

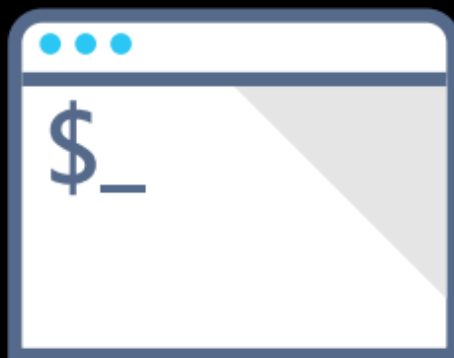
PREVIEW



Power Platform Center of Excellence Command Line Interface

<https://aka.ms/coe-cli>

July 2021 Edition



Grant Archibald

Microsoft Corporation

[1 Overview](#)

[1.1 Quick Start](#)

[1.2 Explain the Concepts](#)

[1.2.1 What Next](#)

[1.2.2 Getting Started](#)

[1.2.3 Getting Help](#)

[1.2.4 Read More](#)

[1.3 How does it work](#)

[2 Installation](#)

[2.1 Local Install](#)

[2.1.1 Prerequisites](#)

[2.1.2 Checking Prerequisites](#)

[2.1.3 Install](#)

[2.2 Docker Install](#)

[2.2.1 Prerequisites](#)

[2.2.2 Install](#)

[2.3 Sample Local Install](#)

[2.4 Sample Docker Install](#)

[3 Upgrade](#)

[3.1 Download](#)

[3.1.1 Local Upgrade](#)

[3.2 Docker Image](#)

[4 ALM Accelerator for Advanced Makers](#)

[4.1 Quick Start](#)

[4.2 Understand the Concepts](#)

[4.2.1 Getting Started](#)

[4.2.2 Overview](#)

[4.2.3 Sample Install](#)

[4.2.4 Install Overview](#)

[4.2.5 Read Next](#)

[4.3 Maturity Model](#)

[4.3.1 Quick Start](#)

[4.3.2 Understand the Concepts](#)

[4.4 Scenarios](#)

[4.5 Tenant Deployments](#)

[4.5.1 Factors to Consider](#)

[4.5.2 Demonstration Deployment](#)

[4.5.3 Enterprise Deployment](#)

[4.5.4 Multi Tenant Deployment](#)

[4.6 DevOps Deployment Model](#)

[4.6.1 Factors to Consider](#)

[4.6.2 Power Platform 1 to 1 DevOps](#)

[4.6.3 Power Platform 1 to Many DevOps](#)

[4.6.4 Multiple Power Platform Environments and DevOps](#)

[4.7 Personas](#)

[4.7.1 Persona Command Mapping](#)

[4.7.2 Persona Description](#)

[4.8 Key Concepts](#)

[4.8.1 Azure Active Directory](#)

[4.8.2 Azure DevOps](#)

[4.8.3 Power Platform](#)

[4.9 Branching and Merging](#)

[4.9.1 Example](#)

[4.9.2 PowerApps Component Framework \(PCF\)](#)

[4.10 Before You Start](#)

[4.10.1 CoE Command Line](#)

[4.10.2 Power Platform](#)

[4.10.3 Maker Environment Dataverse](#)

[4.10.4 Azure](#)

[4.10.5 Azure DevOps](#)

[4.10.6 Read Next](#)

[4.11 Admin Install](#)

[4.11.1 Before You Start](#)

[4.11.2 Initial Install](#)

[4.11.3 Read Next](#)

[4.12 Development Environments](#)

[4.12.1 Admin Maker Setup](#)

[4.12.2 Read Next](#)

[4.13 Maker Setup](#)

[4.13.1 Setup Service Principal](#)

[4.13.2 Maker Create Solution](#)

[4.13.3 Post Setup Checks](#)

[4.13.4 Read Next](#)

[4.14 Center of Excellence - Command Line Interface Help](#)

[4.14.1 Examples](#)

[4.14.2 Contribution](#)

[5 ALM Accelerator for Advanced Makers](#)

[5.1 Examples](#)

[5.1.1 Generate](#)

[5.1.2 Install](#)

[5.1.3 Connection](#)

[5.1.4 Maker](#)

[5.1.5 Branch](#)

[5.2 Generate](#)

[5.2.1 Examples](#)

[5.3 Install Generate](#)

[5.3.1 Description](#)

[5.3.2 Examples](#)

[5.3.3 Parameters](#)

[5.3.4 -s, --includeSchema](#)

[5.4 AA4AM Install Help](#)

[5.4.1 Examples](#)

[5.4.2 Parameters](#)

[5.5 Maker Add Generate](#)

[5.5.1 Description](#)

[5.5.2 Examples](#)

[5.5.3 Parameters](#)

[5.6 ALM Accelerate for Advanced Makers Add](#)

[5.6.1 Description](#)

[5.6.2 Example](#)

[5.6.3 Parameters](#)

[5.7 AA4AM Advanced Maker Solution Branch](#)

[5.7.1 Examples](#)

[5.7.2 Parameters](#)

[5.8 ALM Accelerate for Advanced Connection Add](#)

[5.8.1 Description](#)

[5.8.2 Example](#)

[5.8.3 Parameters](#)

[5.9 ALM Accelerate for Advanced Makers User Add](#)

[5.9.1 Description](#)

[5.9.2 Example](#)

[5.9.3 Parameters](#)

[6 CLI Development](#)

[6.1 Quick Start](#)

[6.1.1 Initial Commands](#)

[6.2 Understand The Concepts](#)

[6.2.1 Documentation](#)

[6.2.2 Development Frameworks](#)

[6.2.3 Contributions](#)

[6.2.4 Development Environment Setup](#)

[6.2.5 Debugging Commands](#)

[6.3 Documentation](#)

[6.3.1 Add Markdown Pages](#)

[6.3.2 Diagrams](#)

[6.3.3 Add Help Pages](#)

[6.3.4 Recording Command Line](#)

[6.4 Adding a new Command](#)

[6.4.1 Connecting the command to the command line](#)

[6.5 E-Book](#)

[6.5.1 Quick Start](#)

[6.5.2 Understand the Concepts](#)

1. Overview

For each set of major commands, this guide is organized in the following tracks to explain the Center of Excellence Command Line Interface (CoE CLI):

1. [Quick start](#) - Follow this track if your goal is to quickly setup a demonstration environment. Using this approach you can rapidly provide a hands on environment that you can use to illustrate the system end to end
2. [Explain the concepts](#) - Aimed at Enterprise deployments where individual components of the solution may be installed by different roles. Alternatively you may need a deeper understanding of the components that the CoE CLI is automating.
3. [How does it work](#) - Dives deeper into how to build, extend and contribute to the CoE CLI.

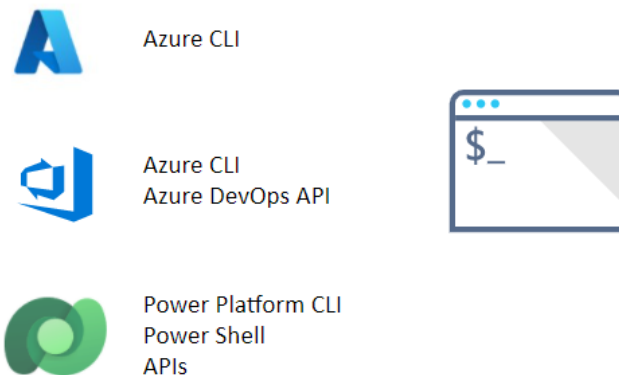
1.1 Quick Start

The following quick start guides exist

- [Install CoE CLI](#) - Install the CoE CLI to your local environment or as a docker container
- [ALM Accelerator for Advanced Makers Quick Start](#) - Install a demonstration environment or an enterprise deployment with Azure Active Directory, Azure DevOps and Power Platform Administration permissions

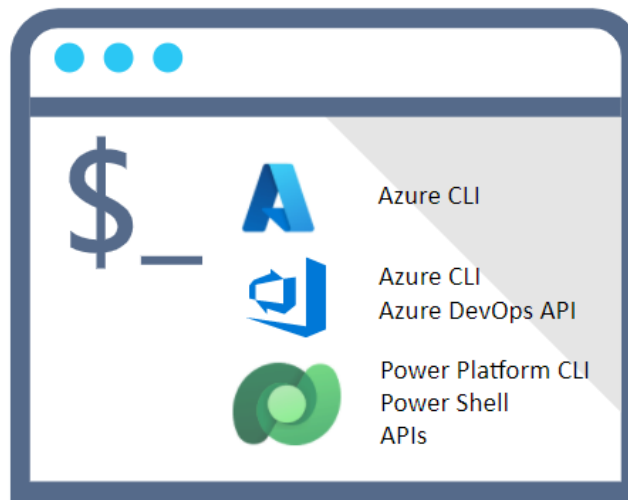
1.2 Explain the Concepts

Why?



The CoE CLI has been designed to provide a set of commands that meet the needs of different [personas](#) across the organization. It can be used to automate the process of installing CoE CLI components covering Azure, Azure DevOps and the Power Platform.

Unified Process



The CoE CLI wraps existing CLI and APIs to provide a set of commands that can be used to holistically automate the end to end Microsoft Solution deployment required by CoE CLI solutions.

Comparing and contrasting the CoE CLI to other CLI / APIs:

1. The CoE CLI aims to automate the end to end deployment of components across the Microsoft Cloud
2. The [Azure CLI](#) is aimed at Automating Azure Resources and via extensions Azure DevOps. The CoE CLI uses the Azure CLI for authentication and managing Azure related resources
3. The [Power Platform CLI](#) is a simple, one-stop developer CLI that empowers developers and ISVs to perform various operations in Microsoft Power Platform related to environment lifecycle features, and to authenticate and work with Microsoft Dataverse environments, solution packages, portals, code components. As new features are added to the cross platform Power Platform CLI the CoE CLI will leverage the Power Platform CLI features
4. The [Azure DevOps Services REST API](#) provides a REST based set of commands to interact with Azure DevOps. The CoE CLI makes use of these APIs to build aggregate commands.

What Next

As you consider an enterprise deployment the following sections outline the key concepts you will need to understand:

1. [Install CoE CLI](#) - How to install the CoE CLI using local host computer or via a docker container.
2. [ALM Accelerator for Advanced Makers](#) - Use CLI commands to setup and configure an environment for Advanced Makers to enable them to achieve more within your organization.

Getting Started

Once the CoE CLI has been [installed](#) you can use -h argument to see help options

```
coe -h
```

Authentication for tasks is managed using the Azure CLI. Using standard az cli commands you can login, logout and select accounts. For example

```
az login
coe aa4am install -c add
az logout
```

Getting Help

You can get short descriptions of any command by adding **--help** to the command line. To get more detailed help you can use the help command. For example to get help on the ALM Accelerator for Advanced Makers use the following command

```
coe help aa4am install
```

Read more in [help](#) pages for detailed description for each command.

Read More

Further reading

- [CoE CLI Upgrade](#) How to upgrade to a new version of the CoE CLI install.

1.3 How does it work

Interested in learning how the CoE CLI works or want to extend the functionality? The [CLI Development](#) provides the best place to start.

2. Installation

To install the CoE CLI

1. Download zip or clone repository
2. Change to unzipped or cloned repository

```
cd coe-starter-kit
```

3. Change to the coe-cli folder

```
cd coe-cli
```

Next select either [Local Install](#) or [Docker Install](#)

2.1 Local Install

Prerequisites

To run the CoE CLI application you will require the following

1. An installation of Node 11+
 - a. <https://nodejs.org/en/download/>
2. Azure CLI (version 2.24.0 or greater) is required for user authentication and Azure Active Directory Integration
 - a. <https://docs.microsoft.com/en-us/cli/azure/install-azure-cli>

Checking Prerequisites

To check prerequisites have been installed correctly and have the correct versions at the command prompt

1. Verify node version

```
node --version
```

2. Verify Azure CLI Version

```
az --version
```

Install

1. Install application dependencies

```
npm install
```

2. Build the application

```
npm run build
```

3. Link to the CoE CLI application

```
npm link
```

NOTE:

1. On Windows you may need to add %APPDATA%\npm to your PATH environment variable to access the coe command
2. Install Azure CLI. Follow install instructions for your operating system at <https://docs.microsoft.com/en-us/cli/azure/install-azure-cli>

2.2 Docker Install

One method of installation is via docker.

Prerequisites

To run the CoE CLI application you will require the following

1. A local install of [Docker](#) this can be either Community Edition or Enterprise Edition. If you are installing onto a desktop environment you will normally pick a Community Edition deployment.

Install

NOTE: On some operating systems you may need to use sudo before each of the docker commands.

1. Build docker image.

```
cd coe-cli  
docker build -t coe-cli .
```

2. Using the docker image

```
docker run -it --rm coe-cli
```

This will start a new interactive console (-it) and remove the docker container (--rm) when the console session exits. Using --rm ensures that any cached credentials are removed when you exit.

2.1 Sample Local Install

The following captures the results of a sample local install



2.1 Sample Docker Install

The following captures the results of a sample docker install on Linux



3. Upgrade

Upgrade will depend on how you installed the CoE CLI

3.1 Download

It you downloaded the CoE CLI as a zip file or a git clone

- Download the new zip file
- Unzip the zip file to a new folder

OR

- Pull changes from git

```
git pull
```

Once you have a local version of the coe-cli change to coe-cli folder

```
cd coe-cli
```

Local Upgrade

In the coe-cli folder run the following commands

1. Install the dependencies

```
npm install
```

5. Build the new version

```
npm run build
```

6. Update coe-cli to new version

```
npm link --force
```

3.2 Docker Image

In the coe-cli folder run the following commands

1. Build new docker image

```
docker build -t coe-cli .
```


4. ALM Accelerator for Advanced Makers

The [ALM Accelerator for Advanced Makers](#) (AA4AM) command allows you to manage common Application Lifecycle Management (ALM) tasks for Advanced Makers to install, setup and administration of Power Platform Solutions.

- [Quick Start](#) - Guides you through the process of a [demo tenant install](#)
- [Understand the Concepts](#) - Discusses Scenarios, Personas and Key Concepts behind AA4AM and the install process using the CoE CLI

4.1 Quick Start

This quick start should take around 30 minutes and by the end as an Administrator configure Azure Active Directory, Azure DevOps and the Power Platform environments.

1. Validate organization [maturity model](#) for AA4AM
2. Create your [Power Platform Environments](#) and [Azure DevOps](#) Organization and Azure DevOps project.
3. Create an install configuration. Review the [install help](#) for install parameters

```
coe aa4am generate install -o quickstart.json
```

More information on the [coe aa4am generate install](#) command

1. If you are creating a [demo tenant install](#) use the following command

```
coe aa4am install -f quickstart.json
```

More information on the [coe aa4am install](#) command

If you are deploying to your enterprise refer to [Enterprise Deployment](#) for further information.

1. [Update permissions for the project build service](#) to enable build pipelines to interact with Git Repositories
2. Have Advanced Makers create [development environments](#) then Add Advanced Makers to Azure DevOps and share the Canvas Application

```
coe aa4am maker add \  
-o https://dev.azure.com/contoso \  
-p alm-sandbox \  
-e https://contoso-userdev.crm.dynamics.com \  
-a aa4am-ado-service-principal \  
-g aa4am-makers -u user@contoso.com
```

More information on the [coe aa4am maker add](#) command

You can also generate a user configuration file. Using this approach will allow you to explore each parameter and review the settings before adding the maker.

```
coe aa4am generate maker add -o user.config  
coe aa4am maker add \  
-f user.config
```

More information on the `[coe aa4am generate maker add](../help/aa4am/generate/maker/add.md)` command

4.2 Understand the Concepts

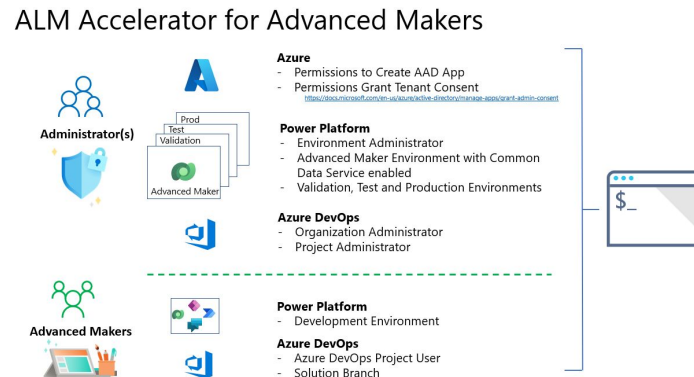
Not sure what AA4AM is and how it can help? The main GitHub project [README](#) provides further context and examples of usage.

Getting Started

- [Scenarios](#) - Discusses different install scenarios for AA4AM from Demo Installs to Enterprise Deployments
- [Personas](#) - Understand the key personas and how they map to AA4AM CLI commands and the wider AA4AM process
- [Key Concepts](#) - Understand the key concepts for the components that are being automated under the hood by the CLI commands
- [Branching and Merging](#) - Provides examples of branching and merging approach using AA4AM

Overview

The diagram below provides an overview of the key components required and permissions required.



Sample Install

The following recording shows a sample generating a install configuration file and installing AA4AM using the configuration file using a [Demo Deployment](#).

```
0: error
1: warn
2: info
3: verbose
4: debug
Default value(s) info
+ Your selection(s) separated by commas for log:
> Which choices for The component(s) to install
0: all
1: aad
2: devops
3: environment
Default value(s) all
+ Your selection(s) separated by commas for components:
> The azure active directory service principal application. Will be created if not exists (Default ALMAcceleratorServicePrincipal):
> The azure active directory servicemake group. Will be created if not exists (Default ALMAcceleratorForAdvancedMakers)
+
> The Azure DevOps organization to install into:https://dev.azure.com/CRM537626
> The Azure DevOps project name. Must already exist (Default alm-sandbox):
> The Azure DevOps pipeline repository. Will be created if not exists (Default pipelines):
> The Power Platform environment to install Managed solution to:https://cascade-maker.crm.dynamics.com/
> Which options for Optional settings
> Which choices for The environments to setup connections and applications user permissions
0: validation
1: test
2: prod
Default value(s) validation,test,prod
+ Your selection(s) separated by commas for installEnvironments:
```

Install Overview

1. Review the [Before You Start](#) to ensure you have the required Power Platform environments and Azure DevOps organizations created
2. As an administrator complete the [Admin Install](#)
3. Have Advanced Makers create [Development Environments](#)
4. Use [Maker Setup](#) to create and setup environment and solution branches in the Azure DevOps repository.

Notes:

1. If this is your first build pipeline you will need to set Pipeline Variables for your environment. At a minimum you will need to set **ServiceConnection** variable to your environment you have setup for validation, test and production.

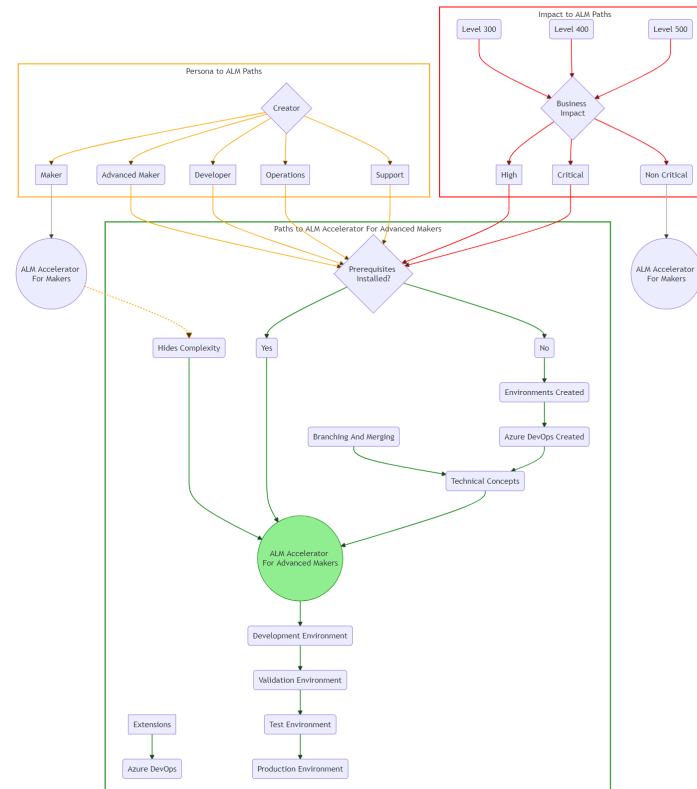
Read Next

- [ALM Accelerator for Advanced Makers](#) - Overview for AA4AM
- Manual Setup - Understand the key steps that the CLI is automating
 - [Foundational Setup](#) - Foundational Setup for Manual steps automated by the CLI install command
 - [Development Project Setup](#) - Documents the Manual steps that are automated by the CLI install command
 - [Solution Setup](#) - Documents the manual steps to setup Azure DevOps that are performed by the CLI install command
 - [Importing Solution](#) - Documents the manual steps to import the managed solution that are performed by the CLI install command

4.1 Maturity Model

The [Power CAT adoption maturity model](#) provides a set of levels and capabilities that can be used to evaluate usage of the Application Lifecycle Management (ALM) and how the ALM Accelerator for Advanced Makers (AA4AM) can assist.

Quick Start



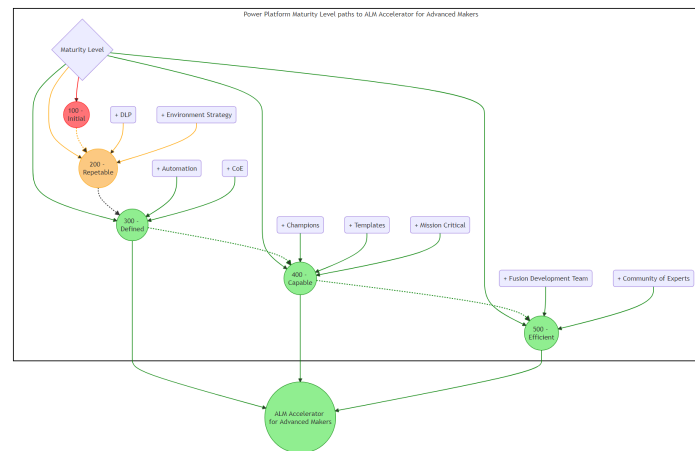
AA4AM is a good match if you can check to see if the following apply:

- What is the impact of the solution?
 - Are the applications classified as Critical or High [Business Impact](#)?
 - Who is using the application?
 - Is this a Productivity application used by everyone in your organization?
 - Is it used by Senior Leadership to make Business Impact Decisions?
 - Is it integrated with External users e.g. Partners / Customer who will rely on solution as part of external process?
 - Are there specific Compliance and Auditing needs?
 - How important is tracking and auditing who is using the application?
 - Is there compliance and auditing requirements of the solution?
- Do you have the prerequisites in place?
 - Do you have a defined [Environment strategy](#) for Development, Validation, Test and Production?
 - Do you use Azure DevOps or can you integrate with Azure DevOps for Source Control and Build Pipelines?
 - Do you have license prerequisites in place?
 - Do you have Basic or above Azure DevOps licenses for Makers?
 - Per App or Per User Power Apps licenses to access the AA4AM Maker Solution?
- Are you looking to move to [Level 300 - Defined](#) or beyond
- Administration
 - Do you have Environment Strategy in place?
 - What is the request strategy for environments?
 - Do you have a process to create Development / Test and Production Environments?
 - Is the process automated to request environments?
- Are Source control concepts understood?
 - Is [Branching and Merging](#) understood?
 - Are [Pull Request](#) used to review and merge changes?

- Are [Build Pipelines](#) used to integrate and deploy between Validation and Test, Production environments?
- Are [Fusion development](#) teams engaged to include Low code and Pro code teams?
 - Are your professional development teams familiar with [Branching and Merging](#) strategies and able to assist makers?
 - Do your Pro Code and Operations teams manage DevOps pipelines?
 - Are you Pro code team creating components in JavaScript?
 - Are you integrating with OpenAPI enabled Web APIs?
 - Are you using or planning to creating [plugins](#) to extend business process in Dataverse?
- Support
 - Who is supporting the application / solution?
 - Do you have formal support team to manage issues for the solution?

Where some gaps exist then a set of proactive training and or workshops can help grow the maturity of people, process and technology to assist them in moving toward Advanced Maker integration inside your organization.

Understand the Concepts



Levels

The maturity model has the following levels

Level 100 Initial

(*organic, chaotic, ad-hoc, individual heroics*): This phase describes the starting point for use of a new or undocumented process. In the Initial phase the organization has pockets of success or experimentation with Power Platform, but without any visibility into organization-wide adoption and use. There is no overall strategy or governance approach, apps are largely in the Default environment and no DLP policies have been put in place. Apps are mostly used by a single team and supported by the makers. Apps primarily use Excel and SharePoint as their data sources. The organization sees the potential of a strategic investment in the Power Platform, but there is no clear path forward for organization-wide execution.

Level 200 Repeatable

This phase describes a process that is at least documented sufficiently such that repeating the same steps may be attempted.

In the context of the Power Platform, organizations in the repeatable stage are taking what they've learned in the Initial stage to put structure around the deployment of Power Platform, often through controls implemented by a central IT team or other team focused on Power Platform.

The CoE Starter Kit is deployed to provide tenant-wide visibility into the use of Power Platform and begin to identify, if not control, applications that are beginning to become broadly used in the organization.

Environments are used as needed, but in an ad-hoc manner – for example, various Production environments and different DLP policies might be created without a consistent strategy. These organizations sometimes believe that the use of the

Power Platform is running “out of control” until they shape their use of the administrative and governance controls available for Power Platform, transitioning to the Defined Stage.

Level 300 Defined

This phase describes a process that is defined/confirmed as a standard business process.

The Defined organization is standardizing the repeatable practices that evolved in the Repeatable phase – for example, Environment and DLP requests are automated, solutions are used to move apps and flows between environments, and makers are starting to share common components. The organization is achieving measurable success with Power Platform to digitally transform and has a defined Power Platform Center of Excellence team.

Much of this transformation may still reflect the organic growth that got the organization to this point, but the Center of Excellence team is working to automate those processes and define standard approaches that will move the organization to the Capable stage.

Level 400 Capable

This phrase describes a process that is quantitatively managed in accordance with agreed-upon metrics.

The Capable organization has standard processes for managing and monitoring Power Platform. These processes, described during the **Defined stage**, are now largely automated and are well understood by makers.

Power Platform capabilities are being used to transform the business broadly and used for enterprise-critical apps and integrations. Platform Champions have established channels for sharing best practices, training new makers and conducting regular hackathons. Standard, branded app templates and components are available to all makers. Business Value assessments are carried out to measure and understand the impact of the Power Platform.

Level 500 – Efficient

This phrase describes a process that is quantitatively managed in accordance with agreed-upon metrics.

In the context of the Power Platform, in the Efficient stage the organization has proven the capabilities of Power Platform to rapidly transform mission critical capabilities. Standardized, automated processes and an established community of experts allow new digitization opportunities to be implemented rapidly, allowing the organization to recognize value quickly and begin to integrate more advanced capabilities, such as artificial intelligence (AI).

Fusion Teams enable legacy capabilities and modern cloud architecture to be used easily within Power Platform unlocking broad use of existing data and automation. In Organizations at the Leading Stage, the Power Platform is key part of the digital transformation strategy and Enterprise Architecture in the organization. Organizations at this stage have exec sponsorship for the Power Platform. Organizations at the Leading stage influence best practices in the community and drive new uses of Power Platform.

Capabilities

Strategy and Vision

Evaluates the ability to reach the organizational goals taking into account the following factors

- Is it aimed at the existing Business Model or targeting new Digital Business Models?
- What is the mix of different approaches from Incremental, Experimental, Evolutionary or Aspirational change?
- What is the effect on current people, process technology?
- What is the alignment with Tactical vs Strategic goals?

What are the challenges that are being considered:

- Investments - Are the investments accelerating business strategy?
- What are the People, Process & Technology Investments required to cover change?
- What is the impact of Disruption? – Introduction of new tech, new generation of economic desires & behaviors, increased pace of business innovation, competitive adaptation

- What is the pace of change?
- Is the pace or change stretching communication and adaptation capabilities?

What are the expected benefits of the strategy and vision. For example:

- Reduction of cost
- Optimize Operations
- Maximize Revenue streams
- Defend market share
- Improve customer satisfaction
- Automate processes
- Agility to react quickly to change

Leading questions

Alignment

- Is innovation driven by Business Areas (bottom up)?
- Is there a Common vision between IT and Business?
- Is there a Dedicated Power Platform product owner?
- Is there an established [Center of Excellence team](#)?
- Is Power Platform a key part of the digital transformation strategy?

Impact

- Is the Power Platform targeting low complexity scenarios?
- Is there limited reuse of common components and services?
- Do applications allow Bottom up and top down innovation?
- Do applications focus on increased delivery efficiency, supporting rapidly changing business needs?
- Are there Organization wide initiatives to deliver larger scale integrated apps?

Strategy

- Is the Power Platform strategy defined?
- Is there a demand management process in place?
- Is there a defined understanding of Power Platform's role in your organization's IT portfolio?
- Are business plans shared across departments?
- Is Vision and strategy understood by all?
- Do Enterprise Architecture decisions include Power Platform capabilities?

Business Value

Evaluate the business value looking at:

Business Strategy Viability

Leading questions to consider

- What Business outcomes will this realize?
- What is the expected time frame?
- What do you do well today?
- What do you want to do today better?
- What thing do you want to do differently?

Technological Viability

Leading questions to consider

- What are manual steps vs automated steps?
- How measurable are the qualitative and quantitative outcomes?
- What is the dashboard/report capability to allow stakeholders to visualize and drill into and tack action on data?
- How available are analytics?

- On what frequency are analytics updated?
- How frequently are changes required
- What is the technical debt that needs to be accounted for?
- What are the security implications?

Financial Viability

Leading questions to consider

- What is the economic value added?
- Does this address current market model or is a new model being developed?
- What time horizon for implementation?
- What investment model is required?

Business Impact

Critical

Production, operations, or deployment deadlines will be severely affected, or there will be a severe impact on production or profitability. Multiple users or services are affected.

Initial Response time < 60 minutes with 24x7 access

Issues demand an immediate response and require 24x7 operation, every day.

High

The system has moderate business impact and can be dealt with during business hours. The expected usage could be by multiple users or single users, customers.

Initial Response time 2, 4 Hours or Next Day, Business hours access with 24x7 available support

Non Critical

The system has minimal business impact. The issue is important but does not have a significant current service or productivity impact. Acceptable workarounds will be considered.

Initial Response time 4-8 hours or greater with Business hours access and support (For example 9:00AM - 5:00PM)

Admin and Governance

Leading questions to consider

- Who can create environments?
- What [Data Loss Prevention \(DLP\)](#) policies are in place?
- [Power Platform Service Admin](#) roles exists to Administer Power Platform tenants and Environments?
- Are tenants / environments isolated from each other?
- Is there monitoring in place?
 - Has the [CoE Starter Kit – Core](#) been installed?
 - Does [License, capacity and consumption monitoring](#) inform decision making?
 - Has the [CoE Starter Kit – Governance Module](#) been adopted to gain compliance insights and archive resources?
 - Does telemetry help identify business-critical apps and makers?
- Are custom environments used for specific use cases and ALM scenarios?

Support

- Are apps created by makers supported by a help desk or dedicated team?
- Has application / solution risk profile been defined detailing what level of support will be received?
- Is there an ongoing continuous improvement plan for the application?
- Are there clearly defined roles and responsibilities for the solution?

- Do the roles and responsibilities include ownership to build and operate the solution?

Nurture and Citizen Makers

- Do you have a [Training and Upskilling](#) program for your makers to help them learn key concepts to grow your pool of Advanced Makers?
- Do you have an Internal [Champions](#) Community established?
- Do you have the [CoE Starter Kit – Nurture](#) module adopted?
- Do you have show and Tell sessions to demonstrate Advanced Maker Concepts?
- Do you have an adoption campaign to demonstrate how fusion development processes work?
- Do you have a career path options for makers?
- Have you built a community of mentors to share Advanced Maker concepts and best practices?
- Do you have a Common development strategy and goals for Citizen and Professional developers?

Automation

- Do you have Environment and DLP connector policy requests that are automated?
- Do you have Communication about processes and compliance between Admin and Makers? Is this process automated?

Fusion Teams

- Do you have [Standard libraries, custom connectors and components](#) to be consumed by makers?
- Do you have the need for fusion teams manage source control and app lifecycle (e.g. Build, Verification, Test, Production)
- Do you have Cross-functional teams that plan and execute work jointly, including makers, testers and operational teams?
- Do you have Common development strategy and goals for Citizen and Pro developers needed for new projects?

4.1 Scenarios

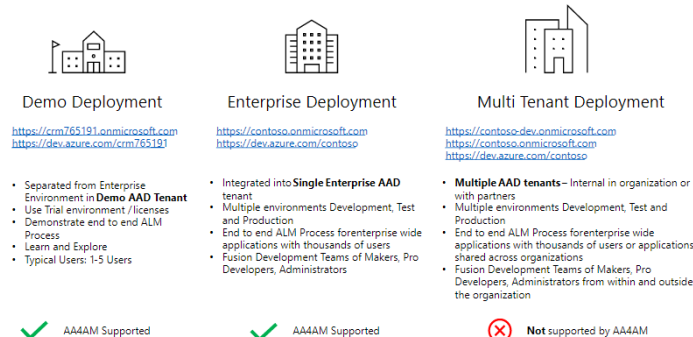
As you plan your ALM Accelerator for Advanced Makers (AA4AM) deployment one or more of the following scenarios may apply to you.

- [Tenant Deployments](#) - Discusses different Azure Active Directory configurations across Demo, Single AAD and Multiple AAD deployment models.
- [DevOps Deployment Model](#) - Discusses the relationship between AA4AM deployments and one or more different Azure DevOps organizations.

4.1 Tenant Deployments

AA4AM can be deployed in the following scenarios single demo tenant, single enterprise tenant.

Currently AA4AM **does not** automatically support a multi tenant enterprise deployment model and additional manual configuration will be required to configure this scenario.



Factors to Consider

- Does ALM for Low Code solutions introduce new concepts to parts of the business that has not been exposed to them before?
- Would a demo deployment provide an environment to allow the different [personas](#) to experiment and accelerate adoption of ALM processes?
- Who will manage and operate the ALM process?
- How will the maker community be expanded to adopt new ALM concepts?
- What steps can be put in place to adopt a self service model to provision environments and move between validation, test and production environments.

Demonstration Deployment

In this scenario you are looking to quickly install AA4AM to demonstrate how it works and showcase the end to end process. For this scenario the following is expected.

1. Using Trial tenant and environments to demonstrate the solution
2. Single Administrator that has rights to the following:
 - Azure Active Directory tenant administrator
 - Power Platform Global Administrator
 - Power Platform Organization Administrator
3. Demo non administration maker users that will be used to show process of creating ALM process for Power Platform solutions
4. Non production applications

Once you have the [Admin Install](#) completed, Advanced makers can create [Development environments](#) and have Administrators add them to Azure DevOps and the required Azure Active Directory Security Group.

This will typically use the following commands as the **single administrator**

```
coe aa4am generate install -o quickstart.json
coe aa4am install -f quickstart.json
```

More information on the [coe aa4am generate install](#) command More information on the [coe aa4am install](#) command

Then add a demo user as a maker

```
coe aa4am maker add \  
-e https://alans-dev.crm.dynamics.com \  
-o https://dev.azure.com/contoso-dev \  
-p alm-sandbox \  
-u alan-s@crm716415.onmicrosoft.com
```

More information on the [coe aa4am maker add](#) command

Once these steps are completed makers can then [Setup Managed Solutions](#)

Enterprise Deployment

In this scenario the aim is to install AA4AM inside an enterprise tenant and the following is expected.

1. Likely to have different administration teams. For example
 - Azure Active Directory Administrators
 - Power Platform Administrators. May be Global Administrator or Environment Administrators
 - Azure DevOps Administrators
2. Configuration files for AA4AM install can be shared among different Administration teams
3. Advanced Makers have separate development environments to work on changes
4. AA4AM Azure DevOps pipeline used to validate and promote to Test and Production environments

Azure Active Directory Administrators

The tenant administration team will need to create the following

1. Azure Active Directory Application that will be used as Service Principal in Azure DevOps and Power Platform Environments
2. Azure Active Directory Group that will be used to grant access to Advanced Makers to Azure DevOps resources, Maker Canvas Application and Dataverse Tables.
3. Grant Tenant Consent for Azure Active Directory Application. This required as the Azure DevOps pipeline uses APIs where an interactive user is not involved. As a result the tenant administrator consent is required.

To install the solution resources the following options can be used

Azure Active Directory

1. Use the CLI to install the AAD components. For example using the default install parameters

```
coe aa4am install -c aad
```

2. Using a shared configuration file and setting components array value to be ["aad"]

```
{  
  "log": [  
    "info"  
  ],  
  "components": [  
    "aad"  
  ],  
  "aad": "ALMAcceleratorServicePrincipal",  
  "group": "ALMAcceleratorForAdvancedMakers",  
  "devOpsOrganization": "https://dev.azure.com/contoso-dev",  
  "project": "alm-sandbox",  
  "repository": "pipelines",  
}
```

```

"settings": {
  "installEnvironments": [
    "validation",
    "test",
    "prod"
  ],
  "validation": "https://sample-validation.crm.dynamics.com",
  "test": "https://sample-test.crm.dynamics.com",
  "prod": "https://sample-prod.crm.dynamics.com",
  "createSecret": "true",
  "region": [
    "NAM"
  ]
},
"importMethod": "api",
"endpoint": "prod"
}

```

Azure DevOps

```

coe aa4am install -c devops \
  -o https://dev.azure.com/contoso-dev \
  -p alm-sandbox

```

Power Platform Environment

```

coe aa4am install -f install.json

```

```

{
  "log": [
    "info"
  ],
  "components": [
    "environment"
  ],
  "aad": "ALMAcceleratorServicePrincipal",
  "group": "ALMAcceleratorForAdvancedMakers",
  "devOpsOrganization": "https://dev.azure.com/contoso-dev",
  "project": "alm-sandbox",
  "repository": "pipelines",
  "settings": {
    "installEnvironments": [
      "validation",
      "test",
      "prod"
    ],
    "validation": "https://sample-validation.crm.dynamics.com",

```

```

    "test": "https://sample-test.crm.dyamics.com",
    "prod": "https://sample-prod.crm.dyamics.com",
    "createSecret": "true",
    "region": [
        "NAM"
    ]
  },
  "importMethod": "api",
  "endpoint": "prod"
}

```

3. Manual install using the [Create An App Registration in your AAD environment](#)

Multi Tenant Deployment

This deployment type involves different Azure Active Directory deployments that separate development, test and production systems. For example the following Azure Active Directory tenants

- contoso.onmicrosoft.com
- contoso-dev.onmicrosoft.com

Currently the AA4AM installation does not automatically support a multi-tenant deployment without further manual updated.

Multi Tenant Deployment Assumptions

The multi tenant deployment is assumed to have one or more of the following

1. Multiple Azure Active Directory tenants
2. Power Platform Environments for Development, Validation, Test and Production may be in different tenants.
3. The Azure DevOps environment may be in the Development tenant
4. Users of the main tenant may use Azure Business to Business authentication to access the development tenant
5. External users from outside the organization maybe invited to the development tenant and not have access to the main tenant

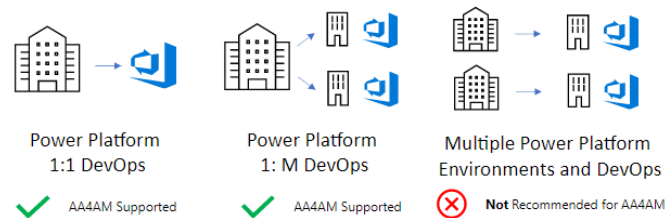
Azure Active Directory Implications

To support multi tenant deployments the Azure Active Directory application will need to be configured to support multi tenant authentication.

Further reading

1. [Tenancy in Azure Active Directory](#)

4.1 DevOps Deployment Model



Factors to Consider

As you plan your DevOps deployment model the following factors may help you decide the best approach:

- Do different teams and / or business units require separate locations to store and manage solutions?
- Cost / benefit of the overhead of managing and maintaining managing multiple DevOps process for different Power Platform Environments?
- Do different teams or business units have differing levels of business sensitivity or data loss prevention that will require separate handling?
- How will common shared assets be shared and managed between multiple teams? For example shared DevOps templates or Shared components.
- Azure Active Directory security model. Will each team manage and maintain Active Directory Applications and Service Principals or will they be shared across different Power Platform environments and Azure DevOps organizations?

Power Platform 1 to 1 DevOps

In this scenario the following is assumed:

- Single Power Platform Advanced Maker Deployment with Shared Power Platform Environments for the Advanced Maker Canvas Application, Validation, Test and Production Environments
- Single Shared Azure DevOps where all Advanced Makers collaborate on Solutions.

Power Platform 1 to Many DevOps

In this scenario the following is assumed:

- Single Power Platform Advanced Maker Deployment with Shared Power Platform Environments for the Advanced Maker Canvas Application, Validation, Test and Production Environments
- Multiple Azure DevOps for different teams or Business Units.
- Each Azure DevOps organization can have a different set of users protected buy different Role Based Access Security Rules
- Multiple pipelines can be defined and used at an Azure DevOps level that allow solutions to be deployed to different validation, test and production environments.

Multiple Power Platform Environments and DevOps

This scenario is not a recommended deployment model as is requires the deployment and management of multiple AA4AM deployments withing the organization.









In this scenario the following is assumed:

- Multiple Power Platform Advanced Maker Deployment with Shared Power Platform Environments for the Advanced Maker Canvas Application, Validation, Test and Production Environments
- Multiple Azure DevOps for different teams or Business Units
- Each Azure DevOps organization can have a different set of users protected by different Role Based Access Security Rules

4.1 Personas

Understanding the roles that different personas play in ALM Accelerator for Advanced Makers is important in helping you plan for and implement a AA4AM deployment in your organization.

Persona Command Mapping

AA4AM CLI Commands				
	coe aa4am branch coe aa4am user add	coe aa4am maker add	coe aa4am connection add	coe aa4am install
Outcome	<ul style="list-style-type: none">Create a Solution branch in Azure DevOpsAdd Application User to Development Environment	<ul style="list-style-type: none">Create Service ConnectionAdd to Security Group	<ul style="list-style-type: none">Create Service Connection<ul style="list-style-type: none">✓ Validation✓ Test✓ Production	<ul style="list-style-type: none">Create AAD User, GroupSetup Azure DevOpsImport Managed AA4AM SolutionSetup Security
Frequency	Run Per New Solution Run Once Per Dev Environment	Run Once Per Maker	Run Once Per Azure DevOps Organization	Run Once Per Organization
Persona	Advanced Makers Professional Developers	Owner of AAD App / Group created in install Azure DevOps Administrator	Owner of AAD App / Group created in install Azure DevOps Administrator	AAD Administrator Power Platform Administrator Azure DevOps Administrator
Components		 	 	  

Solution Setup

Assuming that AA4AM has been setup and installed, the first command that the Advanced Maker will run is the **coe aa4am user add** command. This command will register an Application User created during install as a System administrator in their development environment to integrate with the solution. For example using the default parameters

```
coe aa4am user add \  
-e https://contoso-alans-dev.crm.dynamics.com
```

Read more on the [coe aa4am user add](#)

The **coe aa4am branch** command will be run each time a new solution is created. This command allows a new solution branch to be created in Azure DevOps with the associated ALM DevOps pipeline to validate pull requests and push changes to test and production environments.

```
coe aa4am branch \  
-o https://dev.azure.com/contoso \  
-p alm-sandbox \  
-d MySolution
```

More information on the [coe aa4am branch](#) command

Administrator Setup

As each Advanced Maker or Professional Developer creates a development environment it will need to be registered with Azure DevOps and the Azure Active Directory Application. The user running this command requires Project Administrator rights in Azure DevOps and Owner rights of the Azure Active Directory Application.

```
coe aa4am maker add \  
-o https://dev.azure.com/contoso \  
-p alm-sandbox \  
-e https://contoso-userdev.crm.dynamics.com \  
-a aa4am-ado-service-principal \  
-g aa4am-makers \  
-u alan-s@contoso.com
```

More information on the [coe aa4am maker add](#) command

Each Azure DevOps project will also require connections to deployment environments used by Azure DevOps pipelines

```
coe aa4am connection add \  
-o https://dev.azure.com/contoso \  
-p alm-sandbox \  
-e https://contoso-build.crm.dynamics.com \  
-a aa4am-ado-service-principal  
  
coe aa4am connection add \  
-o https://dev.azure.com/contoso \  
-p alm-sandbox \  
-e https://contoso-test.crm.dynamics.com \  
-a aa4am-ado-service-principal  
  
coe aa4am connection add \  
-o https://dev.azure.com/contoso \  
-p alm-sandbox \  
-e https://contoso-prod.crm.dynamics.com \  
-a aa4am-ado-service-principal
```

More information on the [coe aa4am connection add](#) command

Install

To deploy an instance of AA4AM in your organization the **coe aa4am generate install** and **coe aa4am install** commands are used once to deploy the Managed solution. The install will automate key elements

- Azure Active Directory
 - New Azure Active directory Application that will be used by Azure DevOps for Service Connections and Access to Power Platform Environment
 - New Azure Active Directory Group to provide access to Azure DevOps resources and Share the Maker Canvas application
- Azure DevOps resources
 - Import Azure DevOps Pipelines
 - Variable Groups
 - Create Service Connections to Power Platform Environments
- Power Platform Environments
 - Setup Application Users
 - Import Managed Solution and Setup Security

```
coe aa4am generate install -o install.json  
coe aa4am install -f install.json
```

More information on the [coe aa4am generate install](#) command

More information on the [coe aa4am install](#) command

Persona Description

- **Business Users** - Licensed internal users of the created solutions. Will not directly use the AA4AM tools they will be able to see the shared applications. May report version number of application to the support team.
- **Maker** - Wants to use components or services produced by an advanced maker or professional developer. Uses off the shelf components and documentation. Not directly exposed to the Application Lifecycle as this process is abstracted away. Create and Share the Application with Business Users.

- **Advanced Maker** - Collaborates with Professional Development and IT teams to integrate and build applications. Assumed to be familiar with concepts like ALM, DevOps, Branching and Merging. Works in Development environment and push changes into validation and testing and production environments. Uses managed AA4AM Canvas management application and Azure DevOps website.
- **Professional Developer** - Advanced maker knowledge plus the ability to use lower level development programming languages and SDKs to create components and services. For example JavaScript and PCF controls, Dataverse Plugins in C#, Azure Services and APIs e.g. Azure Functions, API Management. Likely to work in tools like Visual Studio Code.
- **Data Analyst** - Develop data model, Create and manage data services and post data collection analysis / reporting. For example Power BI reporting, Datalake. For data elements that are covered in the Solution system e.g. Dataverse Modeling, AI Models. Items not covered today in solution system like Power BI will have separate ALM process.
- **Operations Teams** - Deploy solution to environments across Power Platform and Microsoft Cloud Services (e.g. Azure). Distribute solutions into Power Platform and run ARM templates in Azure. Will not use the CLI commands directly. May use managed canvas application to view and Azure DevOps pipelines to view the status or promote applications from test to production.
- **Support Teams** - Post application deployment look at version of applications deployed, triage issues. May use managed canvas application to view deployed solution versions.
- **Information Security Team** - Will compare against organization standards for Data Loss Prevention (DLP), Authentication and Authorization, Service Principals, Teams and Security. Review the ALM process against Threat models, risks and mitigations.
- **Architecture Team** - Review the entire ALM process and components and verify matches solution methodology and architecture
- **Administrators**

- **Power Platform Tenant Administrator** - Global right to Power Platform Administration - Manage Environments (Create, Update, Delete). Common commands

```
coe aa4am generate install -o data.json
coe aa4am install -f data.json
```

More information on the [coe aa4am generate install](#) command

More information on the [coe aa4am install](#) command

- **Power Platform Environment Administrator** - Manage One or more Power Platform Environments - Import solution, add users assign roles

```
coe aa4am generate install -o data.json
coe aa4am install \
  -c environment \
  -e https://contoso-maker.crm.dynamics.com
```

More information on the [coe aa4am generate install](#) command

More information on the [coe aa4am install](#) command

Add makers to an environment (Assuming they also have Azure DevOps Administrator rights)

```
coe aa4am maker add \
  -e https://user-Dev.crm.dynamics.com \
  -o https://dev.azure.com/dev12345 \
```

```
-p alm-sandbox \  
-u user@contoso.com
```

More information on the [coe aa4am maker add](#) command

Azure Tenant Administrator Manage the AAD Tenant - Create User, Groups, Applications and Service Principals (O365 or Azure Administrators). Common commands

```
coe aa4am install -c aad
```

More information on the [coe aa4am install](#) command

Azure DevOps Organization Administrators

Azure DevOps Project Administrators

```
coe aa4am install -c devops \  
-o https://dev.azure.com/dev12345 \  
-p alm-sandbox
```

More information on the [coe aa4am install](#) command

4.1 Key Concepts

As you deploy and use the AA4AM CLI it is important to understand the following key concepts that the CLI is automating.

Azure Active Directory

Azure Active Directory Application

The CoE CLI application can create a Azure Active Directory application that automates the following key steps.

1. User authenticated via Azure CLI
 - [Azure Login](#)
2. Create Azure Active Directory Application using Azure CLI
 - [Create AD Application](#)
 - [Create Service Principal](#)
3. Grant Tenant Consent for Applications using Azure CLI
 - [Permissions Admin Consent](#)
4. Azure Application Granted rights via [manifest configuration file](#) to call
 - Azure DevOps
 - Dataverse
 - PowerApps Checker Module - [Read More](#)
5. Client secrets will be created for Azure DevOps Service Connections
 - Client secrets should have an established key rotation process to generate new keys for connections
 - After new keys are generated old keys should be removed

Azure Active Directory Group

The CoE CLI application can create a Azure Active Directory group that is used for Azure DevOps and Power Platform authentication and role based access security.

1. Group Created via Azure CLI
 - [Create Group](#)

Azure DevOps

Install Automation

The CoE CLI application assumes that an Azure DevOps organization and project have already been created.

The install performs the following key steps:

1. Install Azure DevOps Extensions defined in [AzureDevOpsExtensionsDetails.json](#)
 - a. New extensions can be added from <https://marketplace.visualstudio.com/item>. Visit each extension and add the publisher and the **itemName** from the query string in the browser
2. Clone Azure Templates <https://github.com/microsoft/coe-alm-accelerator-templates.git> into a Azure DevOps git repository named **pipelines** by default

3. Create Azure DevOps build pipelines
 - [export-solution-to-git.yml](#) - Export a solution from a Dataverse environment and commit it to a git branch.
 - [import-unmanaged-to-dev-environment.yml](#) - Import solution into Dataverse environment
 - [delete-unmanaged-solution-and-components.yml](#) - Delete or "clean up" an unmanaged solution from a Dataverse environment
4. Setup Azure Active Group access the Azure DevOps project
5. Create Variable Groups for shared variables used by build pipelines
 - [Read More](#)
6. Create Service connections to Power Platform Environments using the Azure Active Directory Service Principal
 - NOTE: Each service connection will receive a separate Azure Active Directory secret.
 - [Read More - Service Connections](#)

Branch Automation

The [coe aa4am branch](#) command performs the following steps

1. Create a new branch to store the Solution
2. Create build pipelines for the Solution Branch (Validation, Test, Production)
3. Create [Branch Policies](#) to ensure validation build completes successfully for pull requests

Other Concepts

In addition to install automation the following concepts are also assumed for Advanced Makers

1. A git [branching strategy](#)
 - The AA4AM assumes a branch per solution
 - Changes merged back into main branch can be promoted to production environment
1. Manage Pull Requests to merge changes into Solution Branches <https://docs.microsoft.com/en-us/azure/devops/repos/git/pull-requests?view=azure-devops>

Power Platform

Automation

The CoE CLI provides the following key steps.

1. Import Managed solution into environment to allow Advanced Makers to Manage git import, create branches, pull requests and updates to test and production.
2. Fix Custom Connectors used to connect to Azure DevOps
3. Connect Flow to Dataverse
4. Add the user to the Azure Active Directory Service Principal to the Power Platform environments
5. Share the Canvas application with the Maker Azure Active Directory Group

Environments

The CoE CLI commands currently assume the following environments have been created <https://aka.ms/ppac> as a Global Administrator or Environment Administrator

1. Maker Environment

- Require Dataverse to be enabled
- Created "Microsoft Dataverse (legacy)" connection - [Read More](#)

2. Validation, Test, Production Environments

- CLI can setup Azure Active Directory application as Application User with System Administrator rights

3. Developer Environments

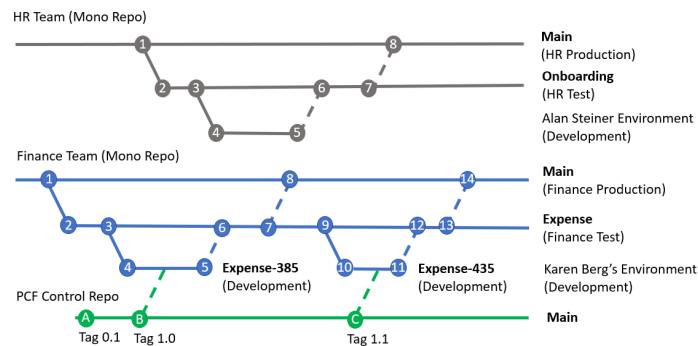
- CLI can setup Azure Active Directory application as Application User with System Administrator rights

4.1 Branching and Merging

Understanding of [Branching and Merging](#), [Pull Request](#) and [Build Pipelines](#) are important concepts that are combined together with fusion development teams of Advanced Makers, Professional Developers and Operations teams will need to understand. The ALM Accelerator for Advanced Makers (AA4AM) builds on these concepts to allow Power Platform solutions to be managed.

Example

The example below illustrates two parts of the organization the HR and Finance teams using different DevOps projects to manage related solutions for each department.



HR Team

The HR team manages one Azure DevOps git repository that stores each solution as a folder within the repository. This approach allows Advanced Makers in the HR team to see and collaborate on HR related solutions.

Steps

1. Create a new Azure DevOps Project and from the main branch for a new solution branch named Onboarding.

```
coe aa4am branch \  
  -o https://dev.azure.com/contoso \  
  -p HR \  
  -r HR-Solutions \  
  -d Onboarding \  
  -s validation=https://contoso-hr-validation.crm.dynamics.com,test=r
```

More information on the [coe aa4am branch](#) command

2. After this command is run a default branch with Azure DevOps pipelines has been created in the repository
3. The Advanced Maker **Alan Steiener** creates an unmanaged Solution named **Onboarding** in his development environment
4. Using the AA4AM Administration application Alan Created a new feature branch and **Push changes to Git**
5. Once the initial set of features is complete **Create Pull Request** using the AA4AM Administration application. The validation build pipeline is executed. The Pull Request is approved and then the feature branch is committed to the solution branch
6. The merged commit can trigger a Continuous Deployment to the test environment
7. When the set of features are ready for a production deployment a Pull Request can be made to merge changes into the main branch

8. The build and deployment pipelines can be configured to package the solution to the production environment

Finance

The Finance team maintains a separate Azure DevOps project for Finance related solutions.

This Azure DevOps project could be in:

- The same Azure DevOps project with a different repository from the HR team (e.g. Different HR-Solutions and Fin-Solutions repositories)
- A separate Azure DevOps project and repository. This would allow different Role based security rights for the project
- A separate Azure DevOps Organization, Project and repository. This would allow different Azure Active Directory tenants to be used.

Steps

Karen as the Advanced Maker in the finance team follows a similar process to what Alan did inside the HR team.

1. **Karen** creates a new Azure DevOps solution branch for the Expense application.

```
coe aa4am branch \  
  -o https://dev.azure.com/contoso \  
  -p Finance \  
  -r Finance-Solutions \  
  -d Expense \  
  -s validation=https://contoso-fin-validation.crm.dynamics.com,test=
```

More information on the [coe aa4am branch](#) command

2. After this command is run a default branch with Azure DevOps pipelines has been created in the repository
3. **Karen Berg** creates an unmanaged Solution named **Expense** in her development environment
4. Using the AA4AM Administration application Karen creates a new feature branch with the ID of the work item that has been assigned **385** and **Push changes to Git**
5. Once the initial set of features is complete **Create Pull Request** using the AA4AM Administration application. The validation build pipeline is executed. The Pull Request is approved and then the feature branch is committed to the solution branch
6. The merged commit can trigger a Continuous Deployment to the test environment
7. When the set of features are ready for a production deployment a Pull Request can be made to merge changes into the main branch
8. The build and deployment pipelines can be configured to package the solution to the production environment

Steps 9 through 14 repeat the process of 3-8 to contribute a new feature to the solution.

PowerApps Component Framework (PCF)

In this example the Finance application makes use of a common component to visually interact with their data.

This PCF component is managed in a separate code repository. As new releases are created they are tagged with release versions.

A release version is imported into a feature branch for a Power Platform solution. This approach allows different versions of the PCF control to be developed and integrated with different solutions over time. In the Finance example version 1.0 to 1.1, which is committed to the **Expense-435** branch to update the PCF control being used.

4.1 Before You Start

Before you start an install of AA4AM ensure that you have the following in place

CoE Command Line

Install the CoE CLI [locally](#) or via a [docker image](#)

Power Platform

Environment	Description
Developer	Development environments that each Advanced Maker will use to create and manage source controlled solutions
ALM Environment	Environment with Dataverse enabled. Will be used to deploy managed solution.
ALM Environment - Dataverse Connection	See Maker Environment Dataverse below to create the Dataverse Connection
Validation	Environment used to validate builds before merging into a solution branch
Test	Preproduction Environment used to test solutions before moving to production
Production	Production Environment for managed solutions

Notes:

1. As a Microsoft Partner you can request access to demo tenants to test ALM Accelerators for Advanced Makers
 1. Visit <https://docs.microsoft.com/en-us/partner-center/mpn-demos> for more information
 2. Go to <https://demos.microsoft.com> to request a new environment
 3. Select My Environments
 4. Select Create Tenant
 5. Select tenant location
 6. Select "Dynamics 365 Customer Engagement"
2. Sample environment from <https://admin.powerplatform.microsoft.com/environments> for a [Demo Deployment](#)

Environments

Environment	Type	State
Alan Steiner's Environment	Developer	Ready
crm781878-prod	Trial (subscription-based)	Ready
crm781878-test	Trial (subscription-based)	Ready
crm781878-validation	Trial (subscription-based)	Ready
Contoso (default)	Default	Ready

Maker Environment Dataverse

In the maker environment you will need a Dataverse connection to be created by the install user.

This can be done using the following steps

1. Go to <https://make.powerapps.com/>
2. Select your maker environment that you will deploy the ALM Accelerator for Advanced Makers into
3. Navigate to Data -> Connections
4. New Connection
5. Microsoft Dataverse (legacy)

Azure

Ensure the user you run with has the following permissions

Component	Description
Global Administrator or Privileged Role Administrator	Grant tenant-wide admin consent to an application Read More

Azure DevOps

The following must be installed before the CoE CLI AA4AM install can begin

Component	Description
Organization Review	Add Organization Users to create Azure DevOps organization and add users
Project	An Azure DevOps project to integrate with. This guide uses the name alm-sandbox as the project name

The following will be setup or used as part of the install and follow on [coe aa4am branch](#)

Component	Description
Extensions	Review the extensions configuration that will be installed
Repository	Ensure the git repository has been initialized with an initial commit

Notes:

1. If installing using demo tenant you can request a trial Azure DevOps environment
 - <https://azure.microsoft.com/en-us/services/devops/>
 - Create Organization
 - Create initial project e.g. alm-sandbox
2. Review the Azure [DevOps Extensions](#) that will be installed by the CLI application to ensure your organization gives consent for them to be installed.
 - [Power Platform Build Tools](#)
 - [Power DevOps Tools](#)
 - [RegexReplace Azure Pipelines Task](#)
 - [SARIF SAST Scans Tab](#)

Read Next

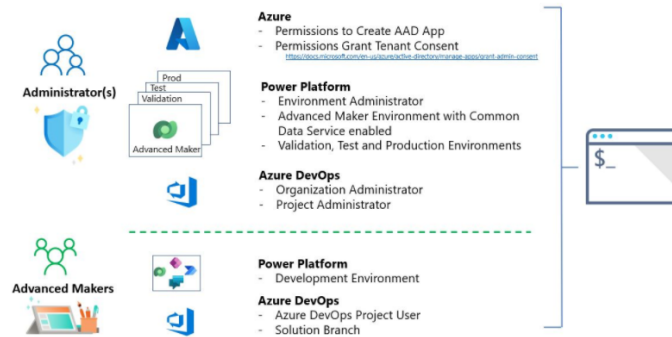
Once you have verified the above

1. Determine the install [deployment scenario](#) you are targeting
2. Complete the [Admin Install](#)

4.1 Admin Install

To complete the initial steps of an AA4AM deployment you will need to complete the administrative tasks. Once this is done Advanced Makers can create and register development environments.

ALM Accelerator for Advanced Makers



It is assumed that the Admin Install will be run by a single user that has Power Platform Global Administrator, DevOps Administrator rights and Azure Active directory Administrator rights.

Before You Start

Complete [Before You Start](#) to ensure that:

- Power Platform Environments have been created
- Azure DevOps Organization and Project has been created
- CoE CLI installed

Initial Install

- Create install configuration file and review the generated JSON file and confirm the settings before you start the install process

```
coe aa4am generate install -o test.json
```

More information on the [coe aa4am generate install](#) command

Which will generate a file similar to

```
{
  "log": [
    "info"
  ],
  "components": [
    "all"
  ],
  "aad": "ALMAcceleratorServicePrincipal",
  "group": "ALMAcceleratorForAdvancedMakers",
  "devOpsOrganization": "https://dev.azure.com/dev1234",
  "project": "alm-sandbox",
  "repository": "pipelines",
  "settings": {
    "installEnvironments": [
```

```
        "validation",
        "test",
        "prod"
    ],
    "validation": "https://sample-validation.crm.dynamics.com",
    "test": "https://sample-test.crm.dynamics.com",
    "prod": "https://sample-prod.crm.dynamics.com",
    "createSecret": "true",
    "region": [
        "NAM"
    ]
},
"importMethod": "api",
"endpoint": "prod"
}
```

2. Review the JSON and install using the following command

```
coe aa4am install -f test.json
```

More information on the [coe aa4am install](#) command

3. [Update permissions for the project build service](#) to enable build pipelines to interact with Git Repositories

Read Next

- Complete the [Install Overview](#)

4.1 Development Environments

Each advanced maker will need a development environment created. Advanced makers can use a community environment to work in. Community environments can be accessed from the sign-up page

<https://web.powerapps.com/community/signup>

Admin Maker Setup

As Azure DevOps Administrator you will need to register each advanced maker environment. The following command will add the required service connection to the development environment and setup security for the user

```
coe aa4am maker add \  
-o https://dev.azure.com/dev12345 \  
-p alm-sandbox \  
-e https://org12345-dev.crm.dynamics.com \  
-u username@contoso.com
```

More information on the [coe aa4am maker add](#) command

Read Next

- Complete the [Install Overview](#)

4.1 Maker Setup

Once a user has been setup with a development environment [Read More](#) they will need to use the following sections to provide access to the service principal and create a solution branch.

Setup Service Principal

The Azure DevOps Pipeline uses a Azure Active Directory service principal to connect to the development environment and import and export the solution. To enable access to the environment the following command will add the service principal as a System Administrator to the developer Power Platform environment.

1. Log out of any existing sessions if not the maker or a Power Platform tenant Administrator

```
az logout
```

2. To ensure the Application User has access to the development environment you need to run the following command

```
coe aa4am user add \  
-e https://org12345-dev.crm.dynamics.com
```

More information on the [coe aa4am user add](#) command

Maker Create Solution

Once the environment has been setup and your development environment created and registered as a service connection in Azure DevOps you can use the following steps to create a source control managed solution.

1. Switch to Developer Environment
2. Create new solution e.g. NewSolution1
3. Add items to the solution. For example
 - a. Select Solution
 - b. Add Canvas Application
 - c. Add Button
 - d. Save Application and Close
4. Create Solution branch using the following CLI command

```
coe aa4am branch \  
-o https://dev.azure.com/dev12345 \  
-p alm-sandbox \  
-d MySolution1
```

More information on the [coe aa4am branch](#) command

NOTES:

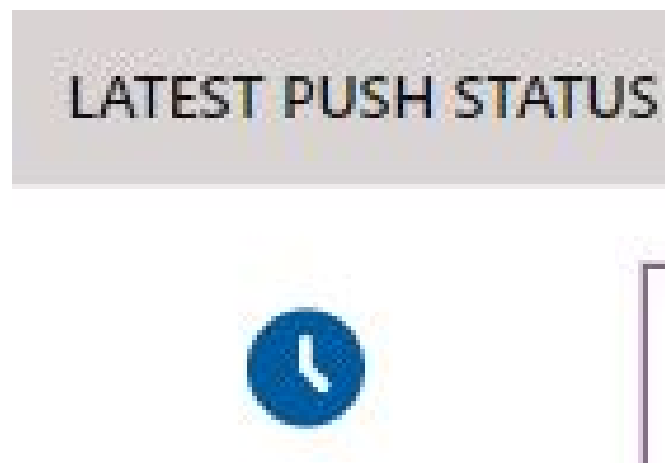
- **-o** is the name of your DevOps Organization
- **-p** is the name of the Azure DevOps Project
- **-d** is the name of the solution branch to create

- If the repository you want to create a branch for is empty you will need to commit an initial commit before a branch can be created.
5. Open ALM Accelerator for Advanced Maker Application
 6. Select Push change to Git
 - a. Create New Branch e.g. MySolution1-WIP
 - b. From existing Solution Branch created above e.g. MySolution1
 - c. Add a comment e.g. Initial version
 7. Click on Latest Push Status
 - a. Permit permissions for pipeline to run (Variable Group, Service Connection, Pipeline)

Post Setup Checks

After setting up an advanced maker you may need to verify the following

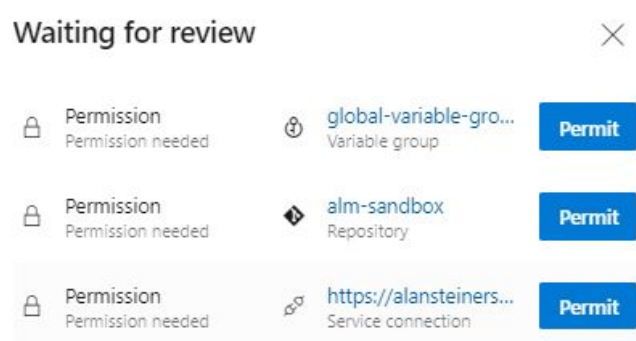
1. If this is your first branch created you will need to check variables applied for the created pipeline
2. The first time that each pipeline is run from the administration application you will need to open the pipeline in Azure DevOps and approve the resources used by the pipeline.
3. Select the blue icon for the Azure DevOps Build in the application



4. Check if is message similar to the following that requires approval of the pipeline to run



5. If required select "View" and permit the build pipeline to access the required resources (Variable group, Repository and Service Connection)



NOTES:

1. If you are using a free Azure Subscription you may receive error "No hosted parallelism has been purchased or granted.". To resolve this issue visit <https://aka.ms/azpipelines-parallelism-request> to request Azure Pipeline build compute

Read Next

- Complete the [Install Overview](#)

4.1 Center of Excellence - Command Line Interface Help

The Center of Excellence Command (CoE) CLI help command provide further information on how to use the provided functionality.

For more information on the CoE CLI you can visit <https://aka.ms/coe-cli>

You can use the **--help** command line argument to discover help for each command and the parameters available.

Examples

To get help for ALM Accelerator for Advanced Makers (AA4AM)

```
coe help aa4am
```

Read more on [ALM Accelerator for Advanced Makers Help](#)

Contribution

The help documents can be updated by making contributions to the markdown files in the **/coe-cli/docs/help** folder of <https://aka.ms/coe-cli>

The help commands are also included in the CoE CLI e-book [Read More](#)

5. ALM Accelerator for Advanced Makers

The ALM Accelerator for Advanced Makers (AA4AM) command line actions allows you to install and configure Azure Active Directory, Azure DevOps and Power Platform.

5.1 Examples

Generate

Creates configuration files for the install command

```
coe help aa4am generate
```

Help for [generate](#)

Install

Allows Administrators to install and configure key components

```
coe help aa4am install
```

Help for [install](#)

Connection

Allows DevOps Administrators to create connections from Azure DevOps to Power Platform environments

```
coe help aa4am connection add
```

Help for [connection add](#)

Maker

Allows administrators to add makers to AA4AM

```
coe help aa4am maker
```

Help for [maker add](#)

Branch

Allows Advanced makers to add solution branches to an Azure DevOps repository

```
coe help aa4am branch
```

Help for [branch](#)

5.1 Generate

The generate command allows you to create JSON files for the install command using an interactive set of questions.

Examples

Install

```
coe help aa4am generate install
```

Help for [generate install](#)

Maker Add

```
coe help aa4am generate maker add
```

Help for [generate maker add](#)

5.1 Install Generate

Description

Generate install configuration file for installation of the ALM Accelerator for Advanced Makers (AA4AM)

Examples

Example command line

```
coe aa4am generate install -o install.json
```

To use the generated install file

```
coe aa4am install -f install.json
```

Read more on [install command](#)

Parameters

-o, --output

Optional name of the output JSON file to be created. If **-o** argument not found JSON results will be displayed in the console

-s, --includeSchema

Defaults to **true** to reference and copy JSON schema in the same folder as the --output file.

If **false** then it will not reference or create the schema file

5.1 AA4AM Install Help

The ALM Accelerator for Advanced Makers (AA4AM) install command allows components to be installed

Examples

```
coe aa4am install -f install.json
```

To generate an install file you can use the [generate install](#) command

```
coe aa4am generate install -o install.json
```

Parameters

-l, --log

The level of logging to be created when running the install in the log file combined.log

Read more <https://github.com/winstonjs/winston#logging-levels>

-f, --file

The install configuration parameters file. Example command to generate install configuration file

```
coe aa4am generate install -o install.json
```

-c, --components

The component(s) to install

- **all** - All components required from Azure Active Directory, Azure DevOps and Power Platform environment
- **aad** - Install and configure Azure active directory components only
- **devops** - Install and configure Azure DevOps components only
- **environment** - Install and configure Power Platform components only

--a, --aad

The Azure Active Directory Application name that will be used as Service Principal to connect from Azure DevOps via Service Connectors to Power Platform Environments.

The application will be created if it does not exist if component **aad** or **all** is selected. Requires Azure Active Directory permissions. [Read More](#)

For **aad** or **devops** install the AAD application will be assigned as the identity to for the service connection.

For **aad** or **environment** install the AAD application will be assigned as the Application User in the Power Platform environment with System Administrator privileges to allow import/export of solutions.

-g, --group

The Azure Active Directory Group name that will be used to grant access to Advanced makers in Azure DevOps and the Power Platform.

The group will be created if it does not exist if component **aad** or **all** is selected.

For **aad** or **devops** the group will be used to grant access to Variable group used by Azure DevOps pipelines.

For **aad** or **environment** the group will be used to share access to run the Canvas application.

-o, --devOpsOrganization

The Azure DevOps organization that will be installed to or referenced.

The value can be in the format <https://dev.azure.com/contoso> or contoso. If the fully qualified Url is not specified then <https://dev.azure.com/> will be inserted before the provided value.

-p, --project

The Azure DevOps project name. The project must already be created in your Azure DevOps organization. This value will be used to:

- Create Variable Group
- Import Azure DevOps pipeline templates
- Create Azure DevOps pipelines to import, export and delete solutions in the Power Platform environments

The default value is **alm-sandbox**

-r, --repository

The Azure DevOps repository where Azure DevOps pipeline templates will be cloned from <https://github.com/microsoft/coe-alm-accelerator-templates.git>

The default value is **pipelines**

-e, --environments

The Power Platform environment where the ALM Accelerator for Advanced Maker solution will be installed. You can enter either the

1. Organization name e.g. contoso-dev1234. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-dev1234.crm.dynamics.com>

You can visit <https://aka.ms/ppac> to list environments that you have access to.

-s, --settings

A dictionary of settings used by the install process

--validation

The name of the build validation environment that will be used to validate pull requests before committing changes to a solution branch.

The value can be either the

1. Organization name e.g. contoso-validation. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-validation.crm.dynamics.com>

--test

The name of the test environment that solutions will be promoted to

The value can be either the

1. Organization name e.g. contoso-test. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-test.crm.dynamics.com>

--prod

The name of the production environment that solutions will be promoted to

The value can be either the

1. Organization name e.g. contoso-prod. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-prod.crm.dynamics.com>

--createSecret

Determine if secrets should be created and assigned to resources that require Service Principal Name (SPN) authentication for **aad** parameter. To create a secret for an Azure Active Directory application you must be assigned as an owner of the Azure Active Directory application

Default value is **true**.

Read more on [Manage user assignment for an app in Azure Active Directory](#)

--region

The region that environments are deployed to. This setting will be used if a fully qualified domain name is not supplied. The default value is **NAM** North America

Further reading:

- [Regions overview](#)
- [Region List](#)

-m, --importMethod

The method to import the managed ALM Accelerator for Advanced Makers into Power Platform environment.

- **api** - The default option that uses the Power Platform APIs to import the solution and fix Dataverse and custom connectors
- **browser** - Indicates that a manual install will be done. The CoE CLI will download the latest release from GitHub for you to manually install.
- **pac** - Install using the Power Platform CLI. See <https://docs.microsoft.com/en-us/powerapps/developer/data-platform/powerapps-cli#solution> for the **pac solution import** and install requirements

--endpoint

The Power Platform Administration environment to interact with. The default value is **prod**

- **prod** - The current production deployment for the commercial cloud
- **usgov** - US Government cloud deployments
- **usgovhigh** - US Government cloud deployments
- **dod** - US Government cloud deployments
- **china** - China regional deployments
- **preview** - Preview environment
- **tip1** - Testing environment for Microsoft internal use
- **tip2** - Testing environment for Microsoft internal use

Read More

- [Microsoft Power Apps US Government](#)
- [Power Apps operated by 21Vianet and Power Automate operated by 21Vianet](#)
- [Power Apps Preview Program](#)

- [What is a CDS endpoint](#)

--subscription

The name of the Azure Active Directory that should be selected. This value is optional in the case you only have access to a single Azure Active Tenant.

If your user account has access to multiple subscriptions this parameter allows you to specify which Azure subscription and tenant should be selected.

The Azure CLI [az account set](#) can be used to set the active subscription.

5.1 Maker Add Generate

Description

Generate configuration file to add an Advanced Maker to an installed the ALM Accelerator for Advanced Makers (AA4AM)

Examples

Example command line

```
coe aa4am generate maker add -o user.json
```

To use the generated configuration file

```
coe aa4am maker add \  
-f user.json
```

Read more on [maker add command](#)

Parameters

-o, --output

Optional name of the output JSON file to be created. If **-o** argument not found JSON results will be displayed in the console

5.1 ALM Accelerate for Advanced Makers Add

Description

Add an Advanced Maker to Azure DevOps and share the canvas application with the user

Example

```
coe aa4am maker add \  
-f user.json
```

Parameters

-f, --file

The install configuration parameters file from. Refer to the [coe aa4am generate maker](#) for more information.

-o, --devOpsOrganization

The Azure DevOps organization that will be installed to or referenced.

The value can be in the format <https://dev.azure.com/contoso> or contoso. If the fully qualified Url is not specified then <https://dev.azure.com/> will be inserted before the provided value.

-p, --project

The Azure DevOps project name. The project must already be created in your Azure DevOps organization. This value will be used to:

- Create Variable Group
- Import Azure DevOps pipeline templates
- Create Azure DevOps pipelines to import, export and delete solutions in the Power Platform environments

-u, --user

The User Principal Name (UPN) of the maker e.g. alan-s@contoso.com

-e, --environment

The Power Platform development environment for the Advanced Makers. You can enter either the

1. Organization name e.g. contoso-alans-dev. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-alans-dev.crm.dynamics.com>

You can visit <https://aka.ms/ppac> to list environments that you have access to.

-g, --group

The Azure Active Directory (AAD) Makers group created during install. This user group will allow the Advanced Maker to access the Variable Group required to run Azure DevOps pipelines and share access to the Advanced Maker administration application.

Note the user running this command must be the creator or owner of the AAD group. The [Add or remove group owners in Azure Active Directory](#) provides more information.

--aad

The Azure Active Directory service principal application created during install. The user will be used to create the service connection to a advanced maker development environment.

Note the user running this command must be the creator or owner of the AAD application. The [Manage user assignment for an app in Azure Active Directory](#) provides more information.

--endpoint

The Power Platform Administration environment to interact with. The default value is **prod**

- **prod** - The current production deployment for the commercial cloud
- **usgov** - US Government cloud deployments
- **usgovhigh** - US Government cloud deployments
- **dod** - US Government cloud deployments
- **china** - China regional deployments
- **preview** - Preview environment
- **tip1** - Testing environment for Microsoft internal use
- **tip2** - Testing environment for Microsoft internal use

Read More

- [Microsoft Power Apps US Government](#)
- [Power Apps operated by 21Vianet and Power Automate operated by 21Vianet](#)
- [Power Apps Preview Program](#)
- [What is a CDS endpoint](#)

--settings

--createSecret

Determine if secrets should be created and assigned to resources that require Service Principal Name (SPN) when adding service connection. To create a secret for an Azure Active Directory application you must be assigned as an owner of the Azure Active Directory application

Default value is **true**.

Read more on [Manage user assignment for an app in Azure Active Directory](#).

--region

The region that environments are deployed to. This setting will be used if a fully qualified domain name is not supplied. The default value is **NAM** North America

Further reading:

- [Regions overview](#)
- [Region List](#)

5.1 AA4AM Advanced Maker Solution Branch

The ALM Accelerator for Advanced Makers (AA4AM) branch command allows Advanced Makers to create a branch in the source code repository to store and build a Power Platform solution. To run the branch command the repository that is being branched must exist and be initialized with an initial commit

The process will

1. Create a source code git branch
2. Create build pipelines to validate, test and move to production

Examples

```
coe aa4am branch \  
  -o https://dev.azure.com/contoso \  
  -p alm-sandbox \  
  -r Operations \  
  -d MyTestSolution \  
  -s validation=https://contoso-validation.crm.dynamics.com, test=http
```

Parameters

-o, --devOpsOrganization

The Azure DevOps organization that will be installed to or referenced.

The value can be in the format <https://dev.azure.com/contoso> or contoso. If the fully qualified Url is not specified then <https://dev.azure.com/> will be inserted before the provided value.

-p, --project

The Azure DevOps project name. The project must already be created in your Azure DevOps organization. This value will be used to:

- Create Variable Group
- Import Azure DevOps pipeline templates
- Create Azure DevOps pipelines to import, export and delete solutions in the Power Platform environments

The default value is **alm-sandbox**

-d, --destination

The destination solution branch name to create

-r, --repository

The Azure DevOps repository to create the branch in. If the name of the repository is not supplied then it will be assumed to be the name of the project.

--source

The source branch to copy from. If not supplied assume that a new branch is being created

--source-build

The source build to copy build variable from. If not supplied the process will

1. Attempt to copy values for validation, test and production from other solution branch build variables
2. Get values from Variable Groups
3. Use placeholder values that must be manually updated by a Build Administrator

-s, --settings

The optional settings. Can be used to set the service connections for the build pipelines **validation**, **test** and **prod**

5.1 ALM Accelerate for Advanced Connection Add

Description

Add an Service Connection to a Power Platform environment into Azure DevOps

The user that executes this command will need the following permissions

- Azure Active Directory application owner
- Azure DevOps Project Administrator

Example

```
coe aa4am connection add \  
  -o https://dev.azure.com/contoso  
  -p alm-sandbox  
  -e https://contoso-test.crm.dynamics.com
```

Parameters

-o, --devOpsOrganization

The Azure DevOps organization that will be installed to or referenced.

The value can be in the format <https://dev.azure.com/contoso> or contoso. If the fully qualified Url is not specified then <https://dev.azure.com/> will be inserted before the provided value.

-p, --project

The Azure DevOps project name. The project must already be created in your Azure DevOps organization. This value will be used to:

- Create Variable Group
- Import Azure DevOps pipeline templates
- Create Azure DevOps pipelines to import, export and delete solutions in the Power Platform environments

-e, --environment

The Power Platform development environment for the Advanced Makers. You can enter either the

1. Organization name e.g. contoso-test. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-test.crm.dynamics.com>

You can visit <https://aka.ms/ppac> to list environments that you have access to.

--aad

The Azure Active Directory service principal application created during install. The user will be used to create the service connection to an advanced maker development environment.

Note the user running this command must be the creator or owner of the AAD application. The [Manage user assignment for an app in Azure Active Directory](#) provides more information.

-u, --user

The optional User Principal Name (UPN) to allow access to the created connection e.g. alan-s@contoso.com

--endpoint

The Power Platform Administration environment to interact with. The default value is **prod**

- **prod** - The current production deployment for the commercial cloud
- **usgov** - US Government cloud deployments
- **usgovhigh** - US Government cloud deployments
- **dod** - US Government cloud deployments
- **china** - China regional deployments
- **preview** - Preview environment
- **tip1** - Testing environment for Microsoft internal use
- **tip2** - Testing environment for Microsoft internal use

Read More

- [Microsoft Power Apps US Government](#)
- [Power Apps operated by 21Vianet and Power Automate operated by 21Vianet](#)
- [Power Apps Preview Program](#)
- [What is a CDS endpoint](#)

--settings

--createSecret

Determine if secrets should be created and assigned to resources that require Service Principal Name (SPN) when adding service connection. To create a secret for an Azure Active Directory application you must be assigned as an owner of the Azure Active Directory application

Default value is **true**.

Read more on [Manage user assignment for an app in Azure Active Directory](#)

--region

The region that environments are deployed to. This setting will be used if a fully qualified domain name is not supplied. The default value is **NAM** North America

Further reading:

- [Regions overview](#)
- [Region List](#)

5.1 ALM Accelerate for Advanced Makers User Add

Description

Add the Azure Active Directory User as a System administrator in a Power Platform environment.

Example

```
coe aa4am user add \  
-e https://contoso-alans-dev.crm.dynamics.com
```

Parameters

-e, --environment

The Power Platform development environment to add the Azure Active Directory as a System Administrator. You can enter either the

1. Organization name e.g. contoso-alans-dev. The **region** parameter will be used to create the full qualified domain name
2. The fully qualified domain name with regional deployment e.g. <http://contoso-alans-dev.crm.dynamics.com>

You can visit <https://aka.ms/ppac> to list environments that you have access to.

--aad

The Azure Active Directory service principal application created during install. The user will be used to create the service connection to an advanced maker development environment.

Note the user running this command must be the creator or owner of the AAD application. The [Manage user assignment for an app in Azure Active Directory](#) provides more information.

-r, --role

The user role to assign. The default value is **System Administrator**

--settings

--region

The region that environments are deployed to. This setting will be used if a fully qualified domain name is not supplied. The default value is **NAM** North America

Further reading:

- [Regions overview](#)
- [Region List](#)

6. CLI Development

This section outlines the following key sections on the CoE CLI. The information below can help guide you in making contributions back to the open source repository.

- [Quick Start](#) - Provides set of links of the technology used to build and test the CLI
- [Contributions](#) - Links to wider contributor guidance
- [Development Environment Setup](#) - How to edit the CLI using Visual Studio Code
- [Adding A New Command](#) - How to add a new CLI command
- [Documentation](#) - Adding documentation for commands

6.1 Quick Start

The coe-cli command line application makes use of following components

1. [NodeJS](#) to provide cross platform support
2. TypeScript to leverage published type definitions for dependent components
3. Jest for unit tests.

Initial Commands

1. Change to co-cli folder

```
cd coe-cli
```

2. Install dependent components

```
npm install
```

3. Build from the source code

```
npm run build
```

4. Install the coe command

```
npm link
```

6.2 Understand The Concepts

Documentation

Documentation is critical for users of the CoE understanding the commands. The [documentation](#) pages describe how to add or modify CoE CLI, the generated [E-Book](#) and associated help documentation.

Development Frameworks

If you are new to TypeScript the following links may help

- [Typescript docs](#)
- [Getting Started with TypeScript - Learning Module](#)

If you are new to unit testing with Jest you can start with

- [Jest getting started](#)

Contributions

Review the general [Contribution Guidance](#).

Development Environment Setup

You can edit and debug the cli using Visual Studio Code

1. If you do not have Visual Studio Code you can visit <https://code.visualstudio.com/Download>
2. Once installed Open the coe-cli folder in Visual Studio Code
3. The [.vscode/launch.json](#) file contains a preconfigured debug launch command
4. You can edit the [sample.json](#) file to the commands that you want to debug
5. Place breakpoints in the TypeScript files you want to debug and Press F5 to start debugging

NOTES:

- Depending on the command you want to debug you may be prompted to login in the DEBUG CONSOLE
- If you are testing with a different account you will need to log out of any existing Azure CLI sessions

```
az logout
```

Debugging Commands

You can debug the coe-cli application commands using Visual Studio Code.

1. Change the sample.json to the command or commands you want to run
2. Open the coe-cli folder in Visual Studio Code
3. Place breakpoints in the TypeScript code you want to debug
4. Press F5 or Select Run -> Start Debugging

6.1 Documentation

Documentation is key to understanding how the cli works. As new commands are added consider the following

- [Add Markdown Pages](#) - Describe the functionality for end users
- [Recording Command Line](#) to demonstrate process

Add Markdown Pages

Add new pages to [docs](#) that describes the new command and how it is expected to be used.

Consider adding the following to the page

1. Static images that summarize the operation
2. [Recording Command Line](#) to demonstrate process

Diagrams

For decision trees the diagrams are generated via [mermaid](#).

Example diagrams:

- [Maturity](#)
- [AA4AM Decision Tree](#)

Diagram Styles

To style the diagram **sample.mmd**

1. Use subgroup styles

```
graph
  subgraph journey[Journey]
    start(Start) --> finish(Finish Here)
  end

  style journey fill:transparent,stroke:green,stroke-width:2px
```

2. Apply CSS styles with a file in the same folder. For example in the above create **sample.css** in same folder as the mmd file

```
#L-start-finish path {
  stroke: red
}
```

NOTES:

1. Review the [node shapes](#) to control the symbols displayed
2. When css file is requested a svg file will also be generated so that you can look at the class and SVG elements generated to apply css styles
3. Circles can be styled using following approach

```
graph TD
  subgraph Journey
```

```
start(Start Here) ---> finish((End Here))
end
```

To color the end circle green using css [starts with selector](#). This is required as each item will have a unique id assigned by mermaid.

```
[id^=flowchart-finish] circle {
  stroke: green;
  fill: lightgreen
}
```

4. Coloring a line. Each line will have the format L-start-finish for the **path** which is the line and the class **.path** which is the line around the arrow head.

```
#L-start-finish path,
#L-start-finish .path
{
  stroke: green;
}
```

Update Diagrams

The static images for each diagram is generated as follows

1. Change to coe-cli folder

```
cd coe-cli
```

2. Generate static files

```
npm run diagrams
```

Add Help Pages

Add new help pages to [help](#) that provides detailed information on the command and options. Help can be accessed using the help command which will display the associated help markdown file in the browser

```
coe help aa4am
```

The command above will display the contents of [help/aa4am/readme.md](#)

Recording Command Line

To include a short animated recording of commands and the expected output you can use the following process

- A. Install termtosvg in a Unix based terminal

```
pip3 install --user termtosvg
```

This process will work cross platform and any of the following options could be used:

- i) Native Unix shell on MacOS or Linux distributions
- ii) Docker images with a Unix shell
- iii) [Windows Subsystem for Linux](#) on Windows

B. Record the session to a cast file

```
termtosvg record test.cast
```

The generated cast file is a simple text file that can be edited with any text editor.

C. Remove pauses using [term-trim.ps1](#)

```
./term-trim.ps1 -Input test.cast -Output test2.cast -Trim 1
```

D. Generate the svg file

```
termtosvg render test2.cast test.svg -t window_frame
```


6.1 Adding a new Command

To add a new sample command you can use the following command to template the initial setup of the TypeScript command and the jest unit test.

```
cd coe-cli
coe cli add -n sample
```

Connecting the command to the command line

Once you have unit test completed for your new command

1. Review <https://www.npmjs.com/package/commander> on commands, options
 2. Update [commands.ts](#) to include a new command or sub command
- Import your files at the top of the file

```
import { SampleArguments, SampleCommand } from './sample';
```

- Add function for mock injection

```
createSampleCommand: () => SampleCommand
```

- Create command in the constructor function

```
this.createSampleCommand = () => new SampleCommand
```

- Add function

```
AddSampleCommand(program: commander.Command) {
    var run = program.command('sample')
        .description('A new sample command')
        .option('-c, --comment <comment>', 'The comment for the command')
        .action(async (options: any) : Promise<void> => {
            let args = new SampleArguments();
            args.comment = options.comment;
            let command = this.createSampleCommand();
            await command.execute(args)
        });
}
```

- Register new command to init function

```
this.AddSampleCommand(program);
```

3. Update [commands.spec.ts](#) to include unit tests
- Include reference to the command

```
import { SampleCommand } from '../../src/commands/sample'
```

- Add a set of Jest tests

```
describe('Sample', () => {  
  test('Execute', async () => {  
    // Arrange  
    var commands = new CoeCliCommands();  
    let mockSampleCommand = mock<SampleCommand>();  
  
    commands.createSampleCommand = () => { return mockSampleCommand;  
  
    mockSampleCommand.execute.mockResolvedValue()  
  
    // Act  
    await commands.execute(['node', 'commands.spec', 'sample', '-c'  
  
    // Assert  
    expect(mockSampleCommand.execute).toHaveBeenCalled()  
  })  
});
```

4. Run the unit tests with new changes

```
npm run test
```


6.1 E-Book

The CoE CLI command packages up the documentation and associated help files as a single e-book that can be distributed a PDF file. This approach allows offline consumption of the documentation and ordering of the content to help the reader consume the content.

Quick Start

1. Change to the coe-cli docs folder

```
cd coe-cli/docs
```

2. Build the required docker image

```
docker build -t cli-mdbook .
```

2. Create the e-book using PowerShell

```
npm run ebook
```

Understand the Concepts

The generated e-book approach has the following main components

1. A cover page [ebook-cover.png](#) image and table of contents are automatically as first pages of the e-book
2. A [index.txt](#) file that controls the reading order of the content
3. The e-book style sheet [book.css](#)
4. Prism.js stylesheet [prism.css](#) and JavaScript [prism.js](#) required for formatting of the code samples
5. **build.sh** shell script to create the PDF document

E-Book Update

To update this e-book with new content you will need access to docker to generate new versions of the pdf file using the following steps

1. Change to the docs folder

```
cd coe-cli/docs
```

2. Build e-book generation docker image

```
docker build -t cli-mdbook .
```

3. Generate a new version of the e-book using PowerShell and the docker image created above.

```
npm run ebook
```

4. The script will generate a file named **Power Platform CoE Toolkit Command Line Interface.pdf** in the docs folder

Notes:

- If you are building an e-book for a branch that is not the main branch you can specify the path to the git hub repository e.g.

```
coe ebook generate \  
-r https://github.com/microsoft/coe-starter-kit/tree/coe-cli/coe-cl
```

Spell Check Process

The [build.sh](#) makes use of [mdspell](#) to spell check the markdown files using US english (en-US).

The mdspell node application is installed inside the [cli-mdbook docker image](#) and executed by the [build.ps1](#). If you need to add an exclusion for a work you can add changes to [.spelling](#) file. Each word or phrase should be on a separate line.

Grammar Checks

The [LanguageTool](#) writing assistant uses the Java bases command line tool inside the docker image to perform duplication, grammar and typographical checks using en-US. The language tool integration is implemented in [grammar.js](#). The [grammar-ignore.txt](#) file contains a list of rules that will be ignored.

E-Book Customization

The [ebook.ts](#) controls the process of

1. Parsing the markdown files
2. Combining the markdown files into a single HTML file
3. Updating references between files and within a markdown file
4. Adding links between files
5. Referencing links to non markdown links to reference the GitHub project