DevOps

Week 03

Murtaza Munawar Fazal

Pipeline

- Business demands continuous delivery of value. Value is created only when a product is delivered to a satisfied customer.
- It's not created when one silo in the process is completed.
- The core idea is to create a repeatable, reliable, and incrementally-improving process for taking software from concept to customer.
- The goal is to enable a constant flow of changes into production via an automated software production line.
- Think of it as a pipeline. The pipeline breaks down the software delivery process into stages.
- Each stage aims to verify the quality of new features from a different angle to validate the new functionality and prevent errors from affecting your users.
- The pipeline should provide feedback to the team. Also, visibility into the changes flows to everyone involved in delivering the new feature(s).

Pipeline

- A delivery pipeline enables the flow of more minor changes more frequently, with a focus on flow.
- Your teams can concentrate on optimizing the delivery of changes that bring quantifiable value to the business.
- This approach leads teams to continuously monitor and learn where they're finding obstacles, resolve those issues, and gradually improve the pipeline's flow.
- As the process continues, the feedback loop provides new insights into new issues and barriers to be resolved.
- The pipeline is the focus of your continuous improvement loop.
- A typical pipeline will include the following stages:
  - build automation and continuous integration,
  - test automation and
  - deployment automation.

Build automation and continuous integration

- The pipeline starts by building the binaries to create the deliverables passed to the following stages. New features implemented by the developers are integrated into the central code base, built, and unit tested. It's the most direct feedback cycle that informs the development team about the health of their application code.

Test automation

- The new version of an application is rigorously tested throughout this stage to ensure that it meets all wished system qualities. It's crucial that all relevant aspects—whether functionality, security, performance, or compliance—are verified by the pipeline. The stage may involve different types of automated or (initially, at least) manual activities.

Deployment Automation

- A deployment is required every time the application is installed in an environment for testing, but the most critical moment for deployment automation is rollout time.
- Since the preceding stages have verified the overall quality of the system, It's a low-risk step.
- The deployment can be staged, with the new version being initially released to a subset of the production environment and monitored before being rolled out.
- The deployment is automated, allowing for the reliable delivery of new functionality to users within minutes if needed.

Azure Pipeline

- Azure Pipelines is a cloud service that automatically builds and tests your code project and makes it available to other users. It works with just about any language or project type.
- Azure Pipelines combines continuous integration (CI) and continuous delivery (CD) to test and build your code and ship it to any target constantly and consistently.

Azure Pipeline

- Languages
  - You can use many languages with Azure Pipelines, such as Python, Java, PHP, Ruby, C#, and Go.
- Version Control Systems
  - Azure Pipelines integrates with GitHub, GitLab, Azure Repos, Bitbucket, and Subversion.
- Application Types
  - You can use Azure Pipelines with most application types, such as Java, JavaScript, Python, .NET, PHP, Go, XCode, and C++.
- Deployment Targets
  - Container registries.
  - Virtual machines.
  - Azure services, or any on-premises or cloud target such:
    - Microsoft Azure.
    - Google Cloud.
    - Amazon Web Services (AWS).

CI / CD

- Continuous integration is used to automate tests and builds for your project. CI helps to catch bugs or issues early in the development cycle when they're easier and faster to fix. Items known as artifacts are produced from CI systems. The continuous delivery release pipelines use them to drive automatic deployments.
- Continuous delivery is used to automatically deploy and test code in multiple stages to help drive quality. Continuous integration systems produce deployable artifacts, which include infrastructure and apps. Automated release pipelines consume these artifacts to release new versions and fixes to the target of your choice.

Azure Pipelines Key Terms

- Agent
  - When your build or deployment runs, the system begins one or more jobs. An agent is installable software that runs a build or deployment job.
- Artifact
  - An artifact is a collection of files or packages published by a build. Artifacts are made available for the tasks, such as distribution or deployment.
- Build
  - A build represents one execution of a pipeline. It collects the logs associated with running the steps and the test results.

Azure Pipelines Key Terms

- Continuous delivery
  - Continuous delivery (CD) (also known as Continuous Deployment) is a process by which code is built, tested, and deployed to one or more test and production stages. Deploying and testing in multiple stages helps drive quality. Continuous integration systems produce deployable artifacts, which include infrastructure and apps. Automated release pipelines consume these artifacts to release new versions and fix existing systems. Monitoring and alerting systems constantly run to drive visibility into the entire CD process. This process ensures that errors are caught often and early.
- Continuous Integration
  - Continuous integration (CI) is the practice used by development teams to simplify the testing and building of code. CI helps to catch bugs or problems early in the development cycle, making them more accessible and faster to fix. Automated tests and builds are run as part of the CI process. The process can run on a schedule, whenever code is pushed, or both. Items known as artifacts are produced from CI systems. The continuous delivery release pipelines use them to drive automatic deployments.

Azure Pipelines Key Terms

- Deployment Targets
  - A deployment target is a virtual machine, container, web app, or any service used to host the developed application. A pipeline might deploy the app to one or more deployment targets after the build is completed and tests are run.

Azure Pipelines Key Terms

- Job
  - A build contains one or more jobs. Most jobs run on an agent. A job represents an execution boundary of a set of steps. All the steps run together on the same agent. For example, you might build two configurations - x86 and x64. In this case, you have one build and two jobs.
- Pipeline
  - A pipeline defines the continuous integration and deployment process for your app. It's made up of steps called tasks. It can be thought of as a script that describes how your test, build, and deployment steps are run.
- Release
  - When you use the visual designer, you can create a release or a build pipeline. A release is a term used to describe one execution of a release pipeline. It's made up of deployments to multiple stages.

Azure Pipelines Key Terms

- Stage
  - Stages are the primary divisions in a pipeline: "build the app," "run integration tests," and "deploy to user acceptance testing" are good examples of stages.
- Task
  - A task is the building block of a pipeline. For example, a build pipeline might consist of build and test tasks. A release pipeline consists of deployment tasks. Each task runs a specific job in the pipeline.
- Trigger
  - A trigger is set up to tell the pipeline when to run. You can configure a pipeline to run upon a push to a repository at scheduled times or upon completing another build. All these actions are known as triggers.

Continuous Integration (CI)

- Continuous integration (CI) is the process of automating the build and testing of code every time a team member commits changes to version control.
- CI encourages developers to share their code and unit tests by merging their changes into a shared version control repository after every small task completion.
- Committing code triggers an automated build system to grab the latest code from the shared repository and build, test, and validate the entire main branch (also known as the trunk or main).

Continuous Integration (CI)

- The idea is to minimize the cost of integration by making it an early consideration.
- Developers can discover conflicts at the boundaries between new and existing code early, while conflicts are still relatively easy to reconcile.
- Once the conflict is resolved, work can continue with confidence that the new code honors the requirements of the existing codebase.
- Integrating code frequently doesn't offer any guarantees about the quality of the new code or functionality.
- In many organizations, integration is costly because manual processes ensure that the code meets standards, introduces bugs, and breaks existing functionality.

Continuous Integration (CI)

- In practice, continuous integration relies on robust test suites and an automated system to run those tests to address this friction within the integration process.
- When a developer merges code into the main repository, automated processes kick off a build of the new code.
- Afterward, test suites are run against the new build to check whether any integration problems were introduced.
- If either the build or the test phase fails, the team is alerted to work to fix the build.
- The end goal of continuous integration is to make integration a simple, repeatable process part of the everyday development workflow to reduce integration costs and respond to early defects.
- Working to make sure the system is robust, automated, and fast while cultivating a team culture that encourages frequent iteration and responsiveness to build issues is fundamental to the strategy's success.

Four pillars of continuous integration

- Version Control System
  - Git
  - Apache Subversion
  - Team Foundation Version Control
- A package management system is used to install, uninstall, and manage software packages.
  - NuGet
  - Node Package Manager (NPM)
  - Chocolatey
  - HomeBrew
  - RPM

Four pillars of continuous integration

- A continuous integration system merges all developer working copies to shared mainline several times a day.
  - Azure DevOps
  - TeamCity
  - Jenkins
- An automated build process creates a software build, including compiling, packaging, and running automated tests.
  - Apache Ant
  - NAnt
  - Gradle

Demo

AB Agile Planning and Por...

Overview

Summary

Dashboards

Wiki

Boards

Repos

Pipelines

Test Plans

Artifacts

Project settings

AB Agile Planning and Portfolio Management with Azure Boards

Private | Invite

About this project

Like 0

Generated by Azure DevOps Demo Generator

Languages

JavaScript CSS

Project stats

Period: Last 7 days

Boards

0 Work items created

0 Work items completed

Repos

0 Pull requests opened

2 Commits by 1 authors

Pipelines

100% Builds succeeded

0% Deployments succeeded

Members 1

Source control types supported by Azure Pipelines

Repository type	Azure Pipelines (YAML)	Azure Pipelines (classic editor)
Azure Repos Git	Yes	Yes
Azure Repos TFVC	No	Yes
Bitbucket Cloud	Yes	Yes
Other Git (generic)	No	Yes
GitHub	Yes	Yes
GitHub Enterprise Server	Yes	Yes
Subversion	No	Yes

Microsoft Hosted Agent

- If your pipelines are in Azure Pipelines, then you've got a convenient option to build and deploy using a Microsoft-hosted agent.
- With a Microsoft-hosted agent, maintenance and upgrades are automatically done.
- Each time a pipeline is run, a new virtual machine (instance) is provided. The virtual machine is discarded after one use.
- For many teams, this is the simplest way to build and deploy.
- You can try it first and see if it works for your build or deployment. If not, you can use a self-hosted agent.
- A Microsoft-hosted agent has job time limits.

Self Hosted Agents

- An agent that you set up and manage on your own to run build and deployment jobs is a self-hosted agent.
- You can use a self-hosted agent in Azure Pipelines. A self-hosted agent gives you more control to install dependent software needed for your builds and deployments.
- You can install the agent on:
  - Linux.
  - macOS.
  - Windows.
  - Linux Docker containers.
- After you've installed the agent on a machine, you can install any other software on that machine as required by your build or deployment jobs.
- A self-hosted agent doesn't have job time limits.

Job Types

- In Azure DevOps, there are four types of jobs available:
  - Agent pool jobs.
  - Container jobs.
  - Deployment group jobs.
  - Agentless jobs.

Agent Pools

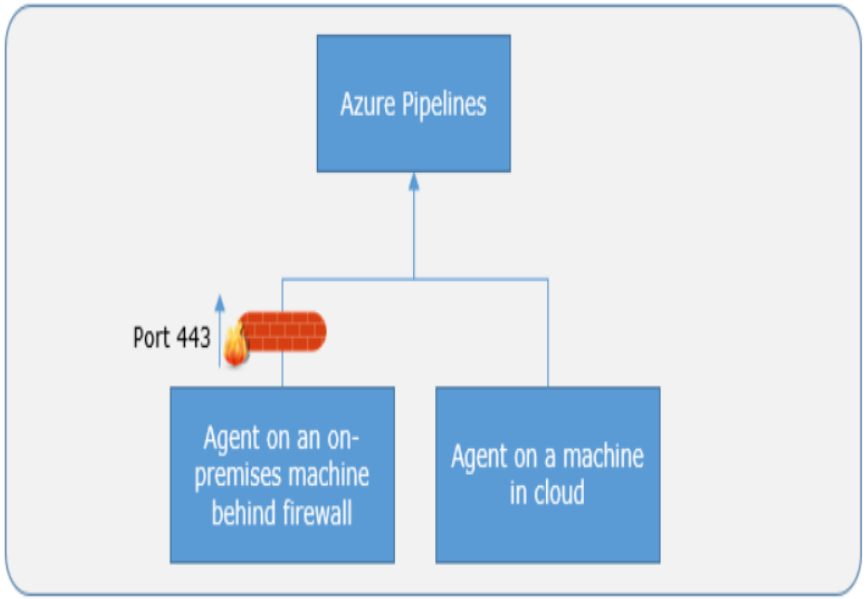
- Instead of managing each agent individually, you organize agents into agent pools. An agent pool defines the sharing boundary for all agents in that pool.
- In Azure Pipelines, pools are scoped to the entire organization so that you can share the agent machines across projects.
- If you create an Agent pool for a specific project, only that project can use the pool until you add the project pool into another project.
- When creating a build or release pipeline, you can specify which pool it uses, organization, or project scope.
- Pools scoped to a project can only use them across build and release pipelines within a project.
- To share an agent pool with multiple projects, use an organization scope agent pool and add them in each of those projects, add an existing agent pool, and choose the organization agent pool. If you create a new agent pool, you can automatically grant access permission to all pipelines.

Predefined Agent Pool

- Azure Pipelines provides a pre-defined agent pool-named Azure Pipelines with Microsoft-hosted agents.
- It will often be an easy way to run jobs without configuring build infrastructure.
- The following virtual machine images are provided by default:
  - Windows Server 2022 with Visual Studio 2022.
  - Windows Server 2019 with Visual Studio 2019.
  - Ubuntu 20.04.
  - Ubuntu 18.04.
  - macOS 11 Big Sur.
  - macOS X Catalina 10.15.

Communicate with Azure Pipelines

- The agent communicates with Azure Pipelines to determine which job it needs to run and report the logs and job status.
- The agent always starts this communication. All the messages from the agent to Azure Pipelines over HTTPS, depending on how you configure the agent.
- This pull model allows the agent to be configured in different topologies



Communicate with Azure Pipelines

- The user registers an agent with Azure Pipelines by adding it to an agent pool. You've to be an agent pool administrator to register an agent in that pool. The identity of the agent pool administrator is needed only at the time of registration.
- Once the registration is complete, the agent downloads a listener OAuth token and uses it to listen to the job queue.
- Periodically, the agent checks to see if a new job request has been posted in the job queue in Azure Pipelines. The agent downloads the job and a job-specific OAuth token when a job is available. This token is generated by Azure Pipelines for the scoped identity specified in the pipeline. That token is short-lived and is used by the agent to access resources (for example, source code) or modify resources (for example, upload test results) on Azure Pipelines within that job.
- Once the job is completed, the agent discards the job-specific OAuth token and checks if there's a new job request using the listener OAuth token.
- The payload of the messages exchanged between the agent, and Azure Pipelines are secured using asymmetric encryption. Each agent has a public-private key pair, and the public key is exchanged with the server during registration.
- The server uses the public key to encrypt the job's payload before sending it to the agent. The agent decrypts the job content using its private key. Secrets stored in build pipelines, release pipelines, or variable groups are secured when exchanged with the agent.

Demo

https://murtaza-fazal.visualstudio.com/Agile%20Planning%20and%20Portfolio%20Management%20with%20Azure%20Boards

Perseus | Personal | MCT | Perseus | Other favorites

Azure DevOps murtaza-fazal / Agile Planning and Portfolio... / Overview / Summary

Search

MF

AB Agile Planning and Por...

- Overview
- Summary
- Dashboards
- Wiki
- Boards
- Repos
- Pipelines
- Test Plans
- Artifacts

Project settings

AB Agile Planning and Portfolio Management with Azure Boards

Private | Invite

About this project

Like 0

Generated by Azure DevOps Demo Generator

Languages

JavaScript CSS

Pipelines
- Pipelines
- Environments
- Releases
- Library
- Task groups
- Deployment groups

Project stats

Period: Last 7 days

Boards

0 Work items created

0 Work items completed

Repos

0 Pull requests opened

3 Commits by 1 authors

Pipelines

100% Builds succeeded

0% Deployments succeeded

Members 1

https://murtaza-fazal.visualstudio.com/Agile Planning and Portfolio Management with Azure Boards/_build