ALM and DevOps Practices

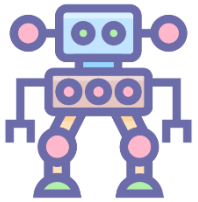
for

Building on the Common Data Service

Colin Vermander

Director of Software Engineering

KPMG Canada Microsoft Practice



Agenda

- ALM and DevOps for CDS
- Solution Fundamentals
- Solution Packaging and Configuration Data
- Team Development
- Demos
- Questions

Understanding ALM and DevOps for CDS

For the Common Data Service

Application Lifecycle Management (ALM)

"..an integrated system of people, processes, and tools that manage the life an application from concept to retirement."
- **Stackify.com**

- Solutions
- Configuration Data
- Source Control

DevOps

"the union of people, process, and products to enable continuous delivery of value to our end users"
- **Donovan Brown, Microsoft Principal DevOps Manager**

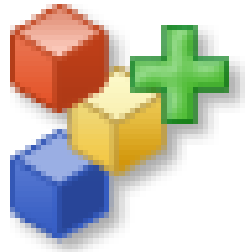
- Packaging
- Deployment
- Automation

ALM for CDS

Application Lifecycle Management for Common Data Service

Solution Fundamentals - what are solutions?

- A mechanism to author, package and maintain functionality
- Contain a series of components that define the metadata:
  - Schema (entities, attributes, relationships)
  - User Interface (forms, views, web resources)
  - Process and Code (workflows, plugins)
  - Security (roles, profiles)
  - And more…
  - *Coming soon: Canvas apps and flows*



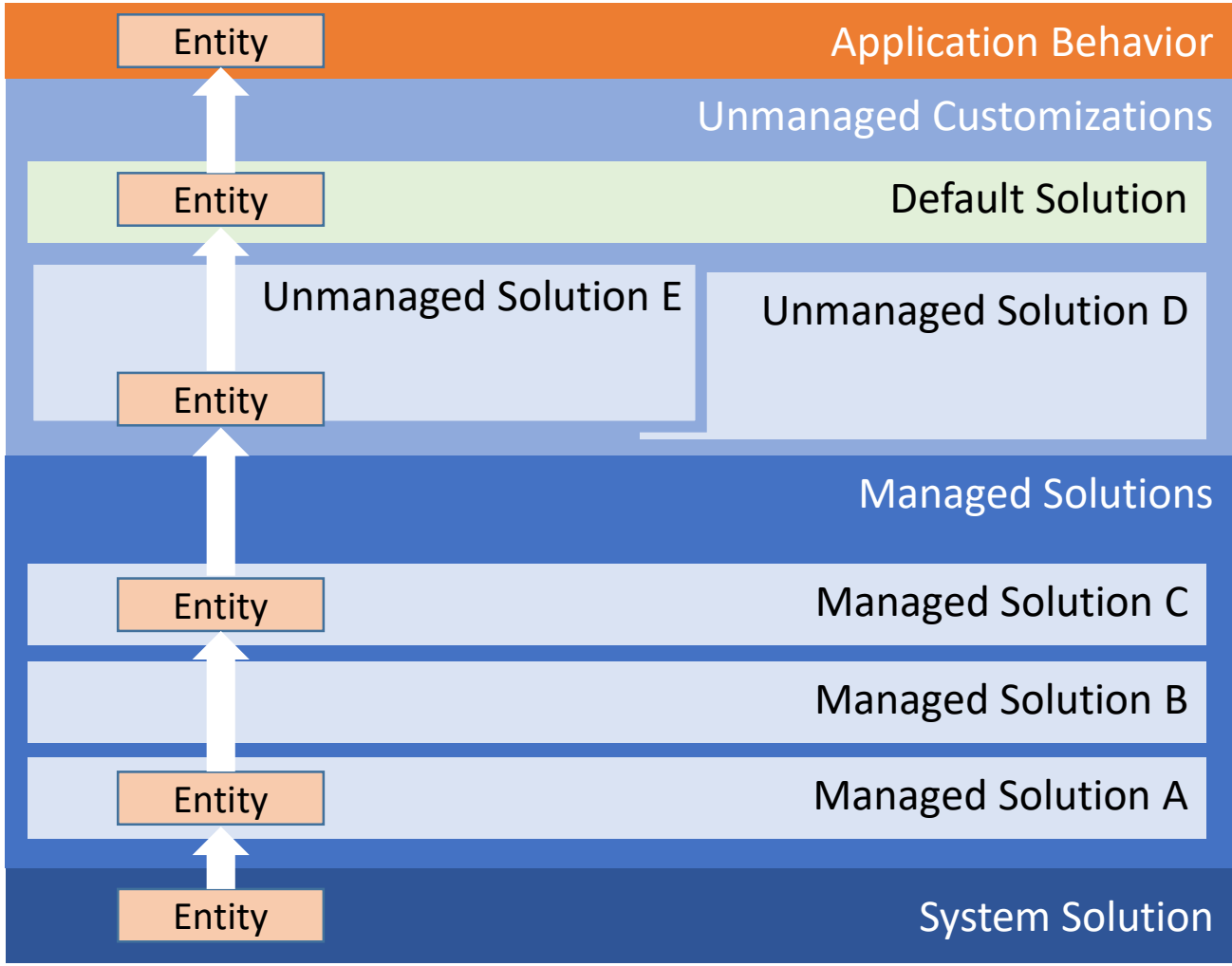
- Allow functionality to be distributed and managed through install and uninstall

Unmanaged and managed solutions

Unmanaged solution	Managed solution
• Development solution or a solution not intended for distribution • Can be exported as unmanaged and managed • Delete does not remove any components or data	• A complete solution that is intended for distribution • Cannot be exported • Cannot edit components directly • Can restrict additional unmanaged customizations • Can be updated, upgraded, and patched • Delete/Uninstall removes all components AND data • Additional controls with managed properties

Solution Layering

- Starting always with a system solution
- Managed solutions can build on each other
  - Order of installation
  - Dependencies
  - Merge
- Unmanaged layer rules
- Continuously modify the same components in different layers



Solution Actions

Update

- Replaces existing solution with new components in same layer
- Does not delete – unused components will remain in the system
- Overwrite or keep unmanaged customizations

Patch

- Create specific fixes and updates to solutions as patch's
- Additive only – no delete scenarios
- Layers on top of parent solution, but does not impact layers above

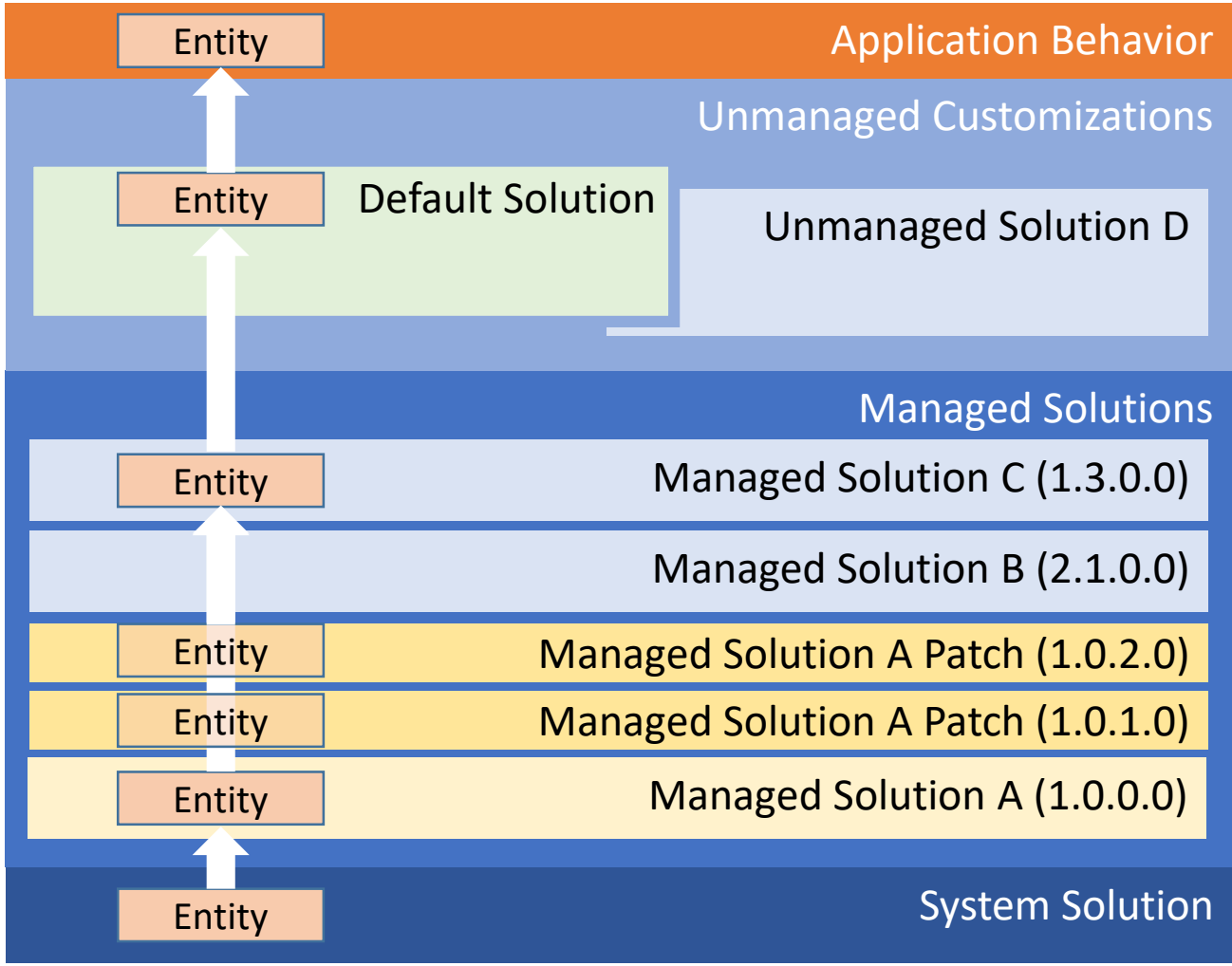
Upgrade

- Upgrade functionality
- Remove unused assets
- Run upgrade logic

Solution Patching

- Patches are installed above the base solution
- Below any managed solutions that sit on top of the parent
- Solutions that sit on top of a patch can still override
- Follow solution version guidelines

  major.minor.build.revision



When to use **managed** solutions

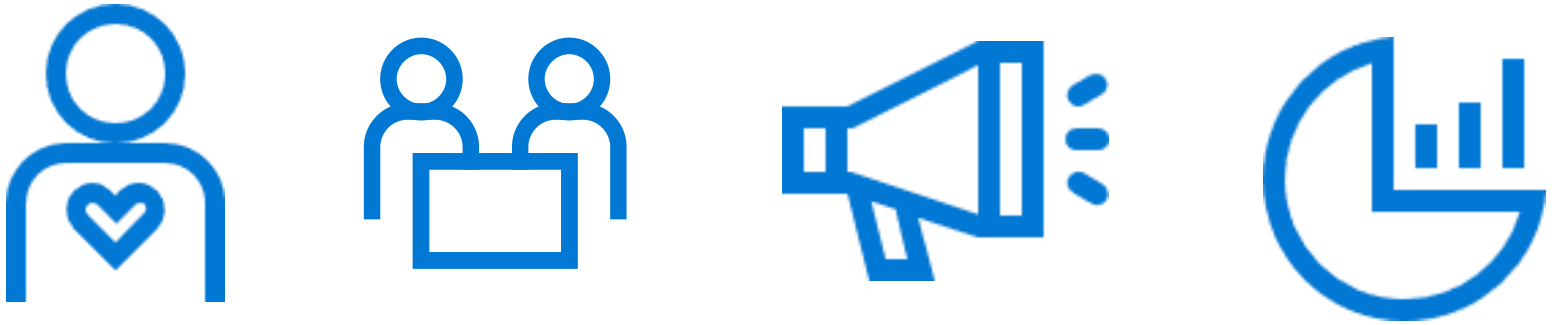
"When the unmanaged solution is complete and you want to distribute it, export it as a managed solution."

docs.microsoft.com – Introduction to Solutions

- Legacy of managed being used only by ISV's
- Guidance is to use managed solutions when leaving development in ISV products **AND** individual projects / services / solutions

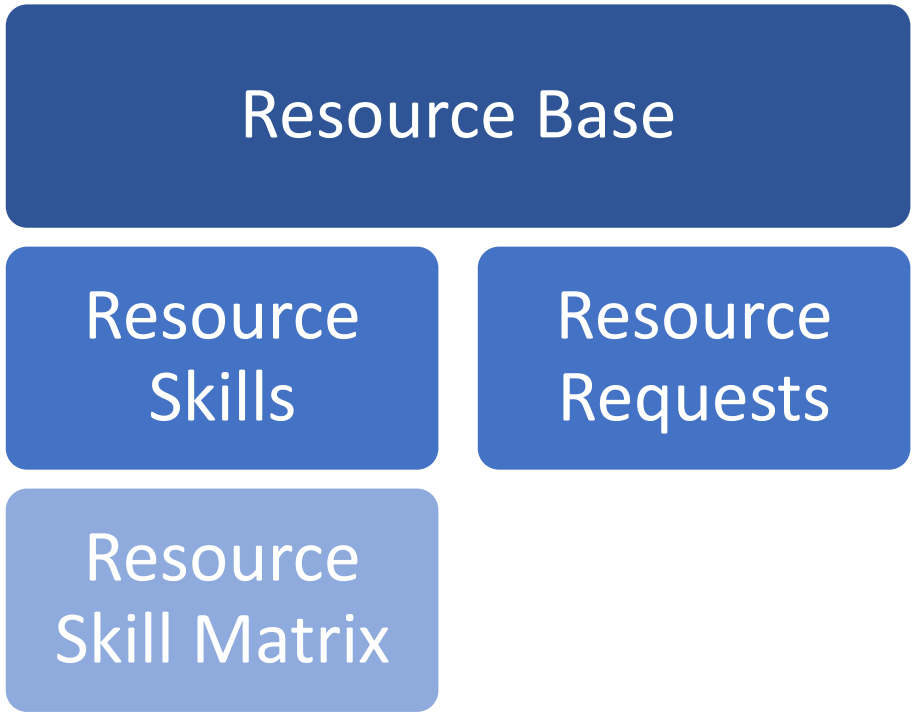
Dynamics 365 Apps Solutions

- 1st party apps are now using the same solution model as customizers
- Sales, Customer Service, Marketing, etc. are all managed solutions
- Build on top of Common Data Service for Apps



Solution Libraries – Modularization/Layering

- Only really possible with managed solutions
- Solution components that can be shared
- Solutions Framework lets you build layers of solutions that depend on each other
- Create a dependency tree for solutions



Architect Solutions with Solution Libraries

- Create reusable functionality across projects/products
- Install functionality as required
- Easily remove functionality
- Lower complexity for testing
- Manageable automation patterns

Packing and Unpacking Solutions

- Solution file is a single binary file (compressed ZIP)
- **Solution Packager** – part of the SDK Tools

Unpacking

- Extracts/decomposes solution components to multiple files
- Components are described using XML within individual files

Packing

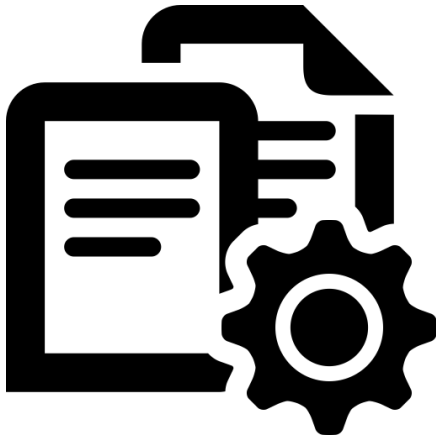
- Reassembles extracted solution XML files
- Creates an new solution archive
- Can be packed as managed or unmanaged

Solution Packager Mapping

- Map solution components to existing file resources
- Type of components that commonly follow this pattern:
  - Web Resources
  - Plugin Assemblies
- Solution component not always the source code
  - Plugin Assemblies are compiled from classes
  - Web Resources can be compiled JavaScript, CSS, or other web technologies

Configuration Data

- Data in the system maybe part of business logic and functionality
- System operation maybe dependent on data
- Examples: Unified Service Desk, Portal

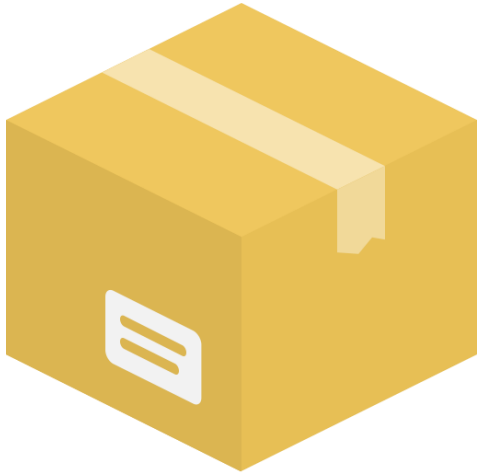


- **Configuration Migration Tool** – part of the SDK
  - Move data across environments (export and import)
  - Define a reusable schema of data (entities and fields)
  - Support for duplicate detection (define uniqueness condition per entity)
  - Disable business processes on import and re-enable upon completion
  - Not scriptable ☹ - give Microsoft feedback!

Best Practices for Configuration Data

- Define uniqueness conditions for each entity
- Extract configuration data compressed ZIP to track changes easily
  - Breakdown complex XML files for entity record level tracking
  - Run data cleansing procedure to remove unwanted data
- Validate import procedures in test/staging environments
- Backup pre-existing data

Package Deployment



- Package - custom installer for functionality
- Package may include:
  - 1 or more solution files
  - 1 or more configuration data files or flat files
  - Documentation content to show at the beginning and end of installation
  - Custom code that executes on installation events

- **Package Deployer** – part of the SDK
  - UI Interface OR PowerShell cmdlet
  - Visual Studio Project Template – for custom code logic
  - XML file to define package contents (solutions, data, USD agent executable)

Team Development

- Multiple developers working on a project or a single solution
- 3 common patterns

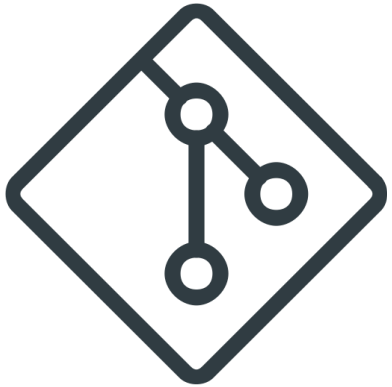
Single environment with shared solution

Single environment with developer solutions

Developer environment

Using Source Control with Solutions and Data

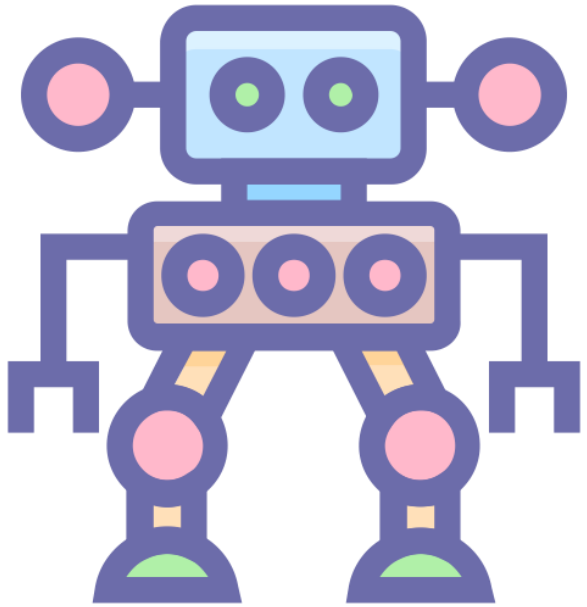
- Solutions component, business logic and configuration data needs to be managed
- Single source of truth, system of record
- Merging and change tracking
- Ability to rollback or restore old customizations
- Create a build validation, testing and release automation process



Enabling Successful Source Control for CDS

- Solution Packager
  - Allows for expansion of solution files
  - Tracking changes at a detailed level
  - Compiled with source files for business logic and components using mapping
- Expanding Configuration Data
  - Track data at the entity record level
- Branching strategy
  - Branch for environment layers
  - Branch for releases
  - Branch for developer feature development

Automate and Script the Environment



- Common Data Service for Apps and PowerApps have scriptable functions
- PowerApps PowerShell cmdlets
- Dynamics 365 Online Management API (API and PowerShell cmdlets)
- Enable developers to execute common environment functions quickly

Automate Working with Customizations

- SDK contains functions to interact with solutions (Import, Export, Publish, etc.)
- 3rd party tools implement these functions in a scriptable manner
  - Microsoft.Xrm.Data.PowerShell (PowerShell)
  - Adoxio.Dynamics.DevOps (PowerShell)
  - Sparkle/spkl (Executable)
- Provide an array of solutions to import/export
- Numerous ways to import or export data for each 3rd party tool

Development Process

- Isolated developer has a full environment to use as they wish
- Developer has the ability to do scripted updates to their own development environment
  - Plugin assemblies
  - Web Resources
- Speed up repetitive actions
- Sparkle even includes the ability to do scripted plugin registrations (PowerShell modules not as easily)

DEMO

PowerApps Environment Reset, Import, Development, Export

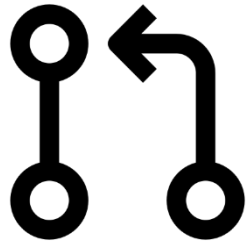
Demo Review

- PowerShell script to execute a provision/reset of a CDS environment
- PowerShell script to execute an import
- Do development, automate repetitive tasks
- Create schema and exported with Configuration Migration Tool
- PowerShell script to execute an export

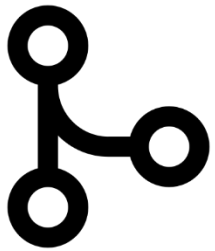
Developer Isolation Procedure

- Branch source code for feature development
- Provision isolated environment specific for feature development
- Make changes for feature development
  - Could mean creating a solution patch, cloning or just revving the solution version for a full upgrade
- Create or update configuration data schema file
- Export data with configuration migration tool
- Exports solution(s) and unpack
- Expand configuration data
- Review changes and push to source control branch
- Pull Request to parent branch (dev, master, etc.)

CDS Continuous Integration



Developer Pull Requests will merge code from other developers

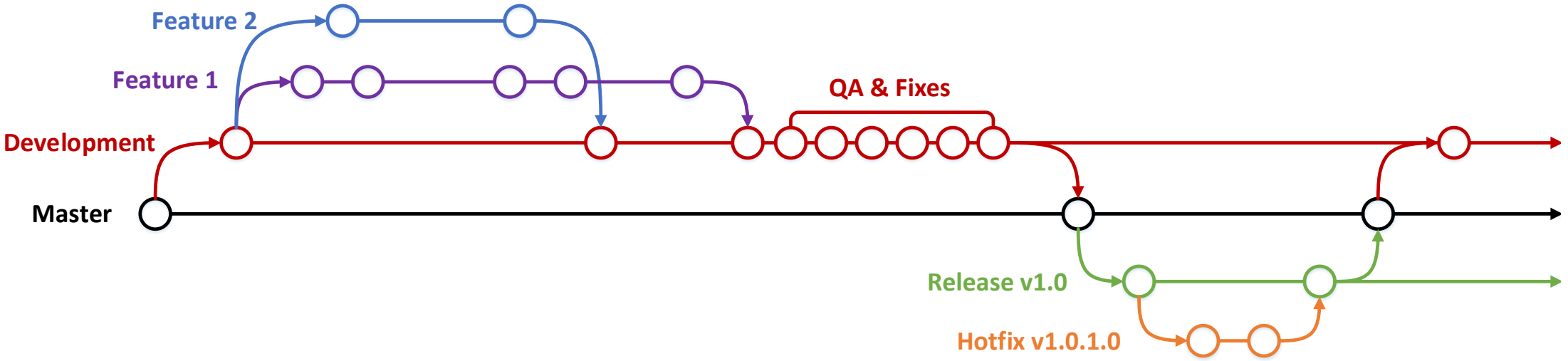


Solution merging possible due to unpacked solution files

DevOps for CDS

DevOps for Common Data Service

Branching Strategy for Development



Source Control is the Authority

Use Source Control as the Package Source

Create packages using an independent agent

Recreate any version, any component on demand

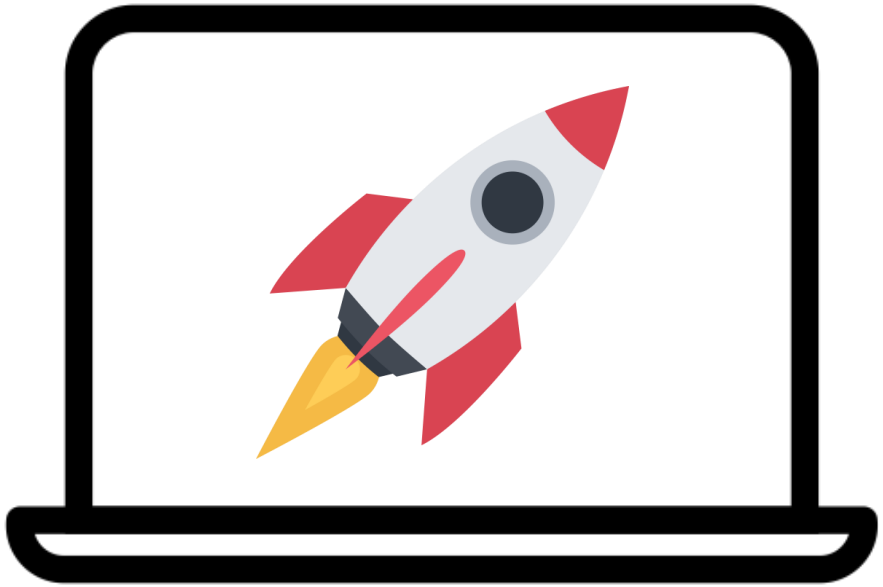
Full Automation

Build and Release Management

- Azure DevOps contains build and release pipelines
- Allows for automated validation, testing and deployment
- As developers complete pull requests and push code validate functionality
- Utilize 3rd party Marketplace Tools for Dynamics 365 that also work with CDS/PowerApps
  - Dynamics 365 Build Tools
    XRM CI Framework

DEMO

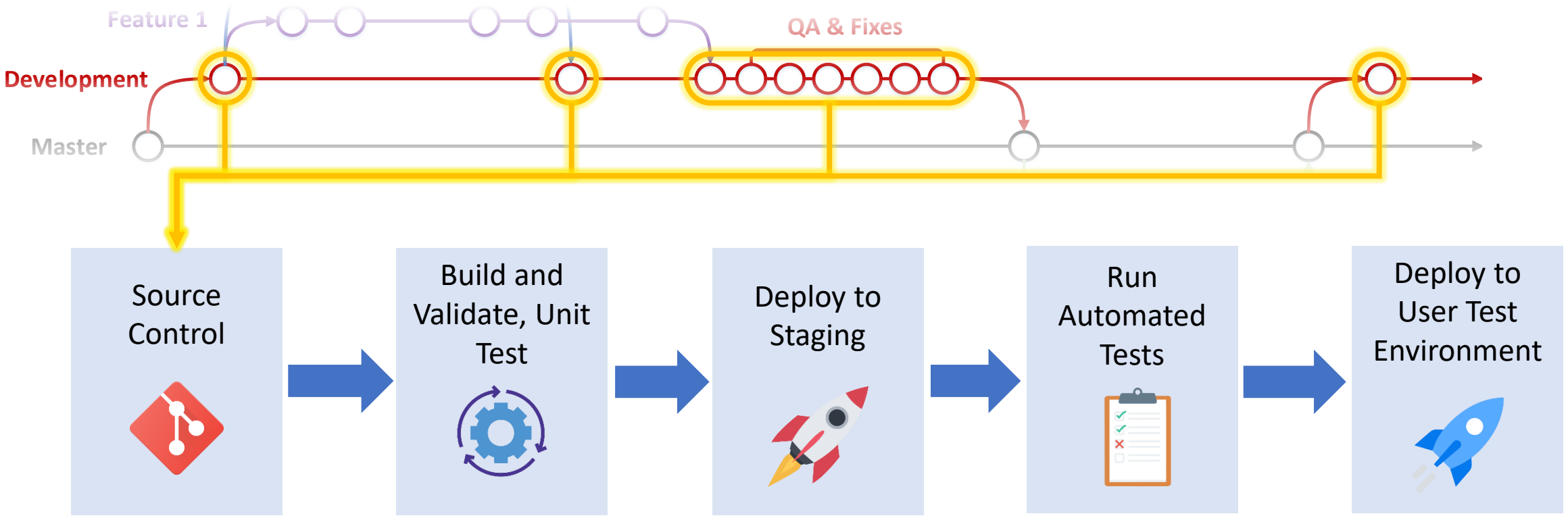
Continues Delivery with Azure DevOps Pipelines



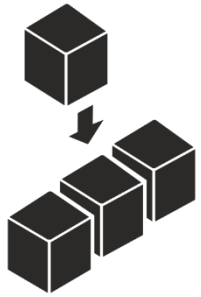
Demo Review

- Source code changes review by developer
- Build Pipeline
  - Agent recreates the build process that a local developer executes
  - Resolve library dependencies (NuGet and npm)
  - Visual Studio solution build to compile project code
  - PowerShell scripts for static Web Resources (HTML, images)
  - PowerShell script to Compress Configuration Data
  - PowerShell script to generate package
  - Publish package
- Release Pipeline
  - Get package
  - Execute package deployment

Fully Automate Your CDS DevOps Pipeline

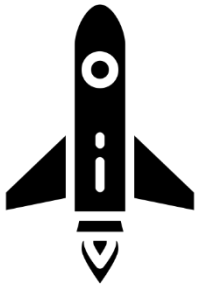


CDS Continuous Delivery



Compile solution source components, pack solutions

Package packed solutions and configuration data



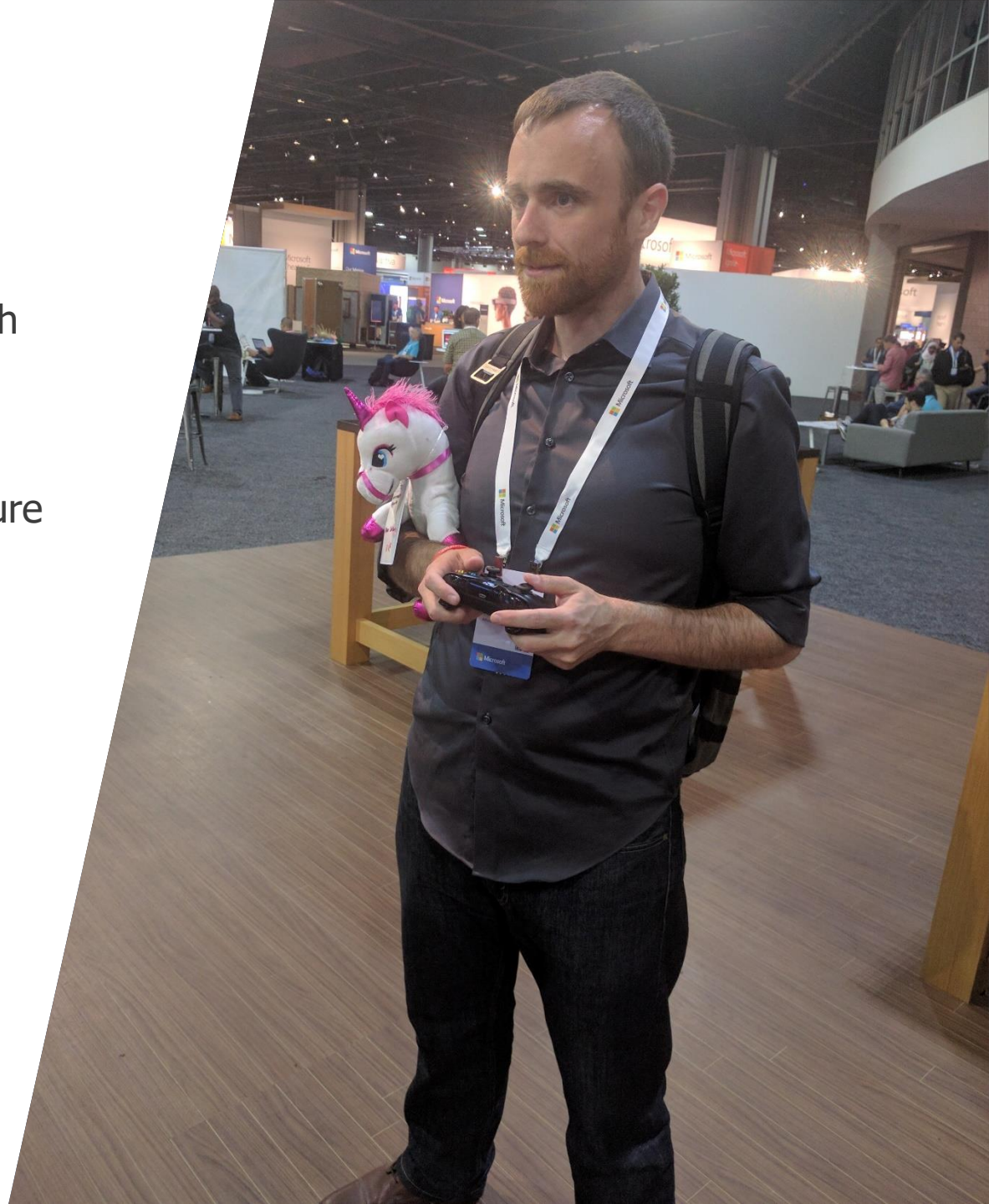
Deploy package with automated agent

Who am I?

Colin Vermander – Director of Software Engineering with **KPMG Canada Microsoft Practice**

- Responsible for technical oversight across all of KPMG Adoxio's product development and services solutions
- Over 15 years of technical experience, from infrastructure to development
- Microsoft Business Applications MVP
- Specialize in Dynamics 365 portals
- **Loves unicorns**

Twitter: @koolin_
LinkedIn: https://www.linkedin.com/in/colinvermander/
GitHub: https://github.com/koolin
Blog: https://colinvermander.com



Example DevOps Project

Source Code

https://github.com/koolin/cds-resourcemgmt-devops

Questions

Twitter: @koolin_

LinkedIn: https://www.linkedin.com/in/colinvermander/

GitHub: https://github.com/koolin

Blog: https://colinvermander.com

Microsoft Resources

- Dynamics 365 CE PowerShell
  https://docs.microsoft.com/en-us/powershell/dynamics365/customer-engagement/overview
- Online Management API for Dynamics 365 CE
  https://docs.microsoft.com/en-us/dynamics365/customer-engagement/developer/online-management-api
- PowerApps PowerShell cmdlets
  https://docs.microsoft.com/en-us/powerapps/administrator/powerapps-powershell
- Azure DevOps Pipelines
  https://docs.microsoft.com/en-us/azure/devops/pipelines

Open Source Resources

- Dynamics 365 Build Tools *by Wael Hamze*
  https://marketplace.visualstudio.com/items?itemName=WaelHamze.xrm-ci-framework-build-tasks
  https://github.com/WaelHamze/dyn365-ce-vsts-tasks
- Adoxio.Dynamics.DevOps *by Alan Mervitz*
  https://github.com/Adoxio/Adoxio.Dynamics.DevOps
- Microsoft.Xrm.Data.PowerShell *by Sean McNellis & Kenichiro Nakamura*
  https://github.com/seanmcne/Microsoft.Xrm.Data.PowerShell
- Dynamics 365 CE DevOps Examples *by Marc Schweigert*
  https://github.com/devkeydet/dyn365-ce-devops
- SparkleXrm\spkl *by Scott Durow*
  https://github.com/scottdurow/SparkleXrm/wiki/spkl

Business Applications Communities

Learn • Connect • Share • Inspire

Join the Microsoft Business Applications Communities where you can connect with peers and experts. Get answers to complex questions, learn from engaging discussions, read informative blogs, view webinars, and find product use examples in galleries.

https://community.dynamics.com

https://community.powerbi.com

https://community.powerapps.com

https://community.flow.microsoft.com

Benefits

- Join for free
- Access tips, answers, and shared knowledge from experts
- Expand your network by engaging with peers

Engagement

- Need help? Ask questions and join in on business or technical discussions in the forums
- Share your expertise by hosting a blog or syndicating your existing blog

Recognition

- Earn badges for participation and engagement
- Become a Community Star and earn appreciation from peers