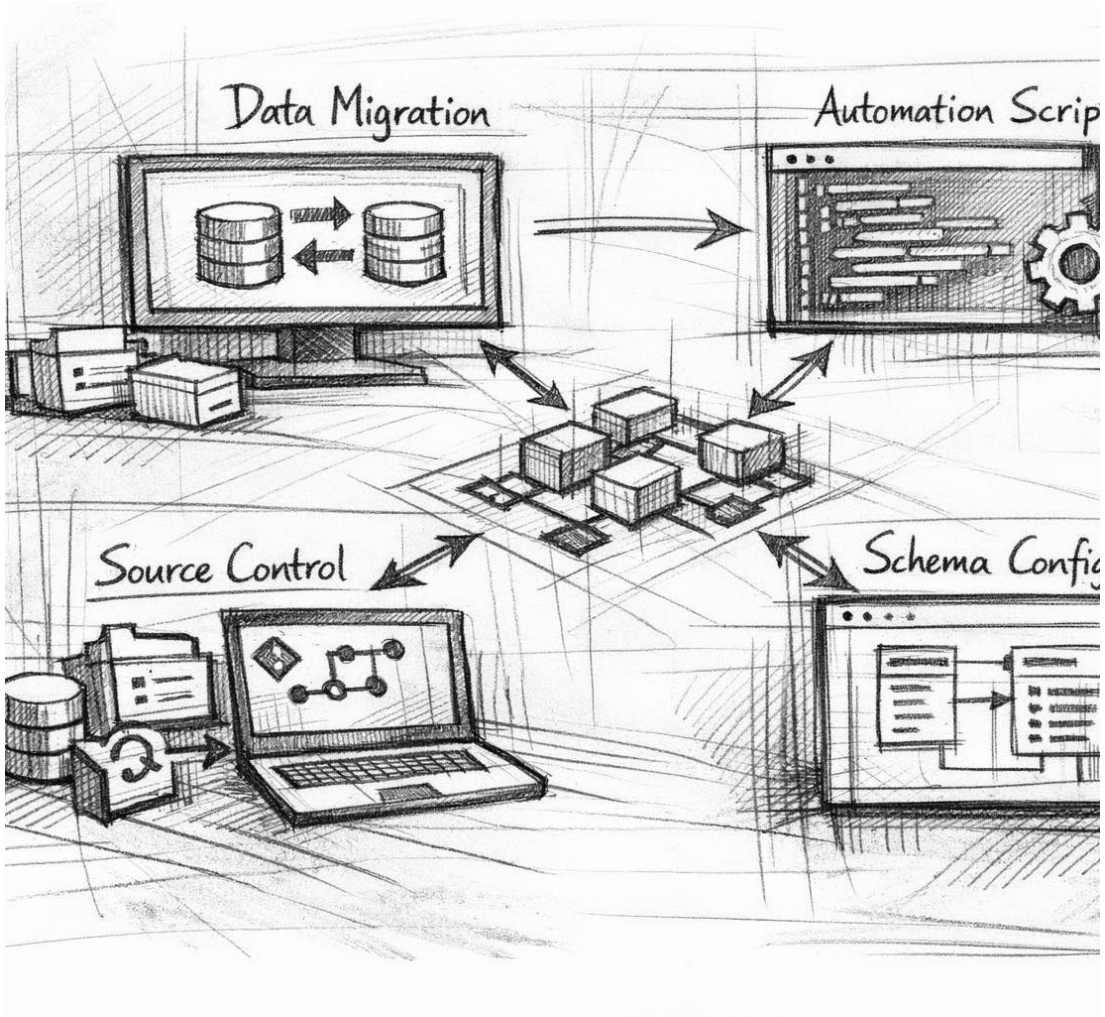




PP Build Tools	PP CLI	UI	Pipelines in PP
Import as a holding solution	--import-as-holding	Stage for upgrade	*Not available*
Stage and Upgrade	--stage-and-upgrade	Upgrade	*Default behavior*
Overwrite unmanaged customizations	--force-overwrite	*Not available*	*Not available*
Activate Plugins	--activate-plugins	Enable plugin steps and flows included in the solution	*Default behavior*
Skip product update dependencies	--skip-dependency-check	*Not available*	*Not available*
Skip lower version	--skip-lower-version	*Default behavior*	Allow redeployments of older versions
Import as a Managed solution	**--convert-to-managed**	**Not available**	**Not available**
Publish customizations changes	*Not available*	*Not available*	*Not available*
No option selected	*No option selected*	Update	*Not available*

https://learn.microsoft.com/en-us/power-platform/alm/performance-recommendations

Configuration Migration Tool



Purpose and Functionality

The tool transports configuration and test data across environments, capturing configuration for source control and testing automation.

Automation and Integration

Supports automation via PowerShell and integrates with source control, enabling usage in build tools and pipelines.

Schema and Data Management

Allows schema definition, selective entity and field export, filters, import, logging, and uniqueness conditions to prevent duplicates.

Reusability and Usage Benefits

Enables reuse of existing schemas for data exports.

Documentation

- Microsoft Learn ALM Documentation
- Application lifecycle management for the Power Platform Training
- Business Process Catalog
- Environment version mismatch warning during solution import
- Managed Environments
- Set the preferred solution
- Source control integration
- Organize Solutions
- Block unmanaged customizations in Dataverse
- Solution Performance recommendations
- Configuration Migration tool

Q&A