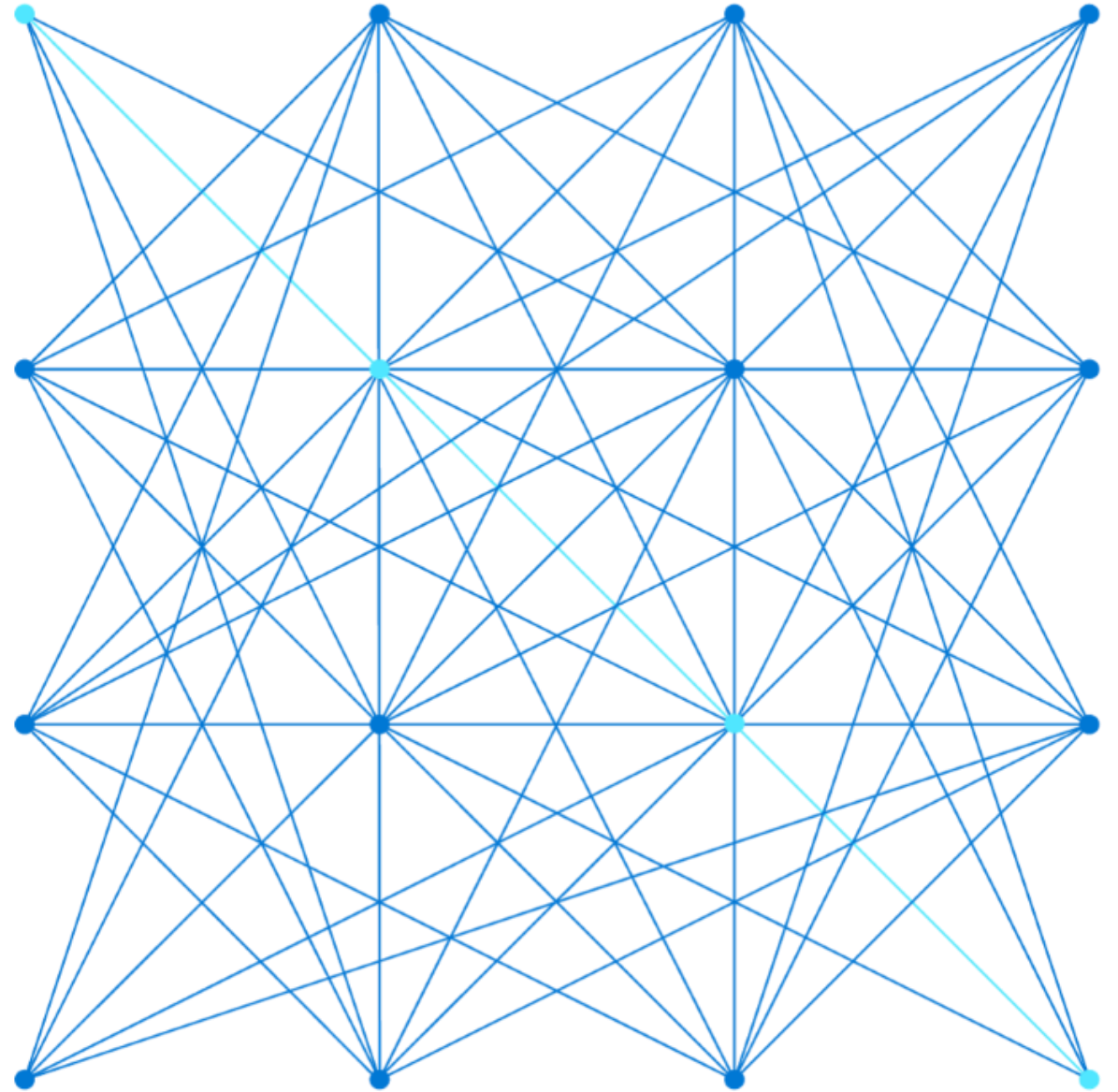


AZ-400.00

Module 5:

Configuring Azure Pipelines



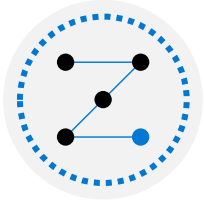
Lesson 01: Module overview



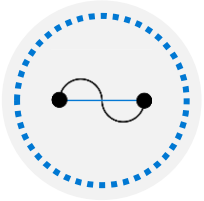
Module overview



Lesson 1: Module overview



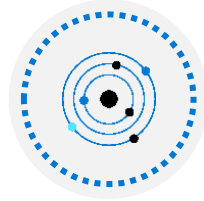
Lesson 2: The concept of pipelines in DevOps



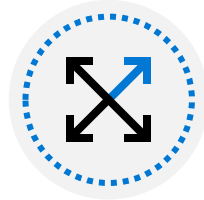
Lesson 3: Azure Pipelines



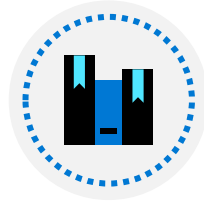
Lesson 4: Evaluate use of hosted versus self-hosted agents



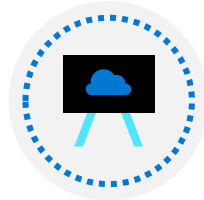
Lesson 5: Agent pools



Lesson 6: Pipelines and concurrency



Lesson 7: Azure DevOps and open-source projects (public projects)



Lesson 8: Azure Pipelines YAML versus Visual Designer

Learning objectives

After completing this module, students will be able to:

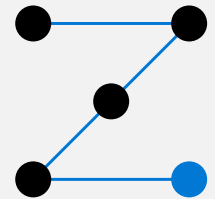


Explain the role of Azure Pipelines and its components

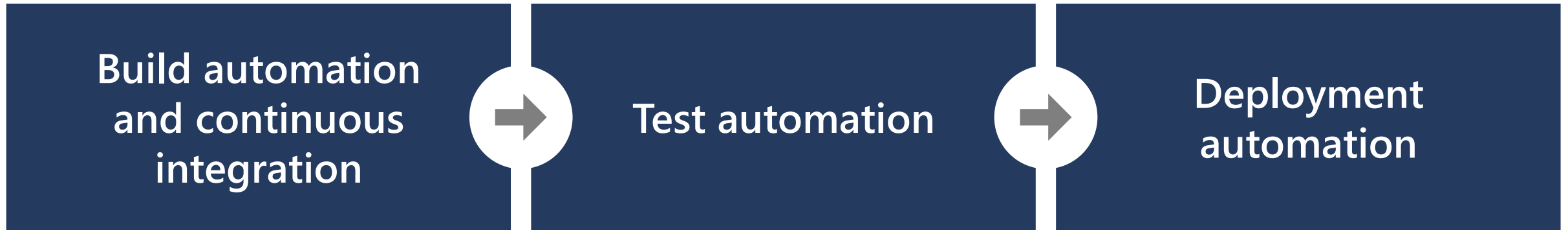


Configure agents for use in Azure Pipelines

Lesson 02: The concept of pipelines in DevOps



The concept of pipelines in DevOps



A pipeline enables a constant flow of changes into production via an automated software production line

Pipelines create a repeatable, reliable and incrementally improving process for taking software from concept to customer

Pipelines require infrastructure, this infrastructure will have a direct impact on the effectiveness of the pipeline

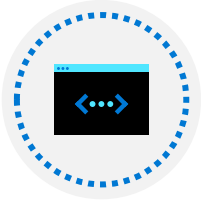
Lesson 03: Azure Pipelines



Azure Pipelines



Azure Pipelines is a cloud service that you can use to automatically **build** and **test** your code project and make it available to other users.



Works great with Continuous Integration and Continuous Delivery:

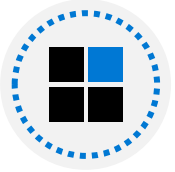
- Work with any language or platform – Python, Java, PHP, Ruby, C#, and Go
- Deploy to different types of targets at the same time (Containers, VMs, K8S..etc)
- Integrate with Azure deployments – Container registries, virtual machines, Azure services, or any on-premises or cloud target (Microsoft Azure, Google Cloud, or AWS)
- Build on Windows, Linux, or macOS machines
- Integrate with GitHub
- Work with open-source projects

(#mw: start with [this picture](#))

Azure Pipelines key terms



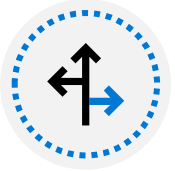
Agent



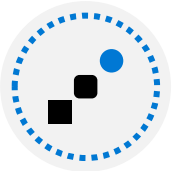
Artifact



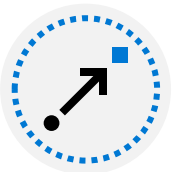
Build



Continuous Delivery (CD)



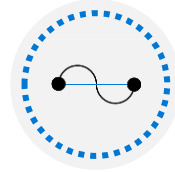
Continuous Integration (CI)



Deployment target



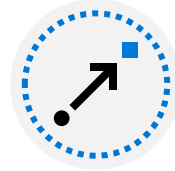
Job



Pipeline



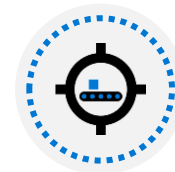
Release



Stage



Task



Trigger

Lesson 04: Evaluate use of Microsoft-hosted vs self-hosted agents



Microsoft-hosted versus self-hosted agents

You generally need at least one agent to build or deploy your project.

An agent is installable software that runs one build or deployment job at a time.

Two types of agents:

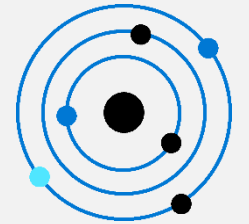


Microsoft-hosted agents – Maintenance and upgrades are automatically done. Each time a pipeline is run, a fresh virtual machine (instance) is provided. There are time limits on jobs run on these agents.



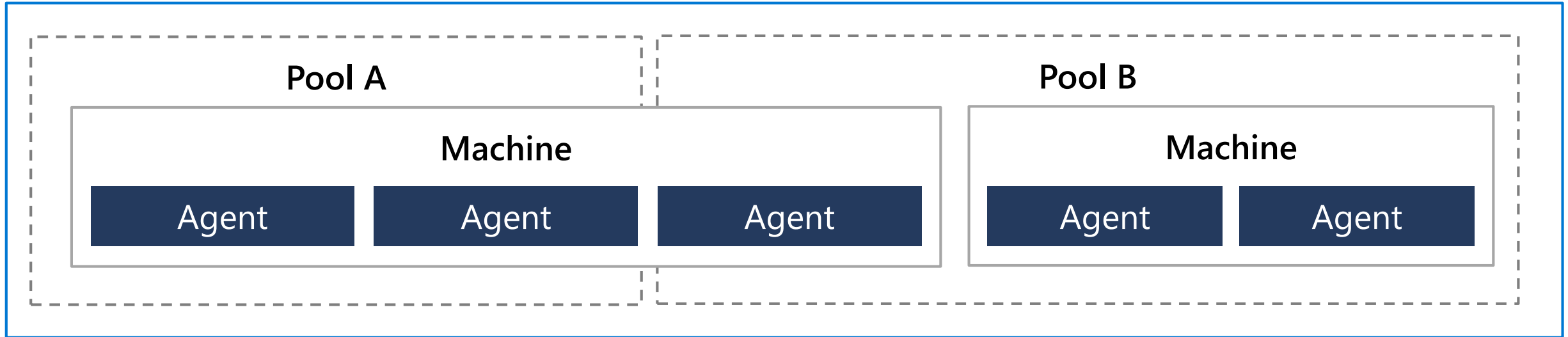
Self-hosted agents – You take care of maintenance and upgrades. Give you more control to install dependent software needed. Can be installed on Linux, macOS, Windows machines, or in a Linux Docker container. There are no time limits on these jobs.

Lesson 05: Agent pools



Agent pools

An **agent** is installable software that runs a build and/or deployment job. (#mw – think of each agent as a process on a server, each agent is running a job)



You can organize agents into agent pools.

An agent pool defines the sharing boundary.

In Azure Pipelines, agent pools are scoped to the Azure DevOps organization; so, **you can share an agent pool across projects.**

Predefined agent pool – Azure pipelines

Hosted VS2019

Hosted VS2017

Hosted Ubuntu 20.04

Hosted Ubuntu 18.04

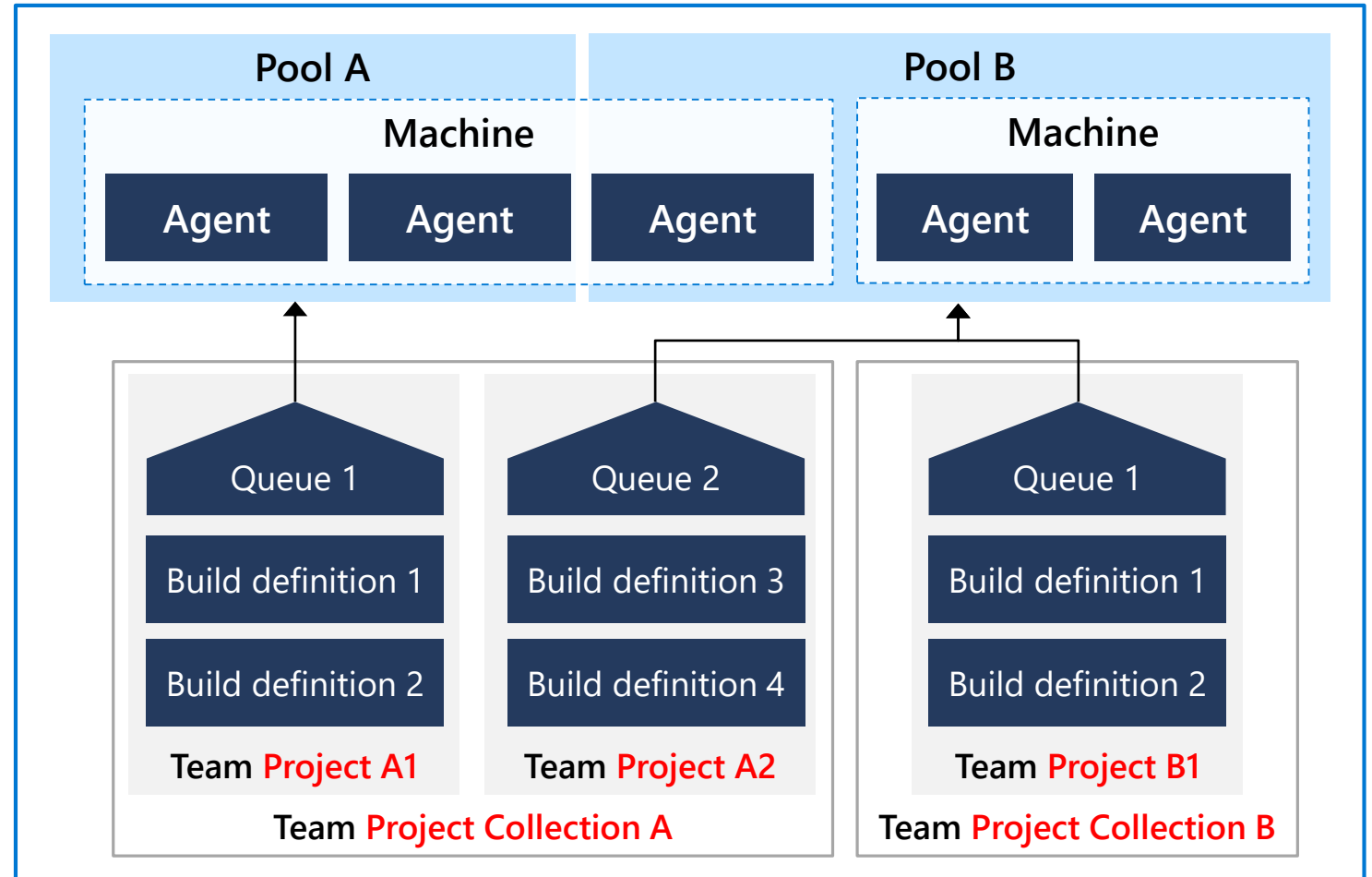
Hosted Ubuntu 16.04

Hosted macOS X Mojave 10.14

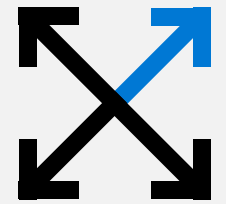
Hosted macOS X Catalina 10.15

Typical situations for agent pools

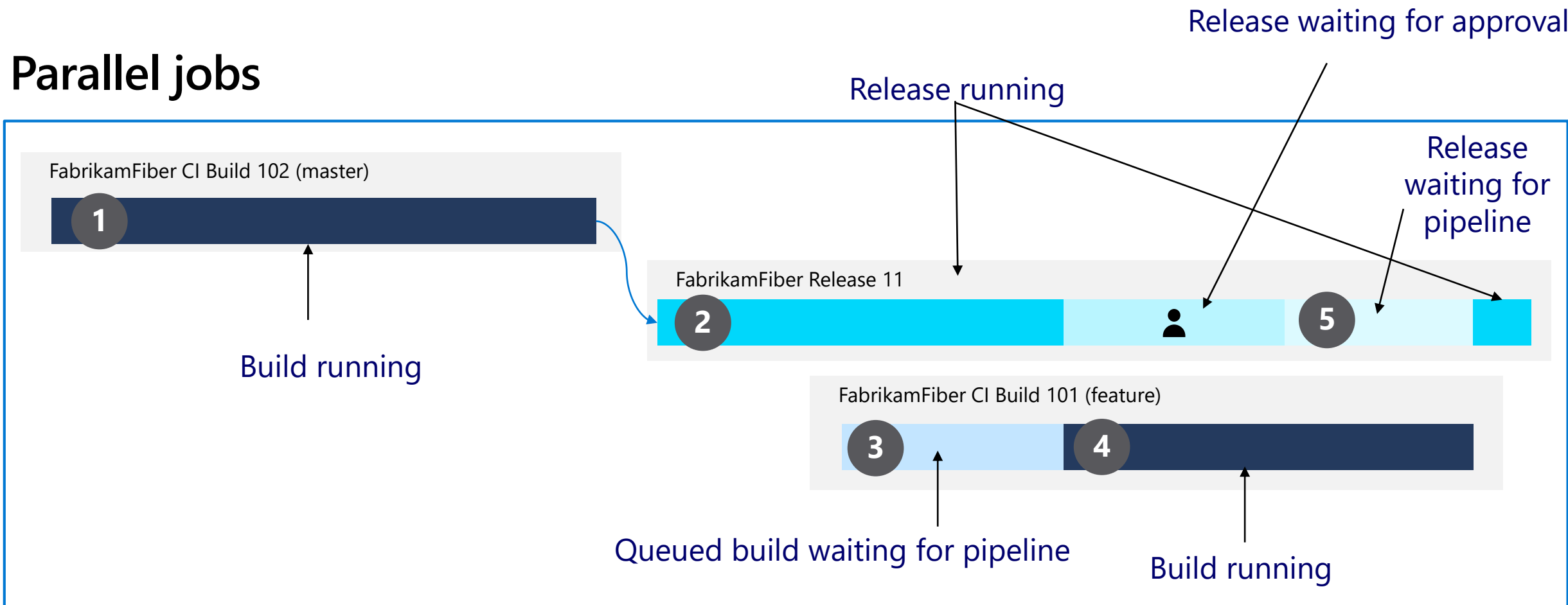
- You're a **member of a project** and you want to use a **set of machines** owned by your team for running build and deployment jobs.
- You're a **member of the infrastructure team** and would like to set up a pool of agents for **use in all projects**.
- You want to **share a set of agent machines with some projects but not all projects**.



Lesson 06: Pipelines and concurrency



Parallel jobs



This is a simple example of a Microsoft-hosted parallel job.

In this example, can only run one build or release job at a time; additional jobs are queued.

A release consumes a parallel job only when it's being actively deployed to a stage.

Estimating parallel jobs

- Roughly you'll need one parallel job for every four to five users in your organization.
- Insufficient Parallel jobs results in longer build and release queues.
- You can buy more. Parallel jobs are purchased at the organization level and are shared by all projects.

Organization Settings > Retention and parallel jobs

> General

> Work

✓ Pipelines

Agent pools

Deployment pools

Retention and parallel jobs

OAuth configurations

Settings **Parallel jobs**

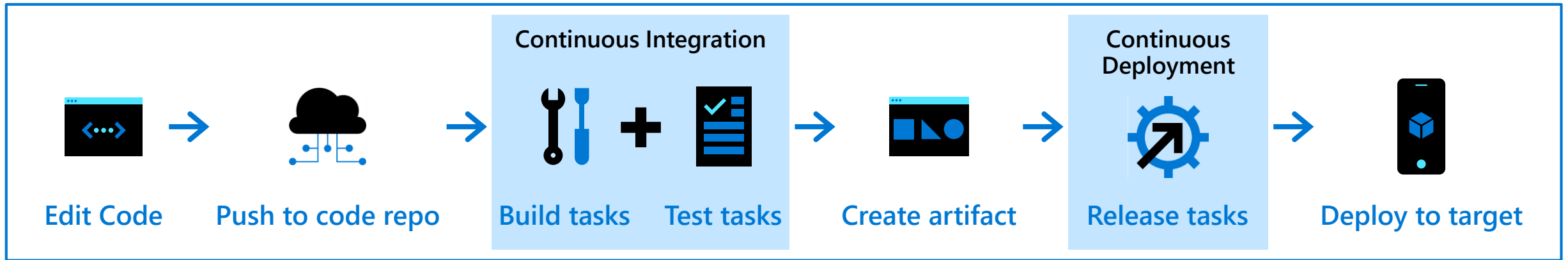
 Microsoft-hosted ⓘ View in-progress jobs	Free tier 1 parallel job up to 1800 mins/mo
Currently 0/1800 minutes are consumed Purchase parallel jobs	
 Self-hosted ⓘ View in-progress jobs	2 Parallel jobs
Free parallel jobs	1
Visual Studio Enterprise subscribers ⓘ	1
Monthly purchases	0 Change

Lesson 08: Azure Pipelines YAML versus Visual Designer



Azure Pipelines and Visual Designer

Configure your pipelines with the Visual Designer



Create and configure your build and release pipelines.

Push your code to your version control repository.

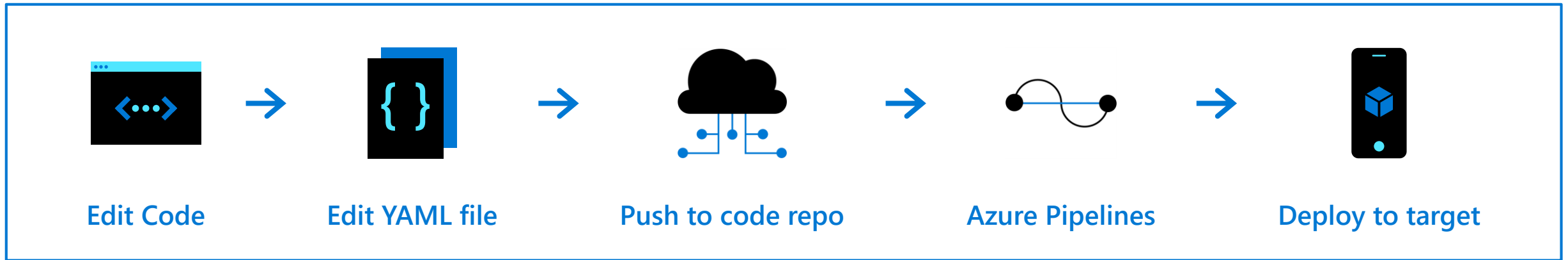
The build creates an artifact that's used by the rest of your pipeline.

Your code is now updated, built, tested, and packaged.

Often referred to as "Classic Pipelines"

Azure Pipelines and YAML

Configure your pipelines in a YAML file that exists alongside your code



Configure Azure Pipelines to use your Git repo.

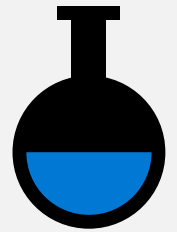
Edit your `azure-pipelines.yml` file to define your build.

Push your code to your version control repository.

Your code is now updated, built, tested, and packaged.

While more code-oriented, defining pipelines using YAML is preferred

Lesson 09: Labs



Lab: Configuring Agent Pools and Understanding Pipeline Styles

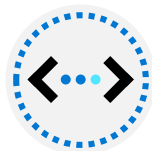


Lab overview:

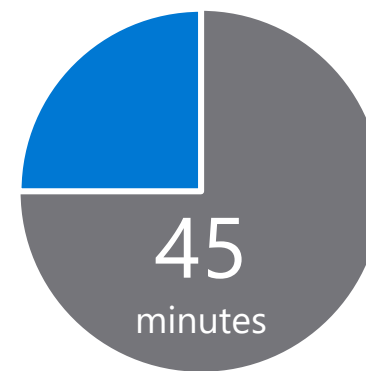
In this lab, you will step through the process of converting a classic pipeline into a YAML-based one and running it first by using a Microsoft-hosted agent and then performing the equivalent task by using a self-hosted agent.

Objectives:

- Implement YAML-based pipelines
- Implement self-hosted agents



Duration:



Lesson 10: Module review and takeaways



What did you learn?



Explain the role of Azure Pipelines and its components



Configure agents for use in Azure Pipelines

Module review questions

1

What are some advantages of Azure Pipelines?

2

What is a pipeline, and why is it used?

3

What are the two types of agents and how are they different?

4

What is an agent pool, and why would you use it?

5

What are two ways to configure your Azure Pipelines?