AzureDevOps_Engineering Knowledge Base

Version: 1.0

Maintainer: Engineering Team

Last Updated: 24 Nov 2025

Purpose

This document serves as a centralized, practical reference for engineering teams working with Azure DevOps—covering core components, best practices, troubleshooting playbooks, and automation templates for CI/CD.

Table of Contents

1. Introduction

Azure DevOps is a cloud service suite for planning work, collaborating on code, and building and deploying applications. It integrates Git-based source control, CI/CD pipelines, agile boards, artifacts, and testing to streamline the software delivery lifecycle.

Why use Azure DevOps for engineering workflows?

• Unified toolchain across repositories, builds, releases, and work tracking. • First-class YAML pipelines enable versioned, repeatable CI/CD. • Strong governance via branch policies, approvals, environments, and secrets management.

2. Core Components

Repos

Git-based source control with branch policies, pull requests (PR), and code reviews. Enforce checks like build validation and status checks before merging.

Pipelines

Automate builds, tests, and deployments via YAML or classic pipelines. Use hosted agents (e.g., ubuntu-latest) or self-hosted agents for custom tooling.

Boards

Plan and track work with epics, features, user stories, tasks, and bugs. Customize processes and use queries and dashboards for reporting.

Artifacts

Package management for NuGet, npm, Maven, and Python. Use scoped feeds and upstream sources to control dependencies.

Test Plans

Manual and exploratory testing integrated with pipelines and defect tracking.

3. Best Practices

Branching Strategies

Choose GitFlow for multi-release products or trunk-based development for high-frequency deployments. Apply branch policies: minimum reviewers, linked work items, build validation, and status checks.

Pipeline Hygiene

• Prefer YAML pipelines and reuse templates. • Cache dependencies to speed builds. • Fail fast on tests; publish test results and code coverage. • Use environments with approvals for production gates.

Security & Compliance

Protect secrets with variable groups linked to Azure Key Vault. Use service connections with least privilege. Enable SAST/DAST scans, signed artifacts, and provenance (SBOM) where applicable.

4. Troubleshooting

Common Pipeline Failures

• Agent capabilities mismatch: ensure required tools are installed or use appropriate vmImage. • Credential errors: rotate PATs/secrets and validate service connections. • Permission denied to repo or feed: check project and feed permissions.

Agent Configuration Issues

Confirm agent pool assignment, network egress rules, and proxy settings. Review agent logs under _work/_diag on self-hosted agents.

Authentication Problems

Use Azure AD with conditional access; prefer service principals over PATs; ensure scopes are limited and tokens rotated.

5. Automation Examples (YAML Pipeline)

```
trigger:
- main

pr:
- main

pool:
  vmImage: 'ubuntu-latest'

variables:
  CONFIGURATION: 'Release'

stages:
- stage: Build
  displayName: 'Build and Test'
  jobs:
  - job: BuildJob
    steps:
    - task: UseDotNet@2
      inputs:
        packageType: 'sdk'
        version: '8.x'
    - task: DotNetCoreCLI@2
      inputs:
        command: 'restore'
        projects: '**/*.csproj'
    - task: DotNetCoreCLI@2
      inputs:
        command: 'build'
        projects: '**/*.csproj'
        arguments: '--configuration $(CONFIGURATION)'
    - task: DotNetCoreCLI@2
      inputs:
        command: 'test'
        projects: '**/*Tests/*.csproj'
        publishTestResults: true

- stage: Deploy
  displayName: 'Deploy to Dev'
  dependsOn: Build
  jobs:
  - deployment: DevDeploy
```

```
    environment: 'dev'
    strategy:
      runOnce:
        deploy:
          steps:
          - script: echo 'Deploying to Dev environment...'
```

6. Appendix: Useful Links & Terms

Key Terms: PR (Pull Request), PAT (Personal Access Token), SBOM (Software Bill of Materials), SAST (Static Application Security Testing), DAST (Dynamic Application Security Testing).