Contents

[Definition of Done 1](#_Toc55836433)

[Who Defines Definition of Done in Scrum 1](#_Toc55836434)

[When is the right time for creating DOD 1](#_Toc55836435)

[Why DOD 1](#_Toc55836436)

[Definition of Done VS Acceptance Criteria 1](#_Toc55836437)

[Continuous Delivery, Continuous Integration and Continuous Deployment 1](#_Toc55836438)

[Continuous Integration explained 2](#_Toc55836439)

# Definition of Done

The ‘Definition of Done is usually a checklist of all the work that the team needs to do before it can call the product increment as DONE. The DOD will vary across organisations, and it may also differ within the organisation across different teams.

## Who Defines Definition of Done in Scrum

Scrum team define it. They are the ones who will be accountable to meet this definition. In some cases, the DOD can be at an organisation lever or at a product level. So, it is essential that the team creates it and agrees to it.

## When is the right time for creating DOD

The team would need to use this definition in the first sprint when they would mark the first product increment as done. The DOD should be created before the first sprint or latest within the first sprint itself.

## Why DOD

There is not one fit all dod. It ensures that when a product increment is Done, everyone has the same shared understanding what this means. It is necessary to ensure that we have artefact transparency.

## Definition of Done VS Acceptance Criteria

Each backlog item that is being worked in a sprint has a set of Acceptance Criteria that the product owner defines. Also the acceptance criteria validate is also part of DOD.

Acceptance criteria is at a story level. DOD is at the product increment level.

As an example: Each development team member that works on a story will have an acceptance criterion that needs to fulfil. But he man not be able to deploy his story stand-alone. It needs some other dependant functionalities that the other team members work upon.

# Continuous Delivery, Continuous Integration and Continuous Deployment

Continuous Delivery means you build software in such a way that it is ready to be released to production at any time.

Continuous Integrations means to integrate, build and test code in Dev or Test or both environments. This enables Continuous Delivery to be ready to be released into production at any time. CI means members of a Dev or automation test team integrate their work frequently at the most daily. Each integration is verified by an automation build and test to detect errors as quickly as possible.

Continuous Deployment means that that every change goes through the pipeline and automatically into production.

## Continuous Integration explained

A practice in SE, of merging all developer working copies to a share mainline several times a day.

* A feature to be introduced
* Developer gets a local copy of the repository
* Makes the changes
* Builds and runs in the local system
* When all good. Either commit it. Now if we have more than one developer have made different changes to the repository, our developer first updates his working copy with their changes and then rebuilds and tests. If no conflict or error found, then commit and update the repository. However our developer commit is not completed. When commited to the repo, we build again on an integration machine. When this build succeeds we can say that the changes are done. This integration build can be executed manually or automatically.

# What is a Build System

A Build System is a system used to automatically

* produce all the artefacts required to successfully
  + deploy,
  + test, and
  + run our production software.

In simple term, A build system is a simple functional program that takes code as input and produces deployable software as output. The most important function of a build system is to compile source code. However, we often want to do more than that as part of the build process.

For eg, in a production build system, we might want to

* update version numbers
* run unit tests
* perform style of coverage analysis
* coalesce the output of several projects into a single artefact
* publish the artifacts to a publicly accessible location.

In order to do this, we create a Build System that can be shared across all the projects. We have a few basic principles that will help create a high quality build system

1. The build system should have a very few environmental dependencies.
2. The build system should not be dependent on a continuous integration server.
3. The build system should do all the right things out of the box
4. The build system should be extensible and flexible.

How is build set up

Build definition - defines a sequence of steps to execute in order to get the required artefacts used to deploy, test and run our code in production. This is a template that we will use to run our build over and over again.

Build tasks– takes some input and produces an output

Build agent – takes the information and executes the tasks in the sequence defined

Build Infrastructure

#### Agents and pipelines

Hosted build agent- provided by MS which is paid per min of usage. Every account has free build mins. Can buy additional mins- 2 options.

1st option

Buy additional build mins

Buy hosted pipelines. Hosted pipelines have unlimited mins for one concurrent build. Reason called pipelines because the agents are also used for release management. Both build and release are together continuous delivery.

2nd option – custom build agents – can be used for release pipeline.

Download the agent software and register with Azure devops

Every custom agent comes with custome pipeline. Default has a free pipeline . if needed more, need to buy.

Build Security

Agent is a process that runs the build on the machine