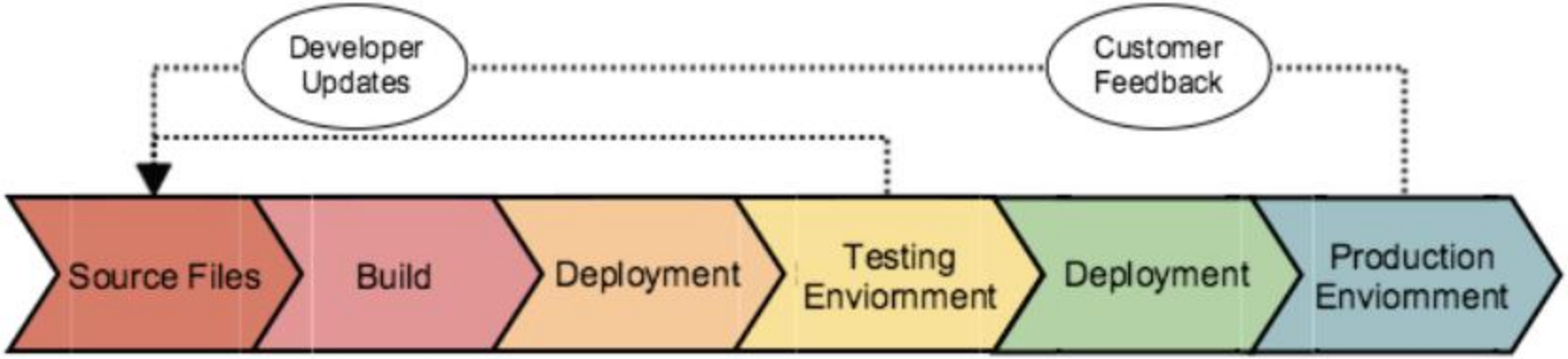
CODE PIPELINE

What is CodePipeline?

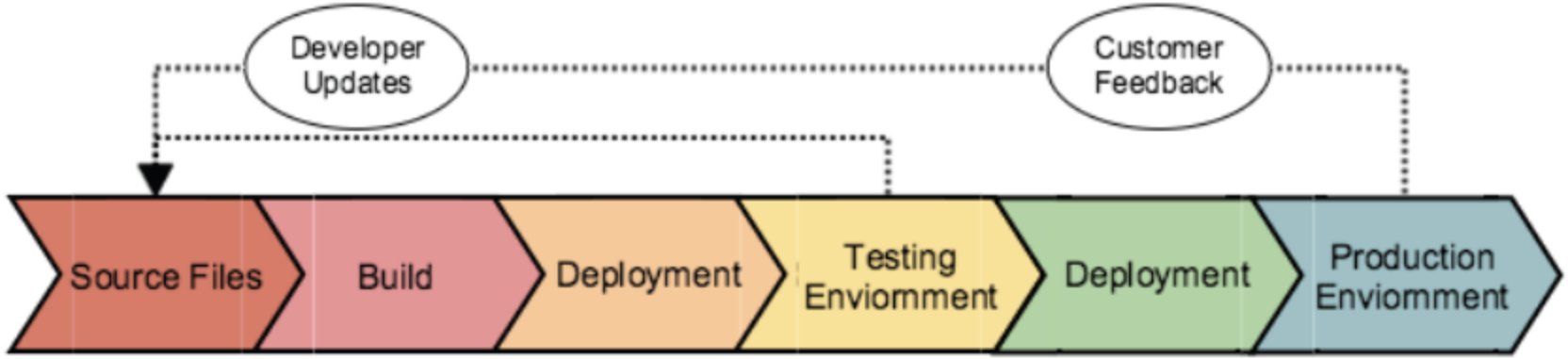
- CodePipeline is a continuous delivery service. It provides the tools to model, visualize, and automate the many steps that are required as part of the software release process.

- ***Automation:***
  - You can easily automate the entire release process, from the source repository all the way to production servers.
- ***Consistency:***
  - Create and repeat a consistent set of steps each time you want to release or update your software.
- ***Speed up the delivery process:***
  - Speed up the release process through automation.
- ***Use existing tools:***
  - CodePipeline integrates with many other AWS services (CodeCommit, CodeDeploy, S3, OpsWorks), as well as other major developer and DevOps platforms such as Jenkins and GitHub.
- ***Easy to visualize and view:***
  - See each stage of the release process, their status, and position.

Common Software Release Process:



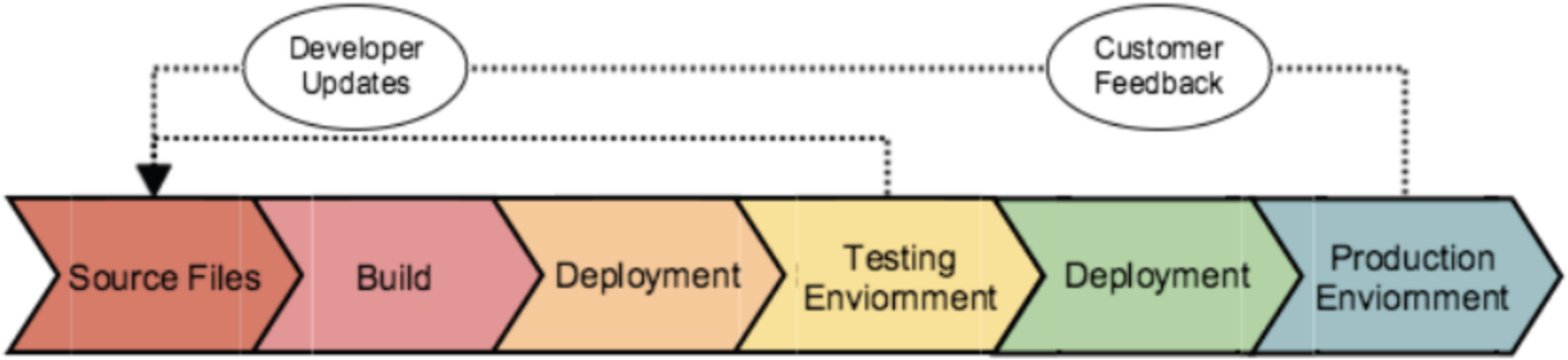
Common Software Release Process:



CodeCommit

S3 Github

Common Software Release Process:



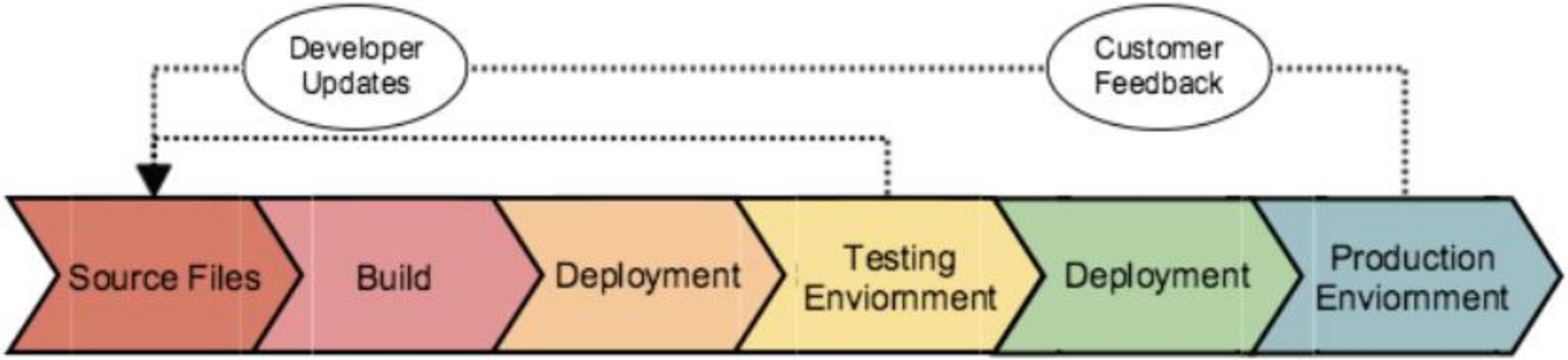
CodeCommit

S3

Github

Jenkins

Common Software Release Process:



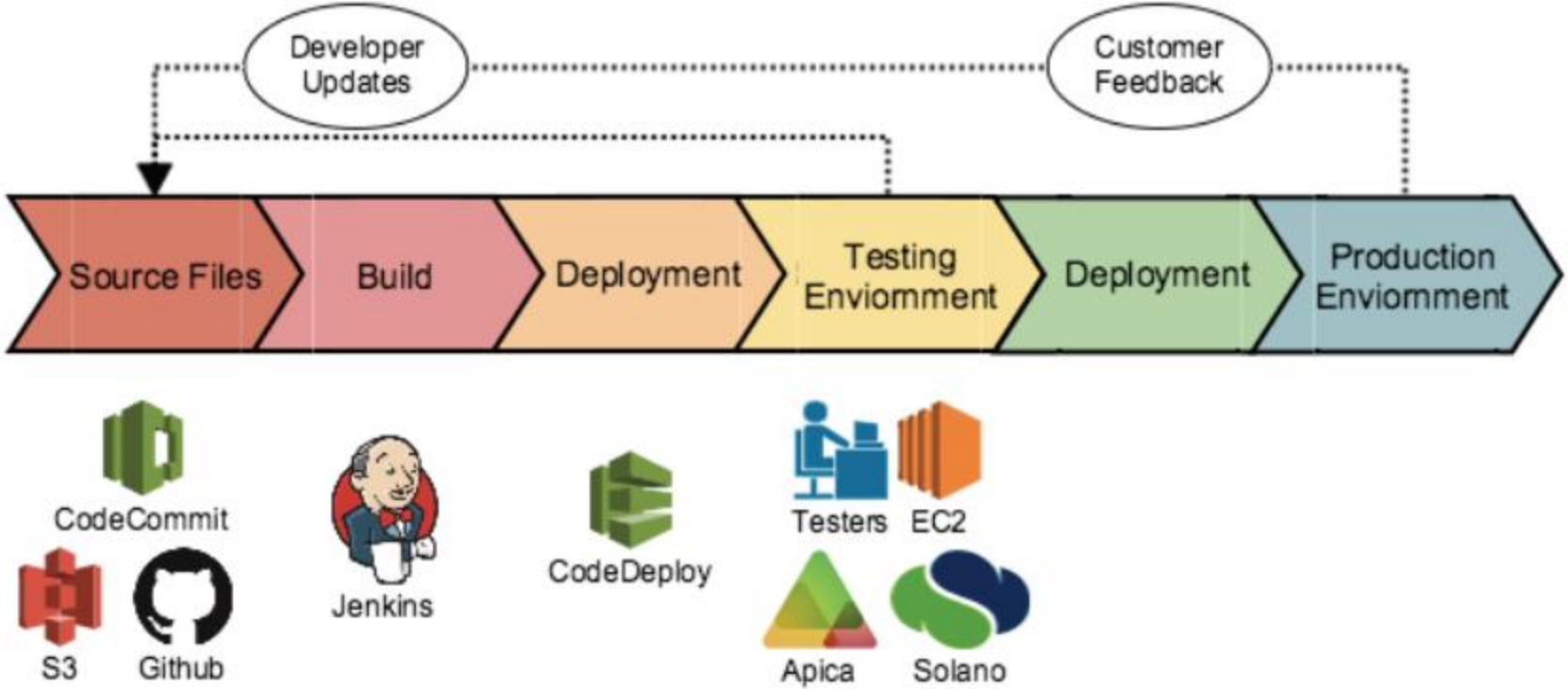
CodeCommit

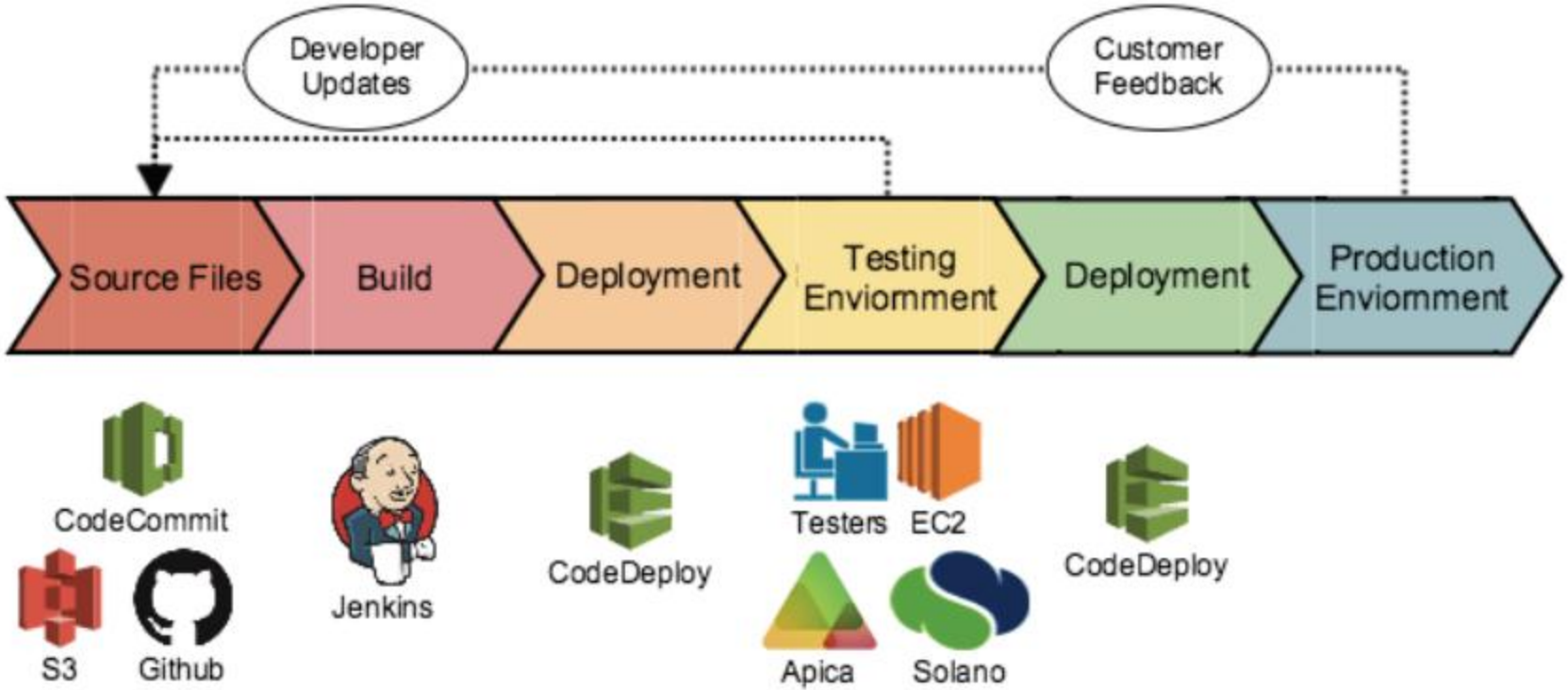
S3

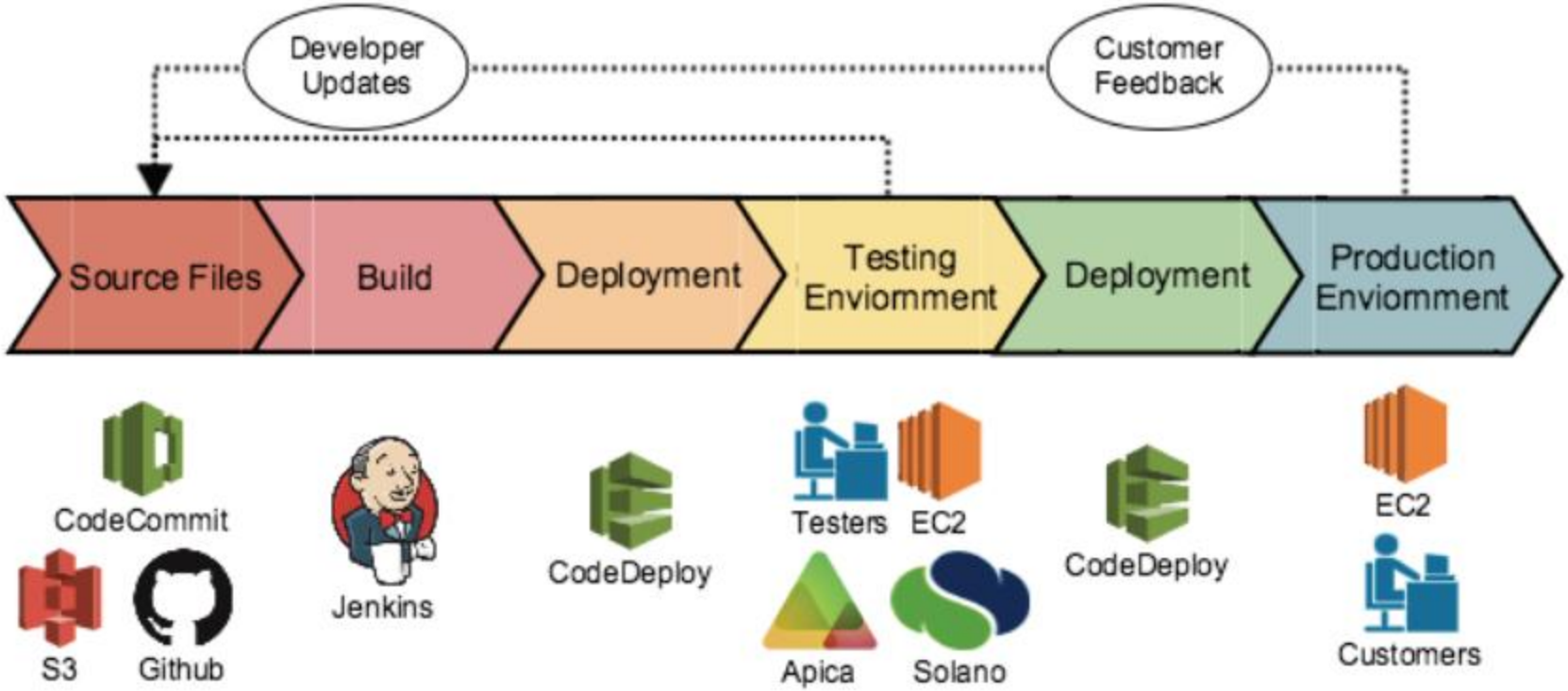
Github

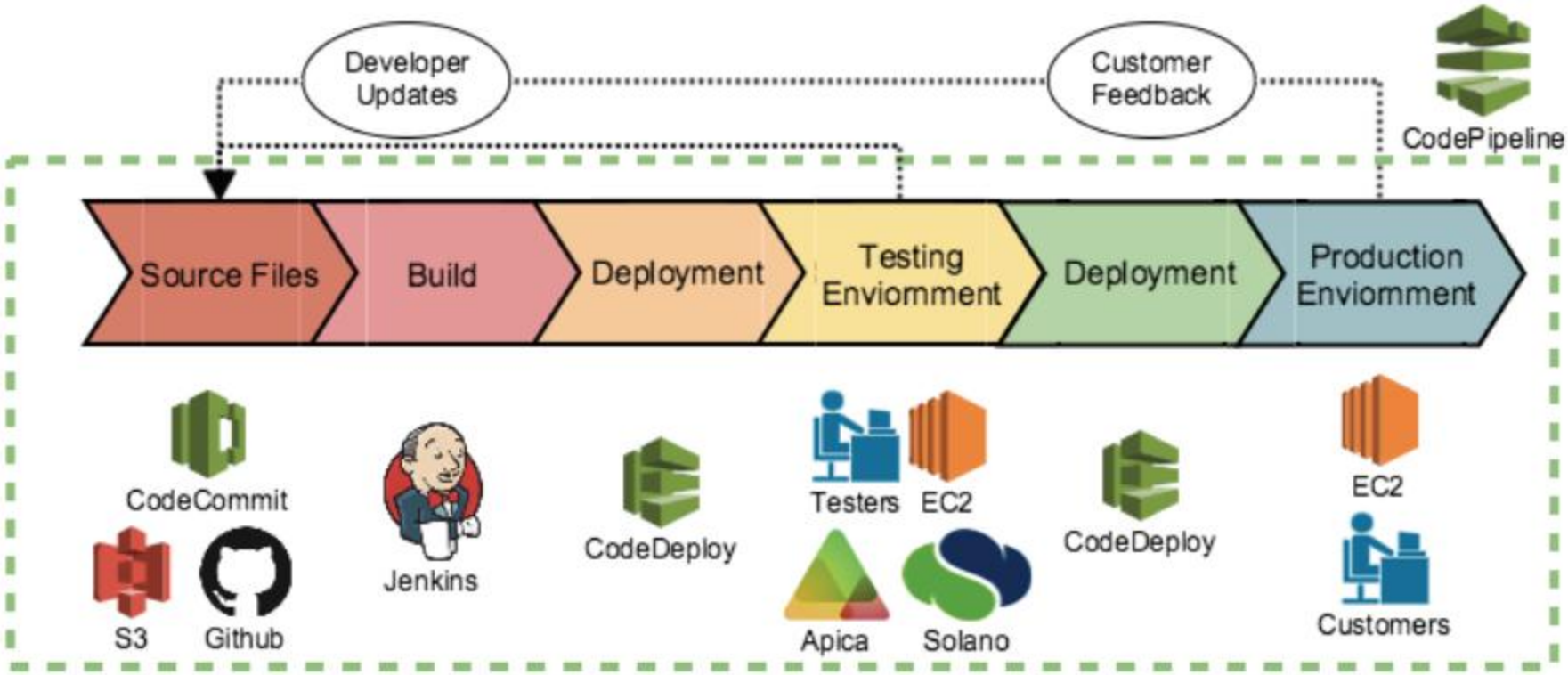
Jenkins

CodeDeploy









Continuous Delivery:

- A software engineering approach where teams produce software in short cycles, ensuring that the software can be reliably released at any time. It aims for building, testing, and releasing software faster and more frequently. The approach helps reduce the cost, time, and risk of delivering changes by allowing for more incremental updates to applications in production. ***A straightforward and repeatable deployment process is important for continuous delivery.***

Continuous Integration:

- The practice of merging all developer working copies to a shared mainline (such as a master branch), at a continuous pace (often several times a day). Each addition (change) is built and tested as quickly as possible.

AWS CodePipeline Concepts & Terminology:

Pipeline:

- A workflow framework that helps you create and manage the release process. (blueprint)
- It is how you specify, build, coordinate, monitor, and execute your specific release process.
- Consists of:
  - **Stages** (which consist of **Actions**)
  - **Transitions** (between each Stage)
- The first time you create a pipeline (AWS Console) an S3 bucket is automatically created that will store the pipeline's **artifacts**:
  - Created in the same region as your pipeline
  - The bucket is named "codepipeline-<REGION>-<RANDOM_10_DIGIT_#>"
- You can have up to 20 pipelines per AWS account
- Pipelines must be in one of the following regions:
  - us-east-1
  - us-west-2
  - eu-west-1

AWS CodePipeline Concepts & Terminology:

Stages:

- Each pipeline is broken up into broad sections called stages.
- Stages are used to categorize, execute, and monitor **actions**.
- Stages are completed in order – as configured in the pipeline.
- Every stage must have at least one action.
- AWS default stages include:
  - Source
  - Build
  - Beta
- A stage can only process one **revision** at a time.
- Stages are connection through **transitions**.

AWS CodePipeline Concepts & Terminology:

Actions:

- An action is a task performed on the **artifact**.
- Action tasks include:
  - **Source**: Monitors the "source" location (i.e. CodeCommit, S3, GitHub) for new commits or uploaded revision – and starts the release process if found
  - **Build**: Code is built (i.e. Jenkins)
  - **Test:** Run the code through a test provider (i.e. BlazeMeter, Apica, etc.)
  - **Deploy**: Install the application files onto a fleet of instances
  - **Invoke**: Trigger a Lambda function
  - **Approve**: Require human approval before moving to the next stage

Note: Services, such as CodeCommit, Github, Jenkins, Apica, CodeDeploy, etc – are referred to as action "providers"

AWS CodePipeline Concepts & Terminology:

Revisions, Artifacts & Transitions:

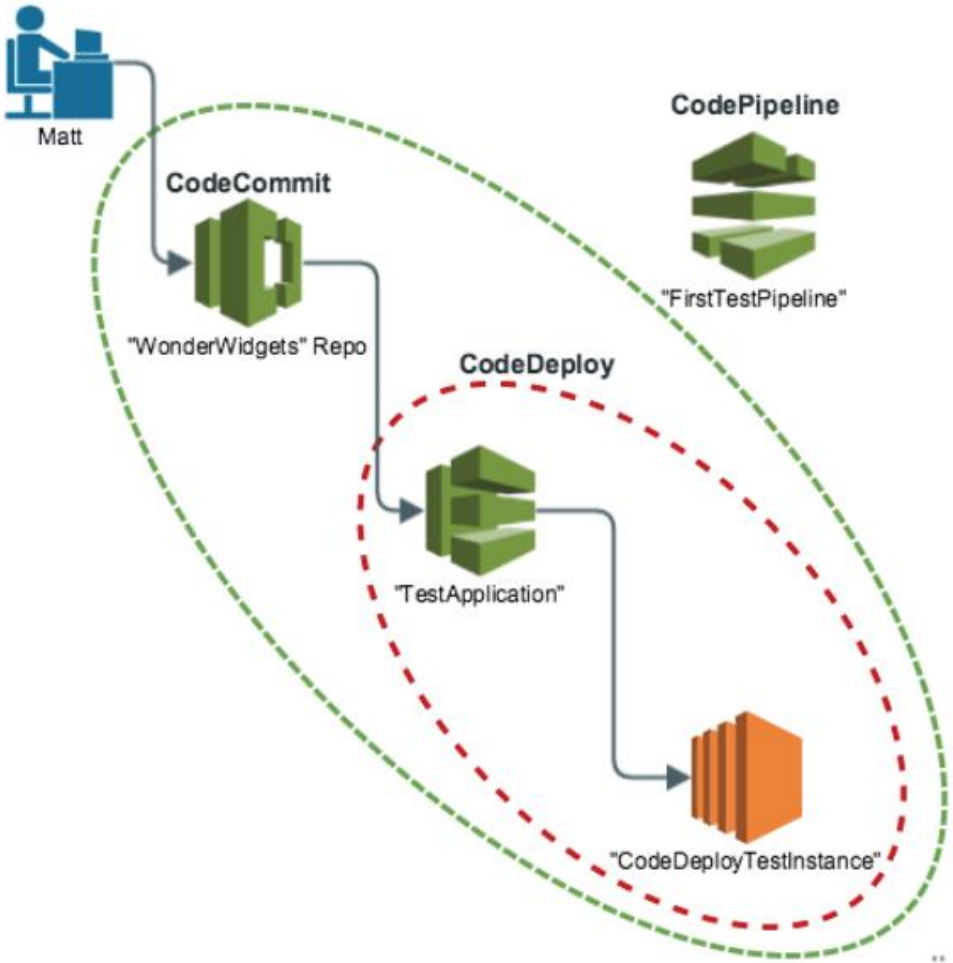
- **Revision** is the general term used to describe the code "update" or version that is currently running through the pipeline.
- **Artifacts** refer to the actual set of files (objects) that are being passed through the pipeline, and is categorized by ***Input*** or ***output*** artifacts.
- **For example:** The un-built source files being passed into the "build" stage are the "input artifacts." The built files (after running through Jenkins), are the "output artifacts."
- The "output artifacts" of a stage are then passed to the next stage via transitions.
- Artifacts are stored in the S3 bucket that was created or designated when you create a pipeline.
- **Transitions** tell the artifact what stage to go to next, and act as a delivery system between them.
- Transitions can be enabled or disabled to allow or prevent upcoming stages to be run.

CodePipeline Setup & Configuration:

1) IAM Policy:
   - *Attach the **"AWSCodePipelineFullAccess"** IAM policy to any users or roles that you want to grant CodePipeline access.*

2) Install and configure the AWS CLI:
   - *If you have followed this course from the beginning (CodeCommit section) then you should already have the CLI setup and configured.*

   - *If you skipped ahead to this section (CodePipeline), you will need to set up and configure the AWS CLI. Please refer to the following lesson (located in the CodeCommit section of this course) depending on your operating system:*
     - *For Linux/OSX users:*
       (7) OSX/Linux: AWS CLI Installation
     - *For Windows users:*
       (2) Windows: GIT & AWS CLI Installation



Creating a Pipeline (AWS Console):

1) *Navigate to CodePipeline and click on "Get Started" or "Create a Pipeline".*

2) *Give the pipeline a name.*

3) *Select a Source (CodeCommit).*

4) *Select a Build Provider (optional).*

5) *Select a Deployment Provider (CodeDeploy).*

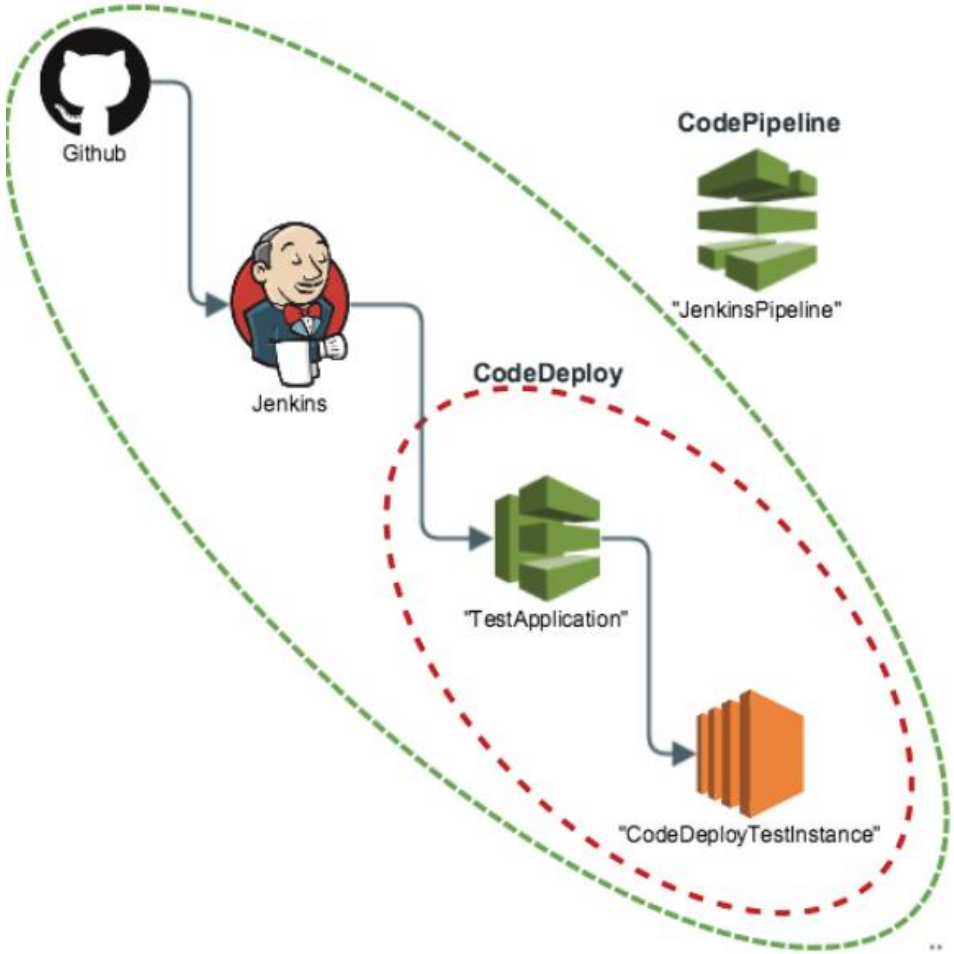
6) *Create/Select an AWS Service Role (permissions).*

7) *Review the Pipeline & create it.*

8) *View the results.*

Items We can Manage in the AWS Console:

1) View pipelines and detailed pipeline information
2) Edit pipelines
3) Disable or enable transitions between stages
4) Retry failed actions
5) Delete pipelines



http://docs.aws.amazon.com/codebuild/latest/userguide/sample-codedeploy.html