MATLAB EXPO 2021

MATLAB과 Simulink를 이용한 Continuous Integration 활용 방안

김 학범

MathWorks

Today's Agenda

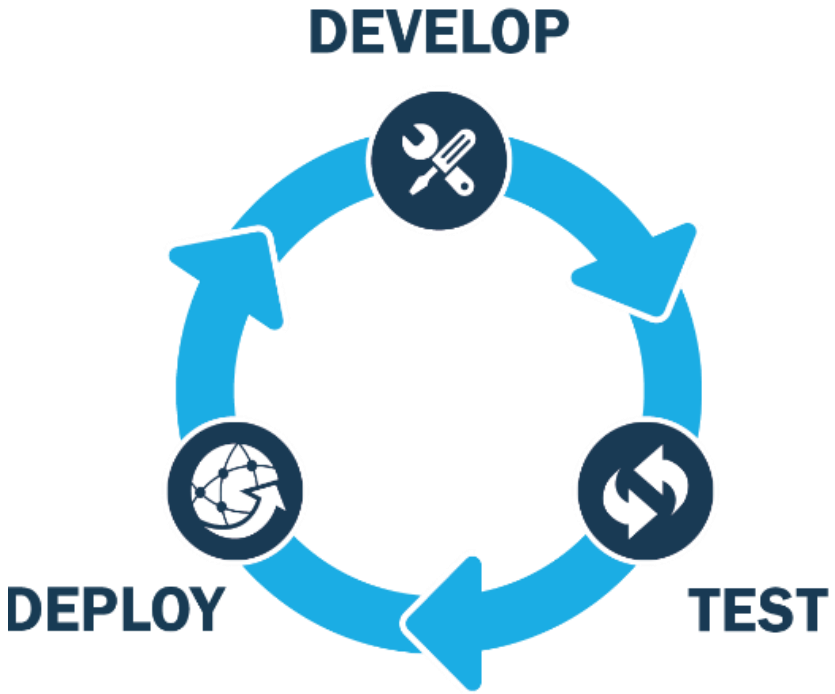
- What is Continuous Integration (CI)?
- Continuous Integration Workflow with Model-Based Design
- Easiest way to get started with CI
- Free CI Support for Public GitHub Repositories
- Summary and Resources

Continuous Integration (CI)

Automatically build, test, package, and deploy code

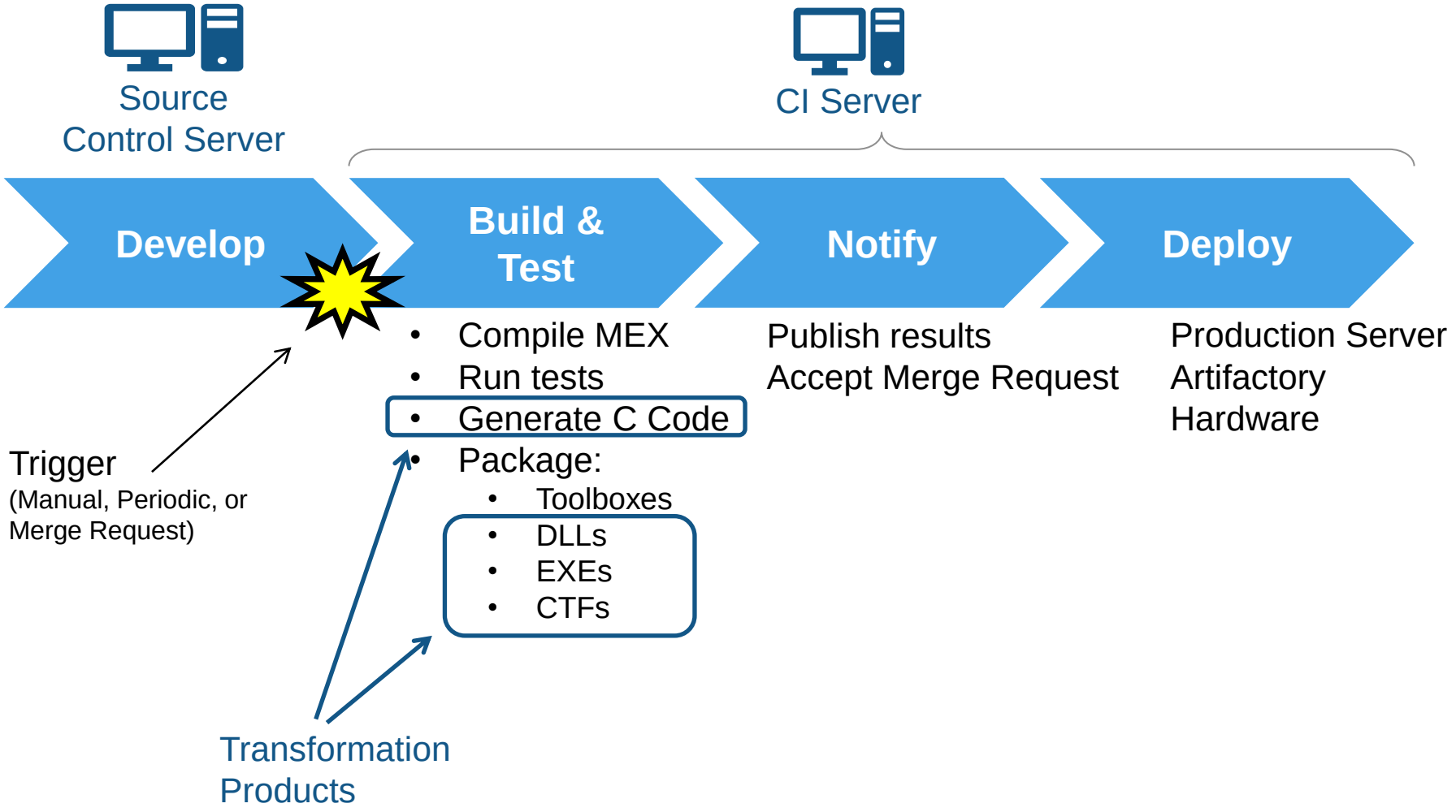
Why automate?

- Better code quality
- Faster development
- Easier collaboration
- Traceability



CI Server automates the process

Many pipeline variations!!



Benefits of Continuous Integration

Consistent, repeatable environment

Automate testing of latest changes

Test early, test often

Identify integration issues quickly

Trace issues to the source

Who uses CI?

Traditionally, only software companies used CI…

Today, engineers and scientists are riding the CI wave, too!

What do engineers and scientists do with CI?

Tessella Designs Attitude and Orbit Control Algorithms for Solar Orbiter Spacecraft Using Model-Based Design

"We saw the benefits of Model-Based Design on several previous projects. On this project, MATLAB and Simulink enabled us to create a detailed specification that minimized deviation between the prototype algorithms we developed, tuned, and tested in Simulink and the final software implementation."

— *Andrew Pollard, Tessella*

Artist's rendition of the Solar Orbiter.

Accelerating Development of a Diabetes Management System with Model-Based Design: Q&A with Bigfoot Biomedical

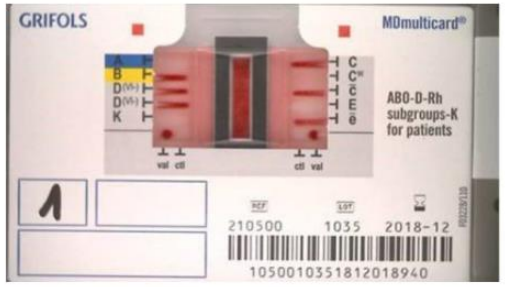
"Having seen the advantages of Model-Based Design in other industries, we know it will enable us to generate knowledge faster than traditional clinical methods and accelerate the process of bringing a medical system to market."

— *Lane Desborough, Bigfoot Biomedical*

IDNEO Develops Embedded Computer Vision and Machine Learning Algorithms for Interpreting Blood Type Results

"With the time we saved by generating code with Embedded Coder, we were able to experiment with new features and complete additional iterations in MATLAB, incorporating customer feedback on the early prototypes."

— *Marc Blanch, IDNEO*



The Grifols MDmulticard.

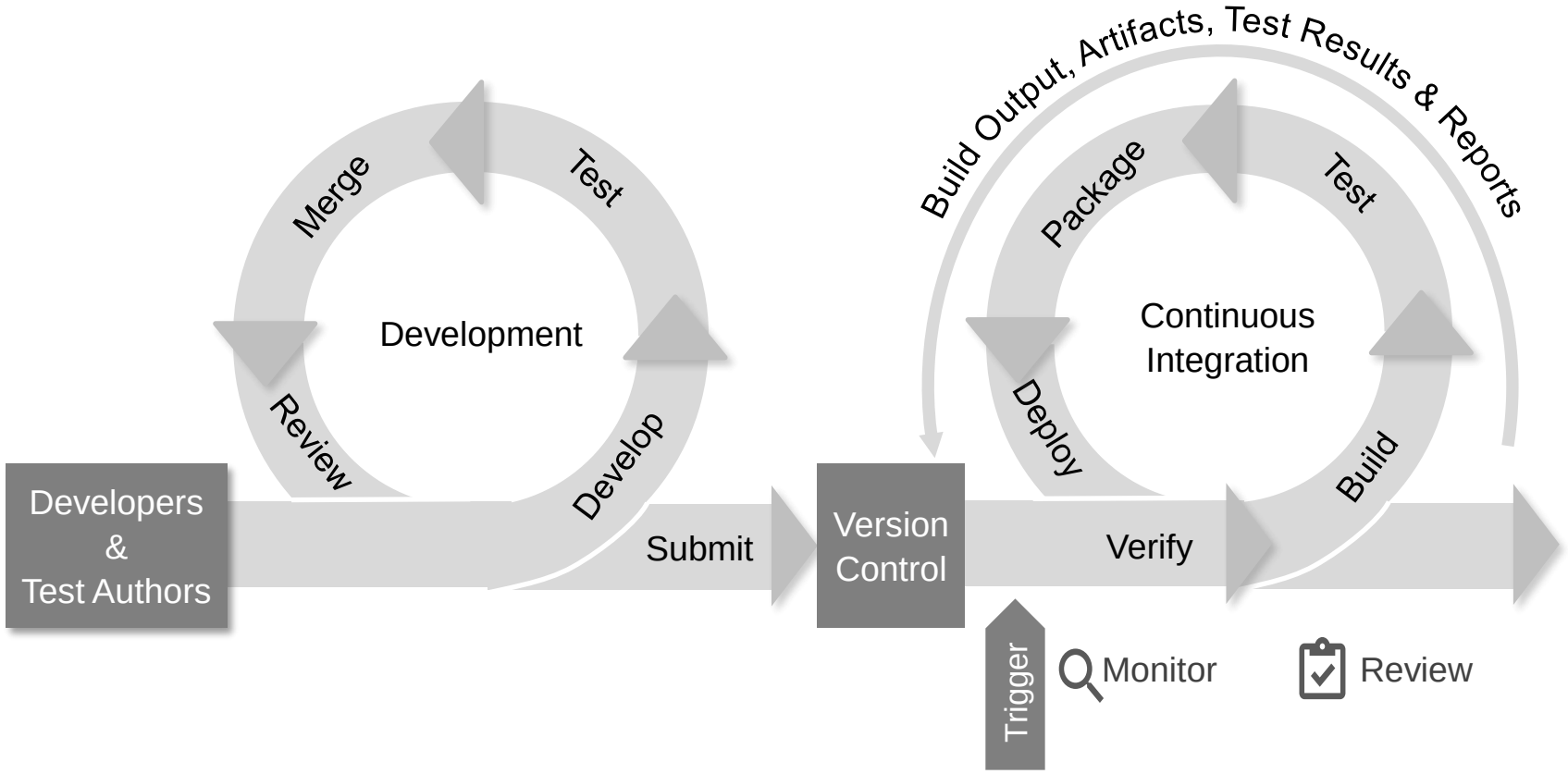
Vestas Develops Control Software for Wind Power Plants with Model-Based Design and Continuous Integration

"With Simulink and Embedded Coder, we can show our customers and grid operators a simulation that incorporates the actual code that will run in our power plant controller. That's what grid operators want, and it gives Vestas an advantage over competitors who still use conventional approaches."

— *Per Hagen Nielsen, Vestas*

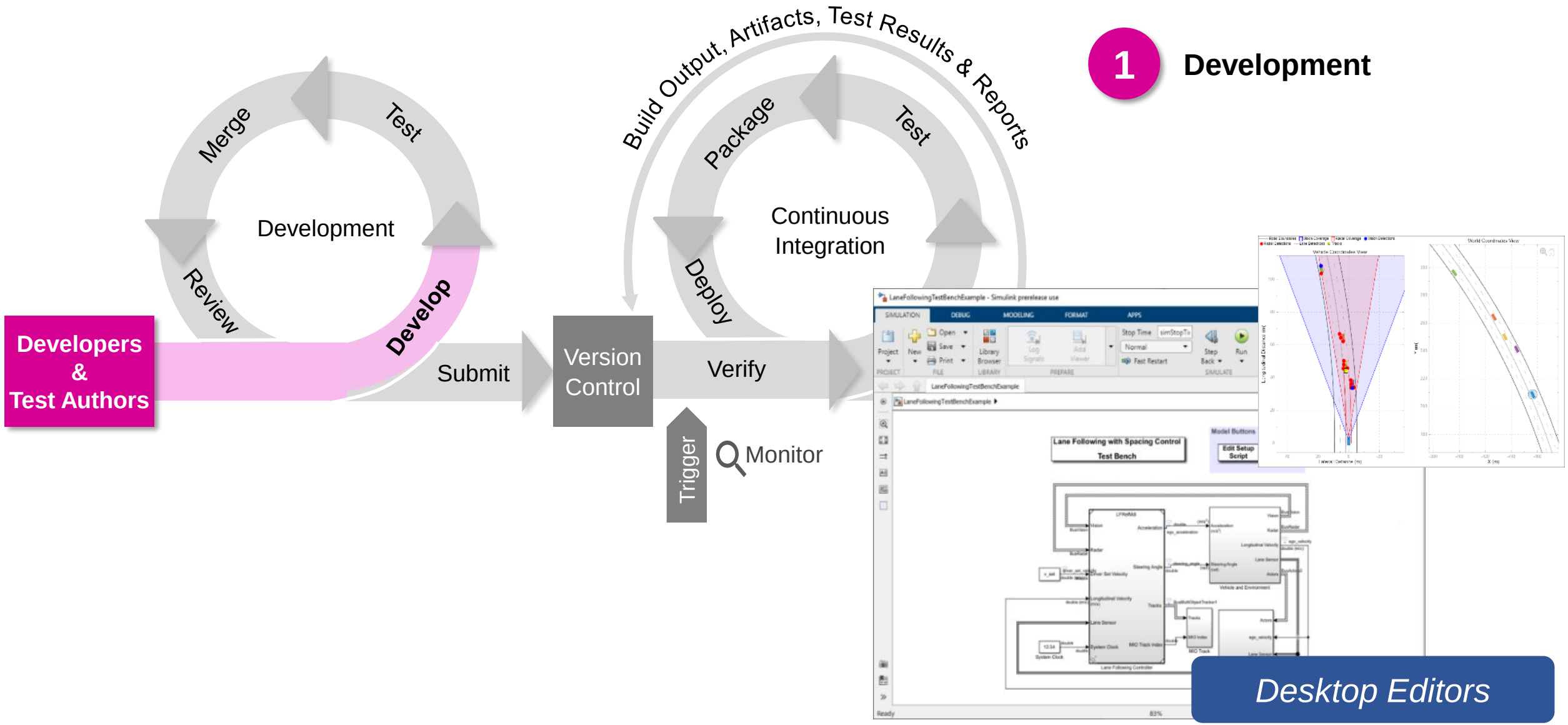
Vestas turbines and power plant control.

Continuous Integration Workflow and Tools are Language and Domain-Neutral

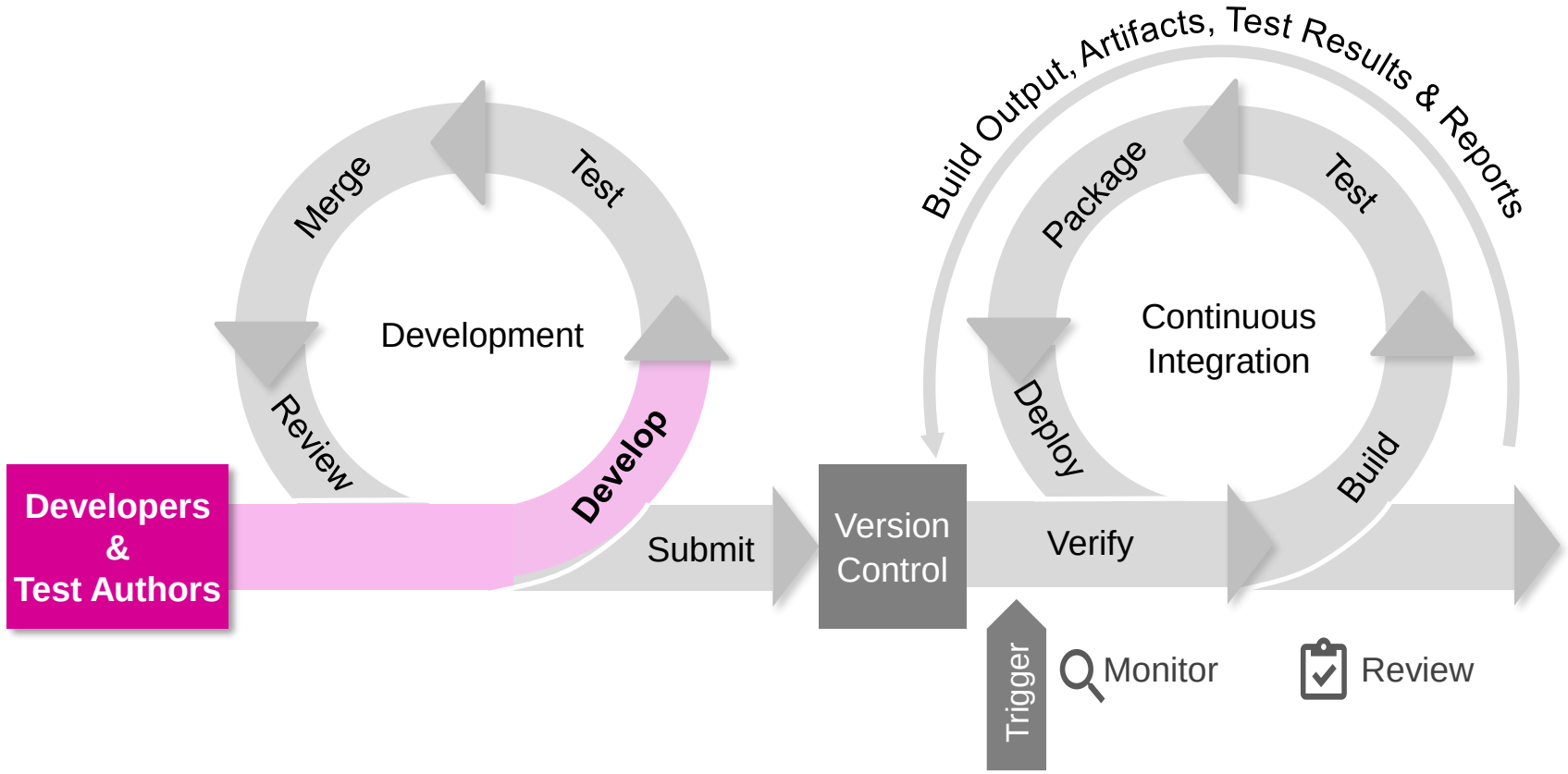


Each of these must “speak” MATLAB and Model-Based Design

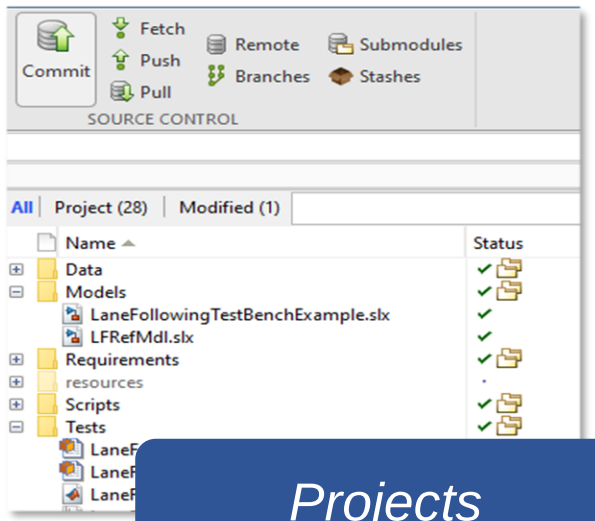
Continuous Integration Workflow with Model-Based Design



Continuous Integration Workflow with Model-Based Design

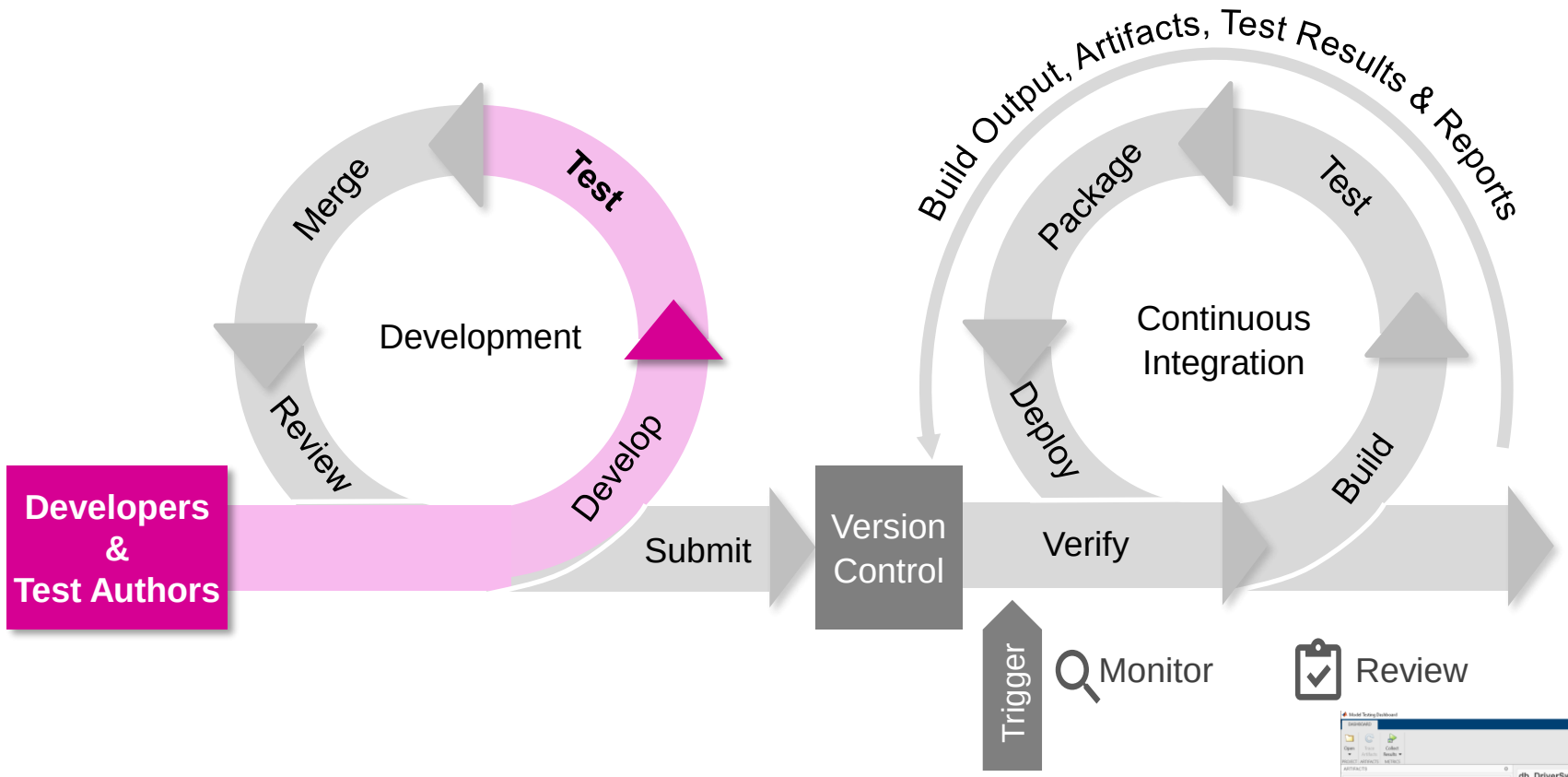


1 **Development**

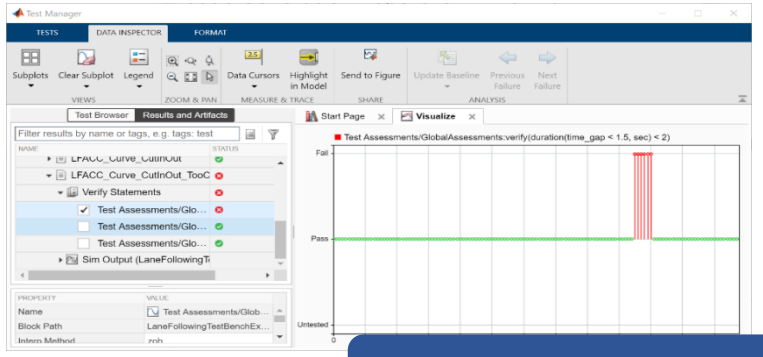


Projects

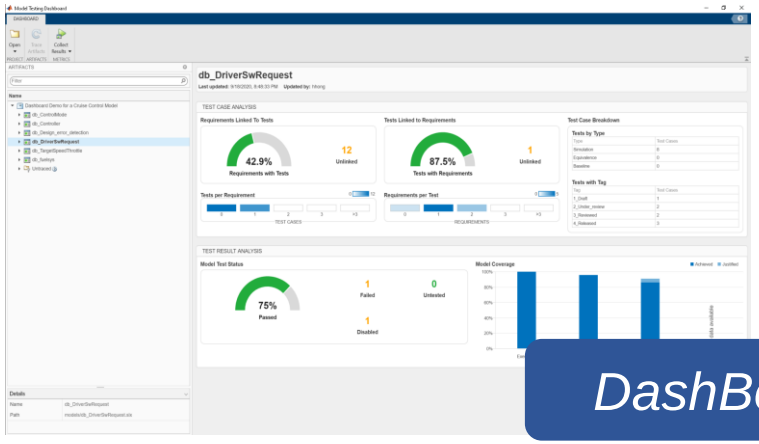
Continuous Integration Workflow with Model-Based Design



1 **Development**

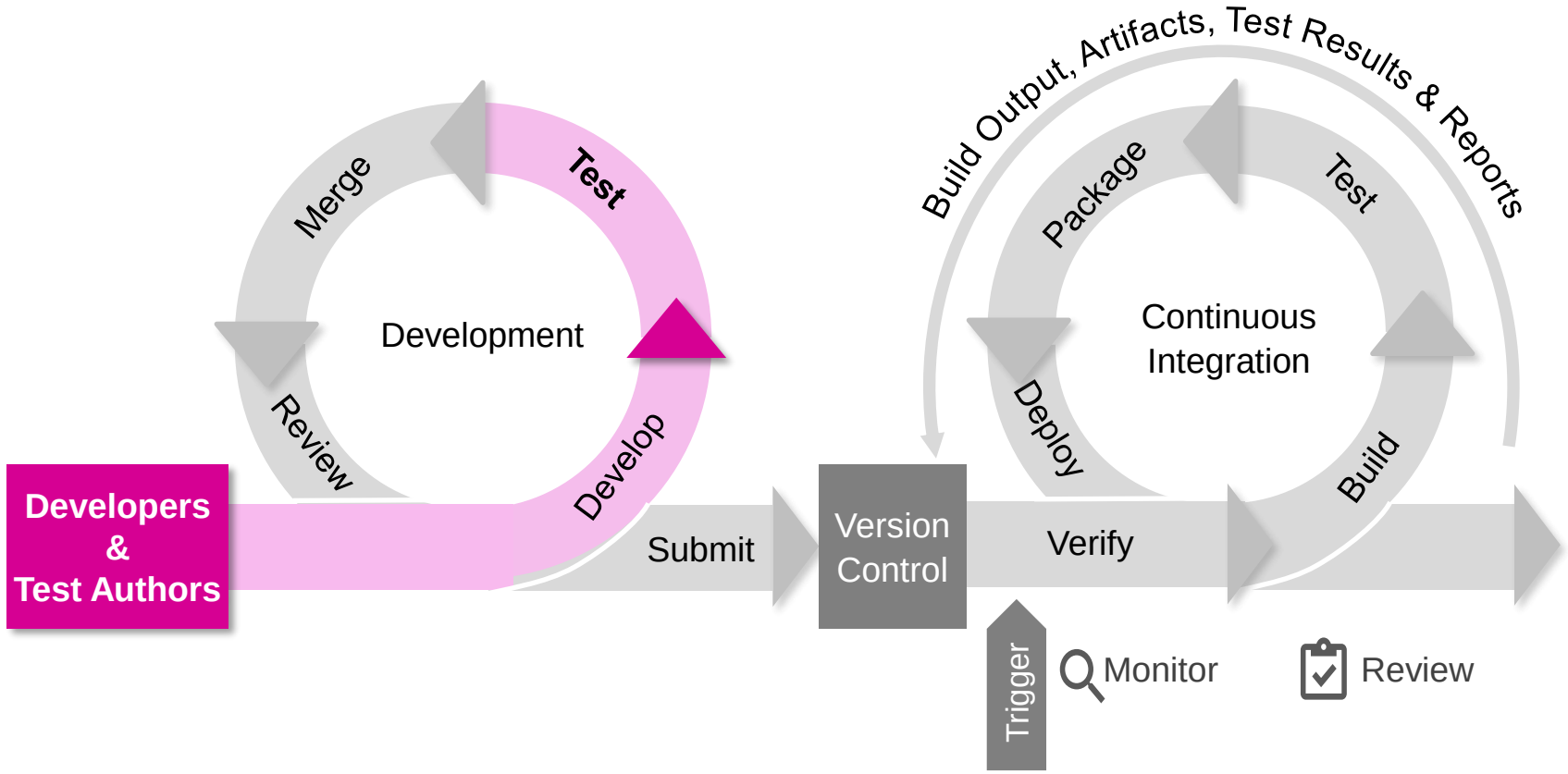


Simulink Test



DashBoard

Continuous Integration Workflow with Model-Based Design

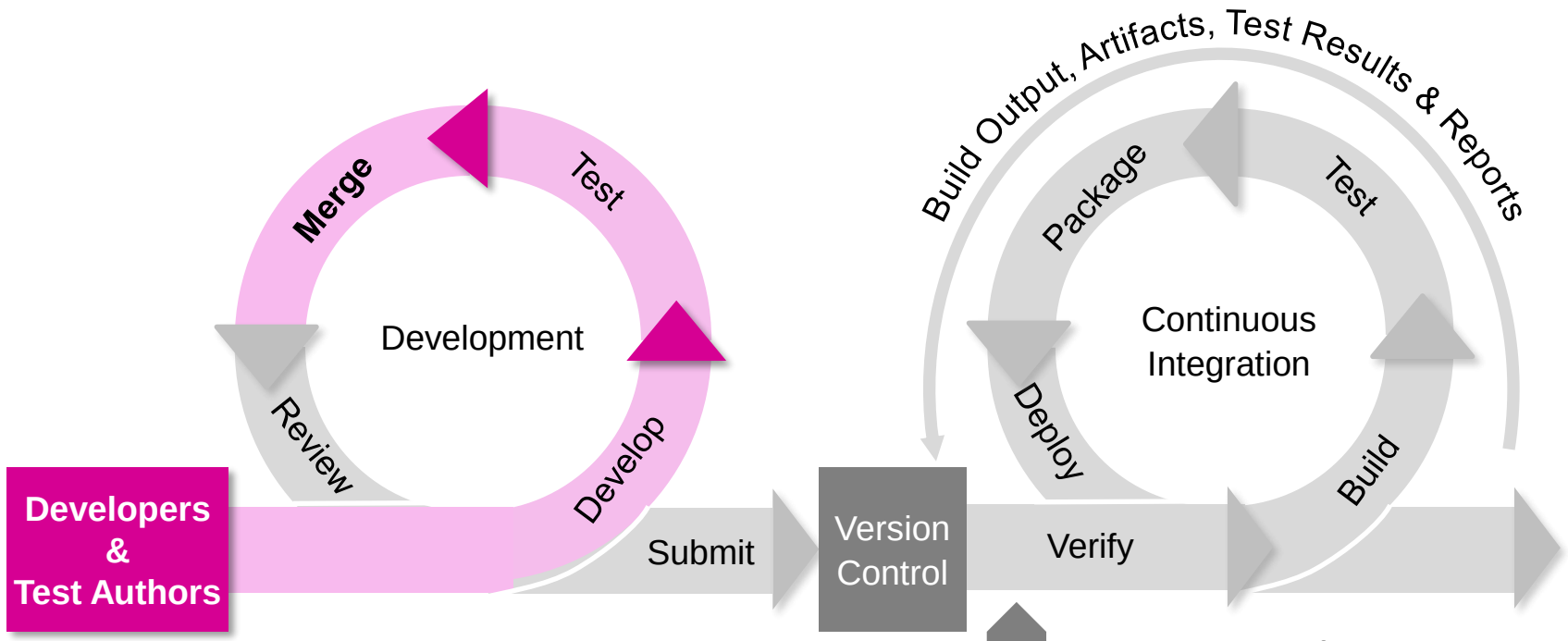


1 **Development**

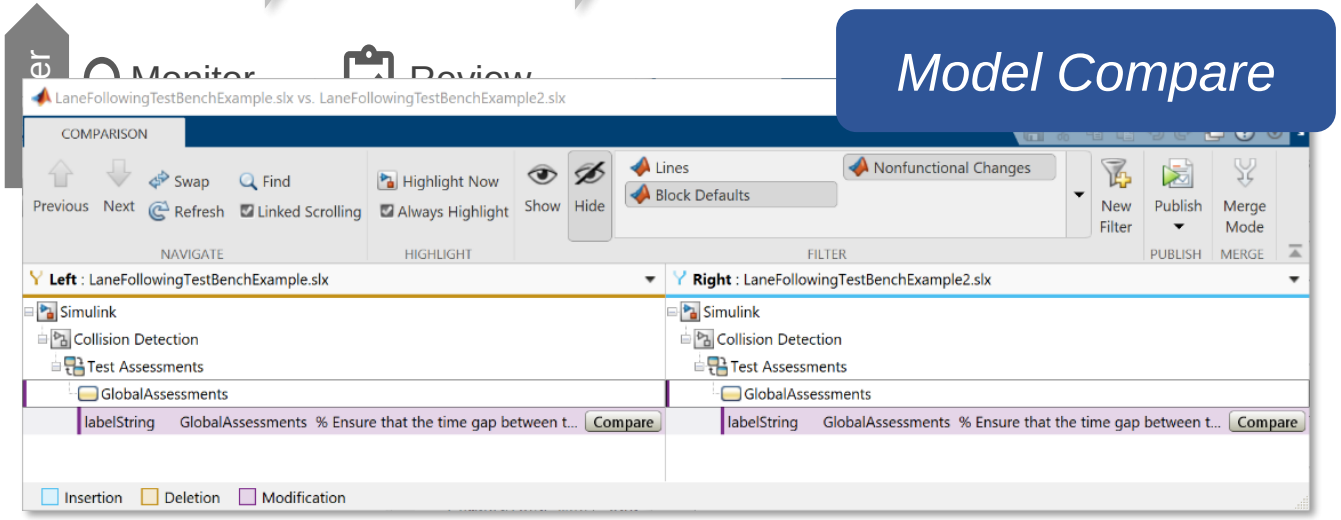
```
function tests = exampleTest
tests = functiontests(localfunctions);
end

function testEquivalence(testCase)
mdl = 'topHarness';
blk = 'topHarness/mySineWave';

% Load the model, and close it once finished
addTeardown(testCase, @()close_system(mdl, 0));
load_system(mdl);

% Perform normal simulation
set_param(blk,'SimulationMode','Normal')
simout_normal = sim(mdl);

% Perform SIL simulation
set_param(blk,'SimulationMode','Software-in-the-loop (SIL)');
simout_sil = sim(mdl);

verifyEqual(testCase,...
            simout_normal.yout{1}.Values.Data,...
            simout_sil.yout{1}.Values.Data);
end
```

MATLAB Unit

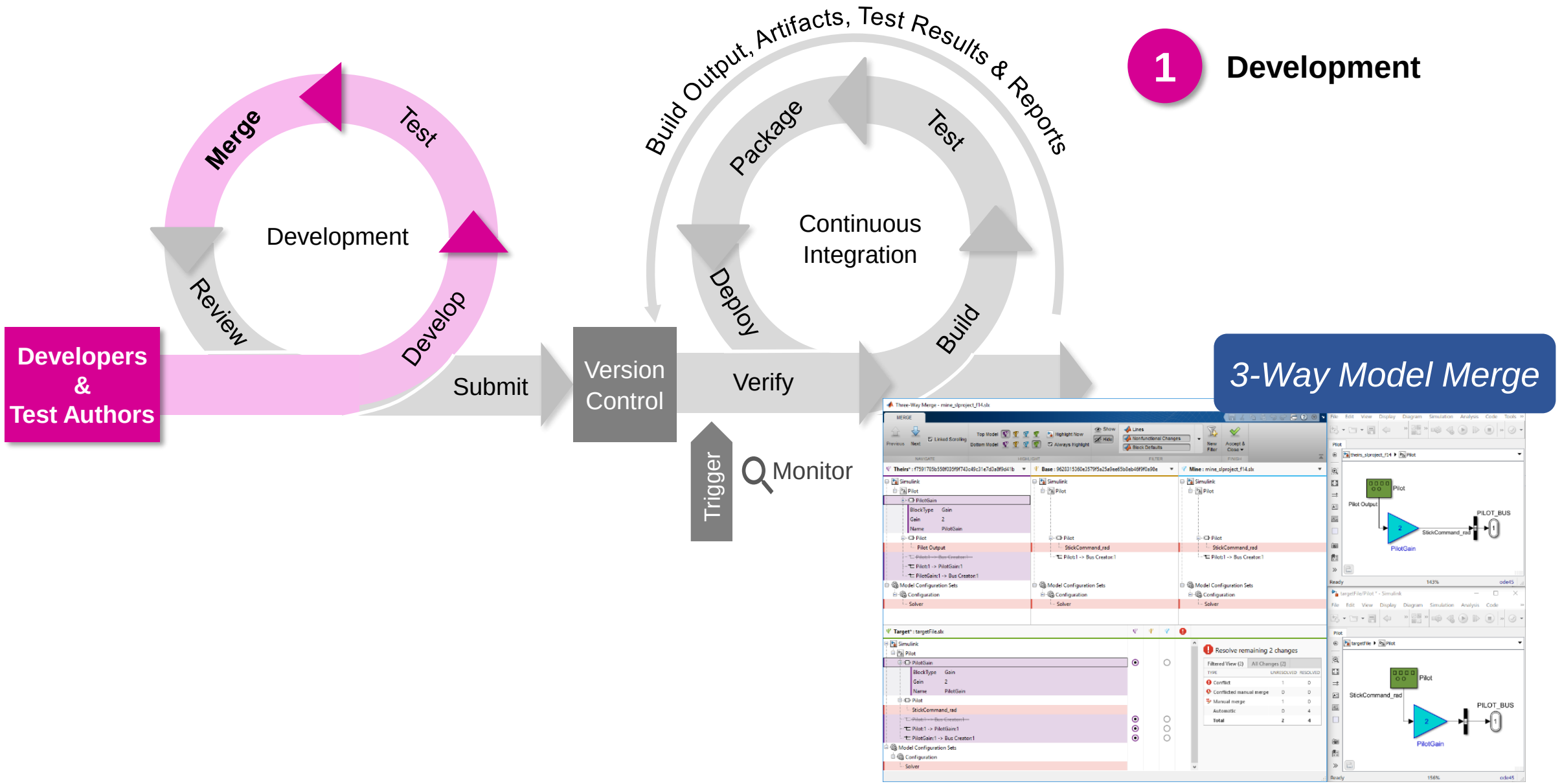
Continuous Integration Workflow with Model-Based Design



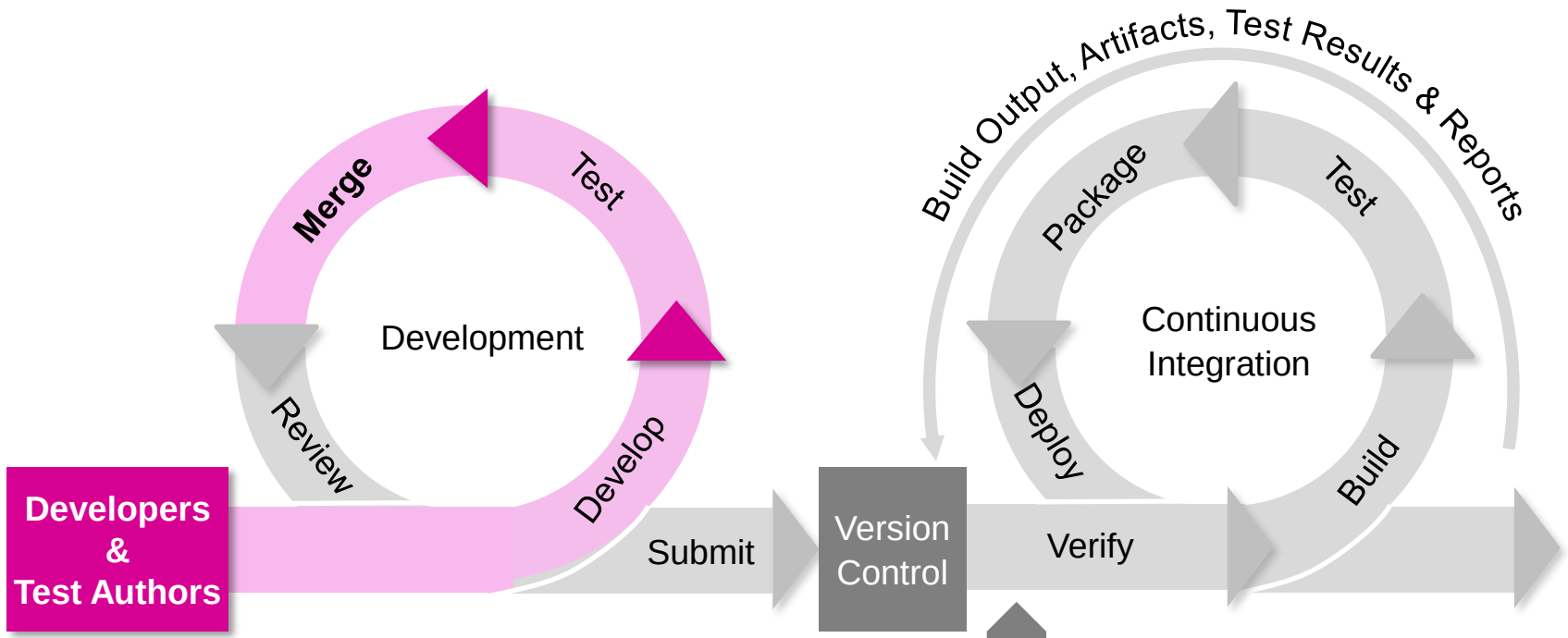
1 **Development**



Continuous Integration Workflow with Model-Based Design

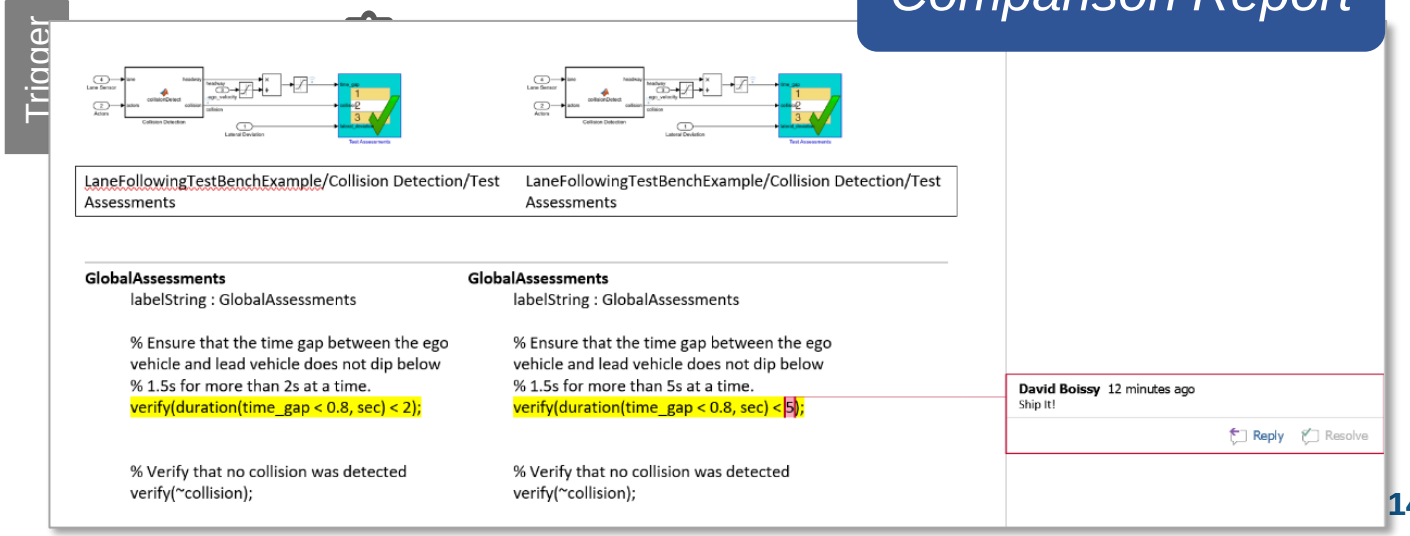


Continuous Integration Workflow with Model-Based Design



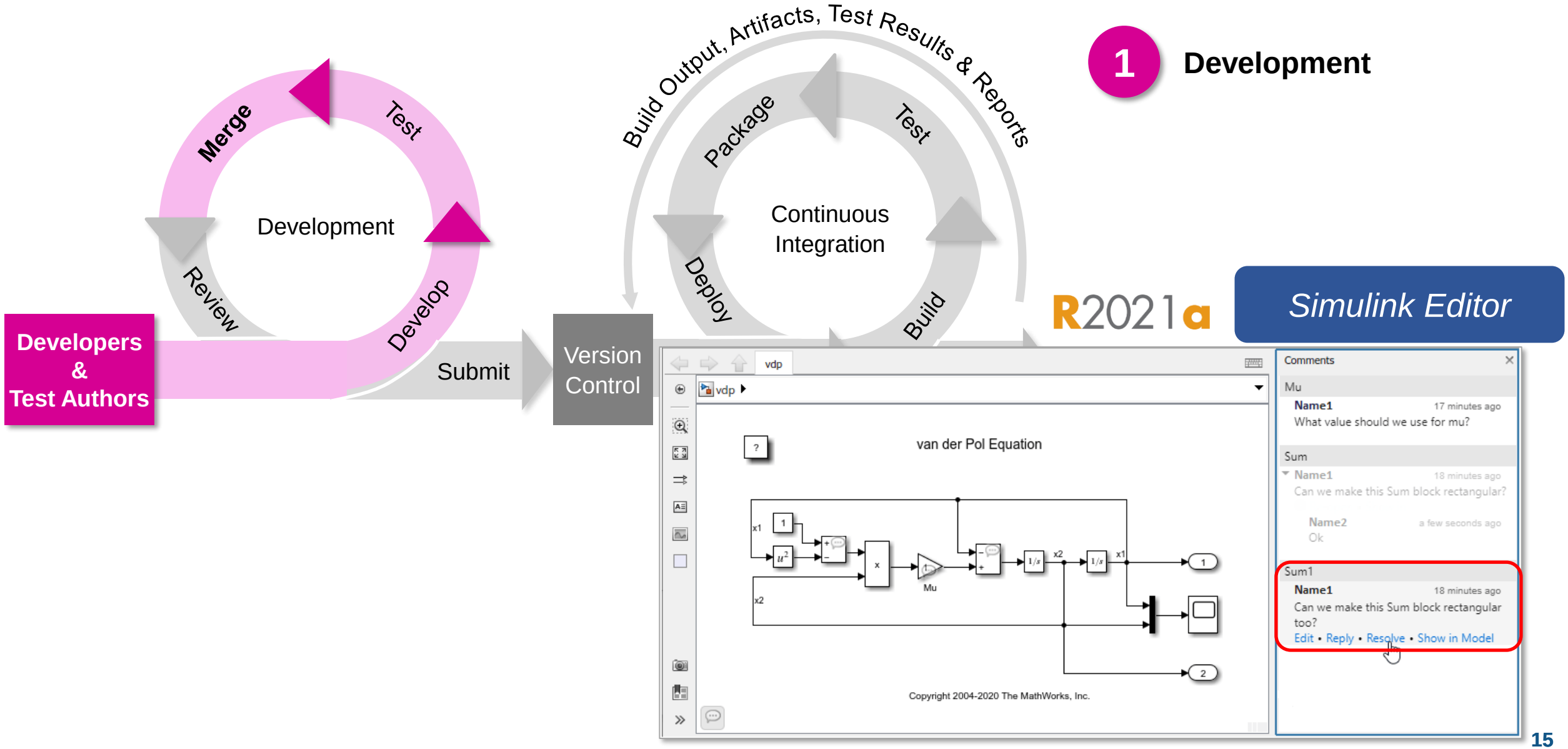
1 **Development**

Trigger

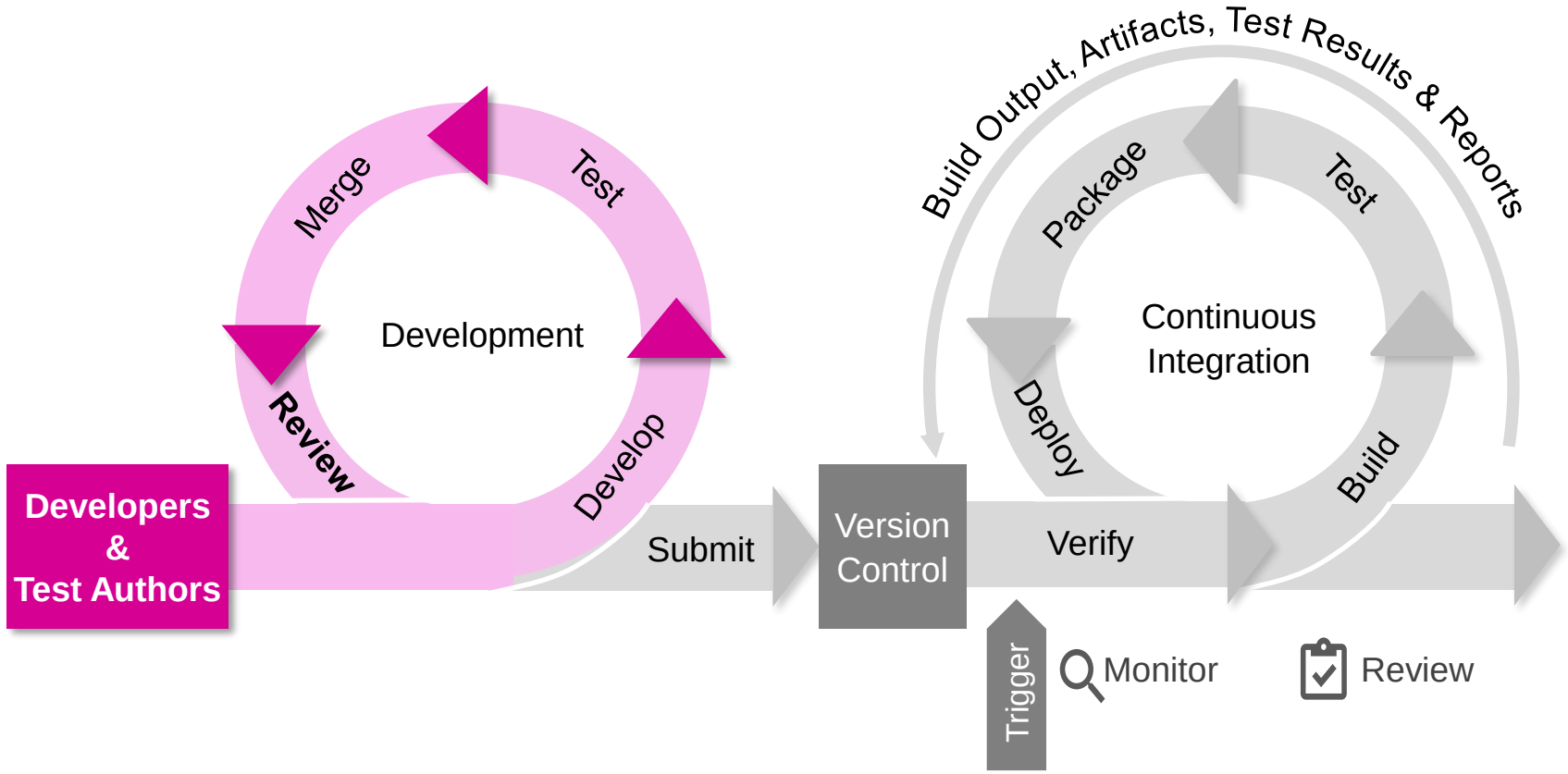


Comparison Report

Continuous Integration Workflow with Model-Based Design

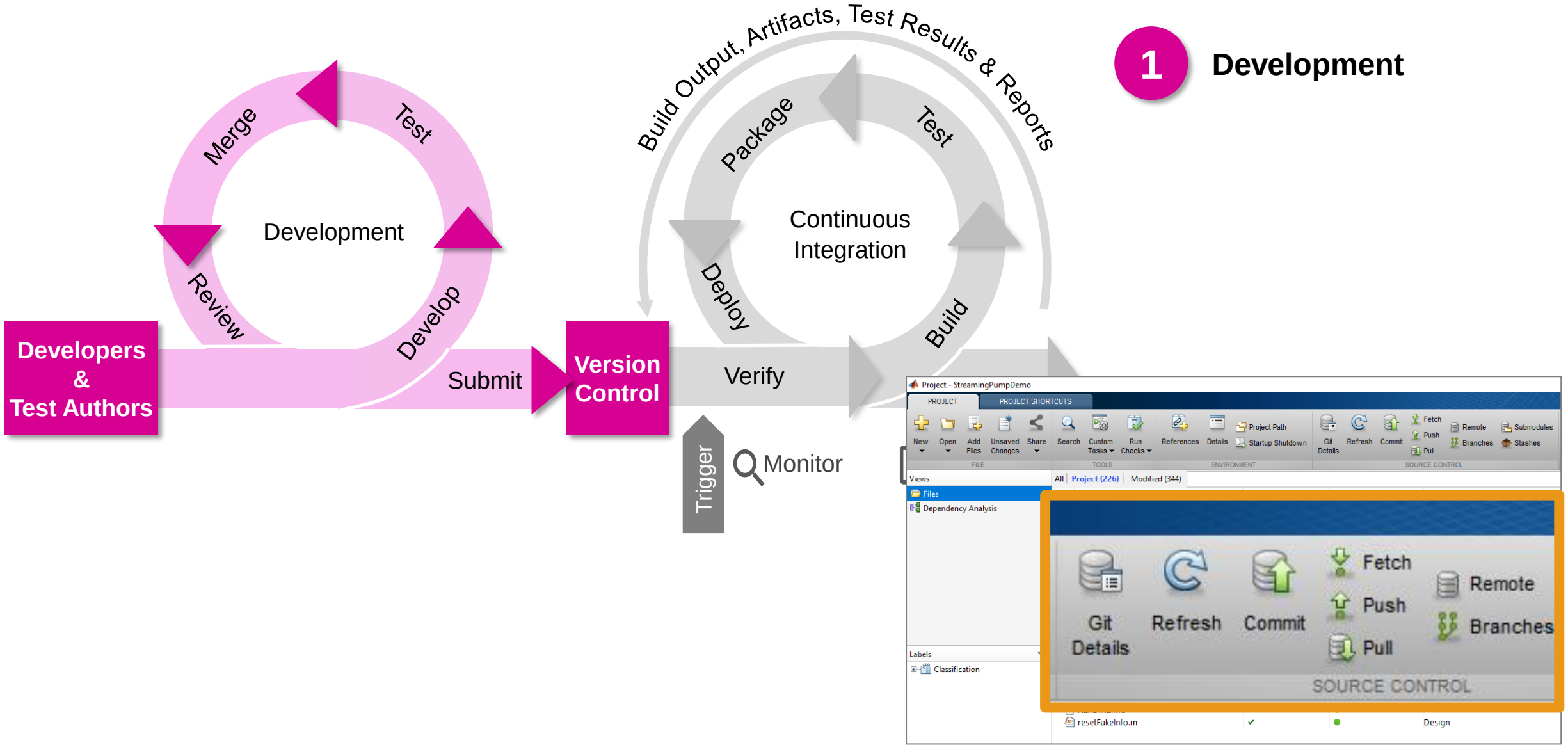


Continuous Integration Workflow with Model-Based Design

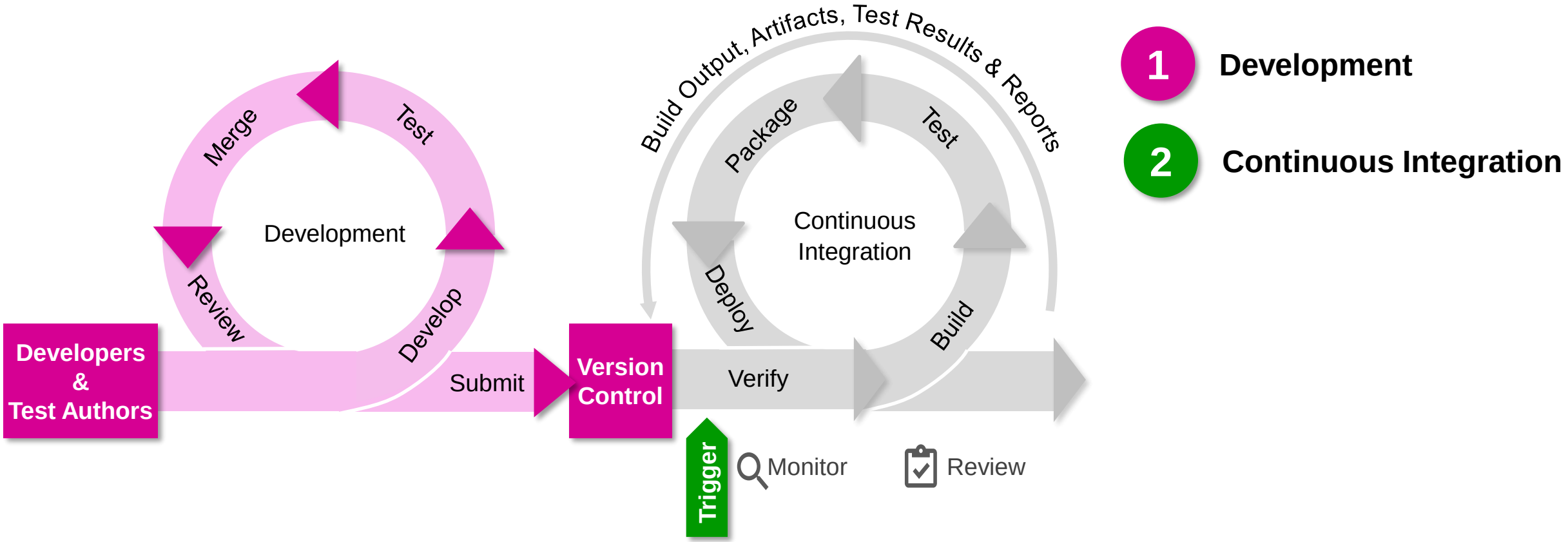


1 **Development**

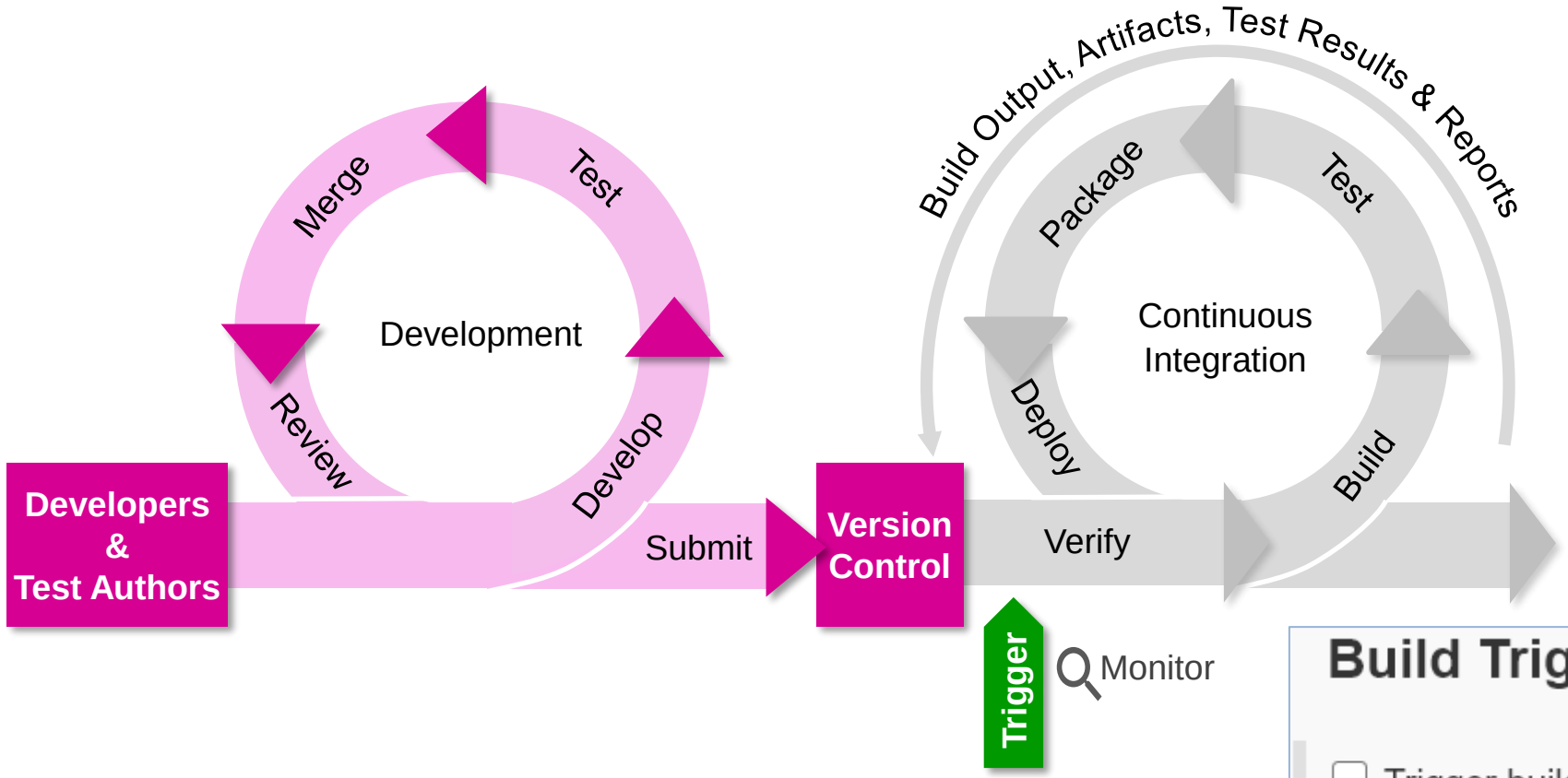
Continuous Integration Workflow with Model-Based Design



Continuous Integration Workflow with Model-Based Design



Continuous Integration Workflow with Model-Based Design



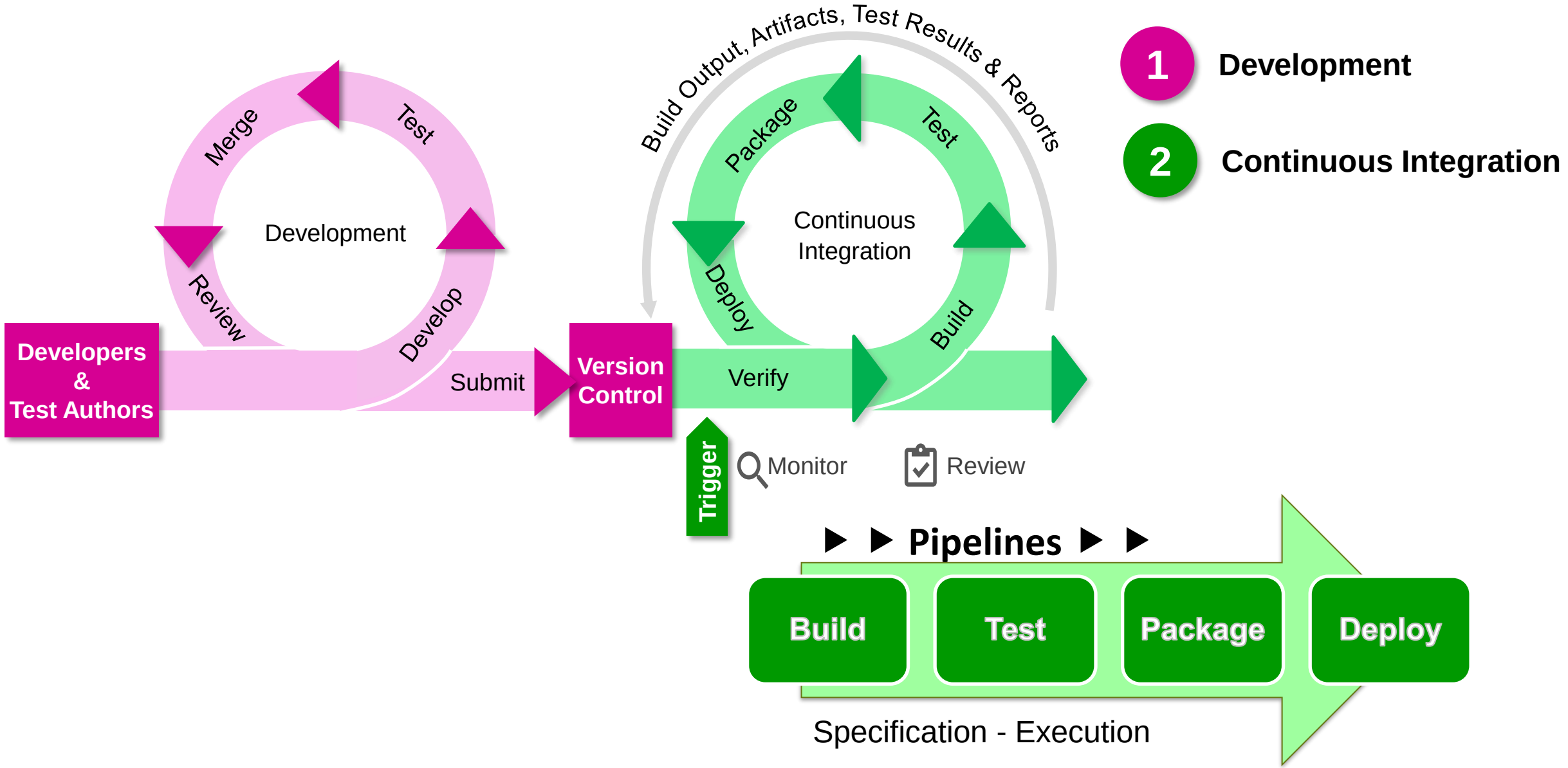
1 **Development**

2 **Continuous Integration**

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☒ Poll SCM

Continuous Integration Workflow with Model-Based Design



CI Servers vs CI Services

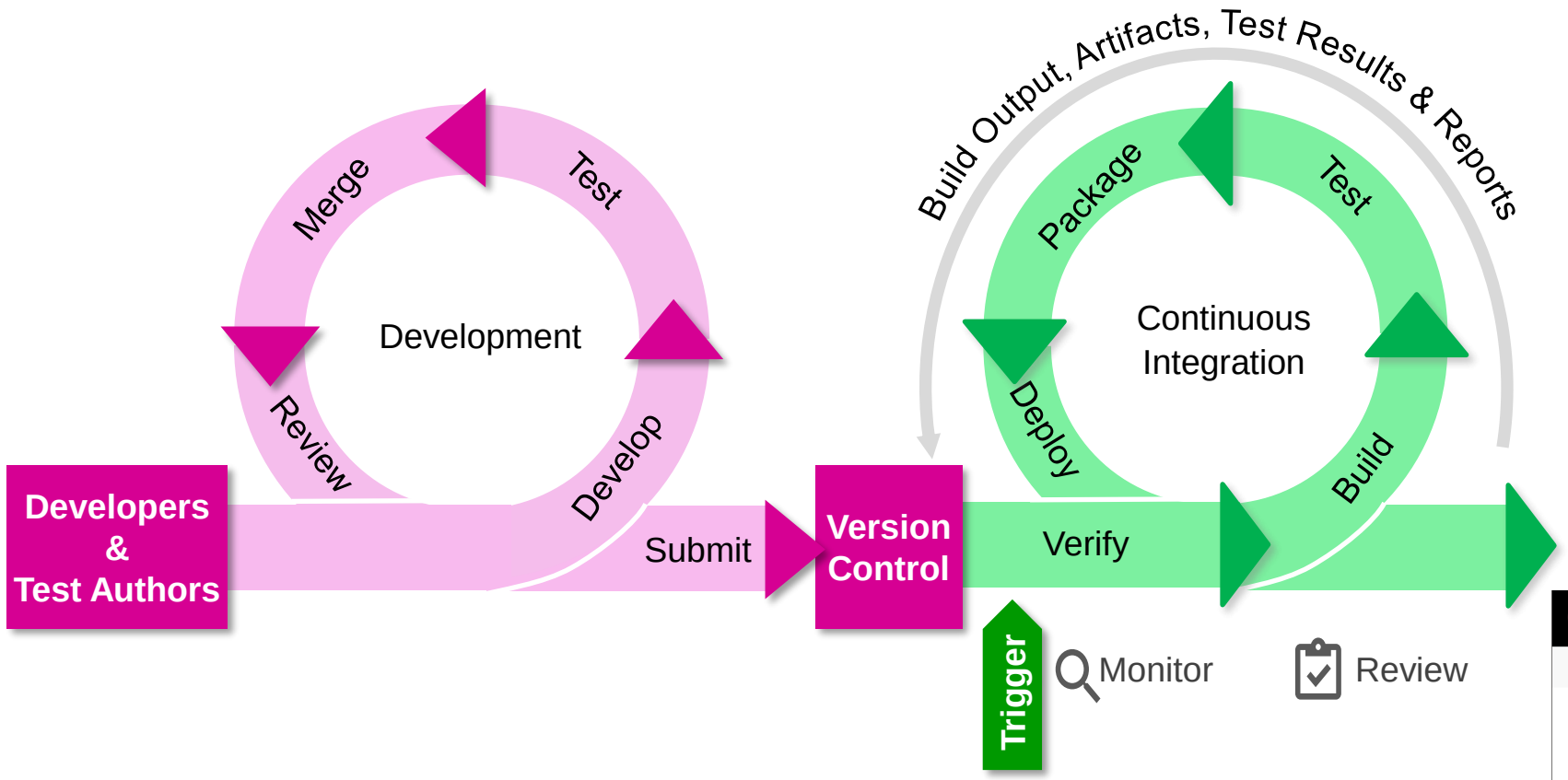
	CI Servers (Local CI)
Setup	Primary + Agents on your hardware
Pros	▪ No service charge ▪ Code privacy
Cons	▪ Install + maintenance ▪ Harder to scale
Ex	▪ Jenkins ▪ Bamboo ▪ GitLab CI ▪ TeamCity

	CI Services (Cloud CI)
Setup	Primary + Agents on cloud hardware
Pros	▪ No installation + maintenance ▪ Easy to scale (↓ complexity) ▪ Service support
Cons	▪ Service costs ▪ Code privacy issues
Ex	▪ Azure DevOps ▪ GitHub Actions ▪ CircleCI ▪ Travis CI

CI Servers vs CI Services

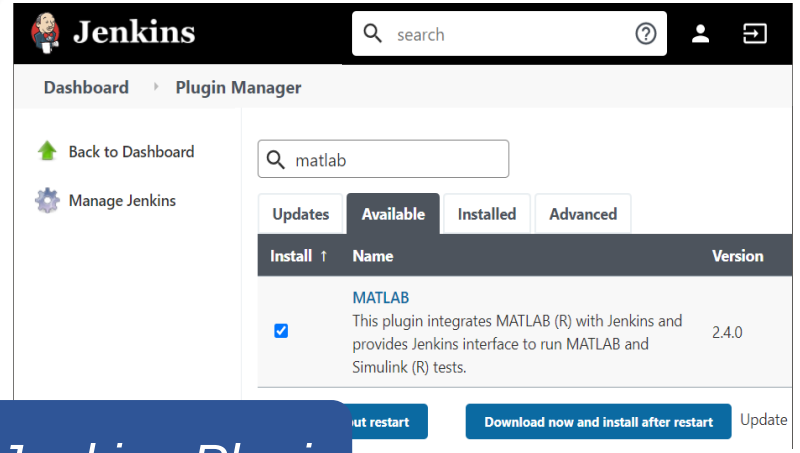
	CI Servers (Local CI)
Setup	Primary + Agents on your hardware
Pros	▪ No service charge ▪ Code privacy 
Cons	▪ Install + maintenance ▪ Harder to scale 
Ex	▪ Jenkins ▪ Bamboo ▪ GitLab CI ▪ TeamCity 

Continuous Integration Workflow with Model-Based Design



1 **Development**

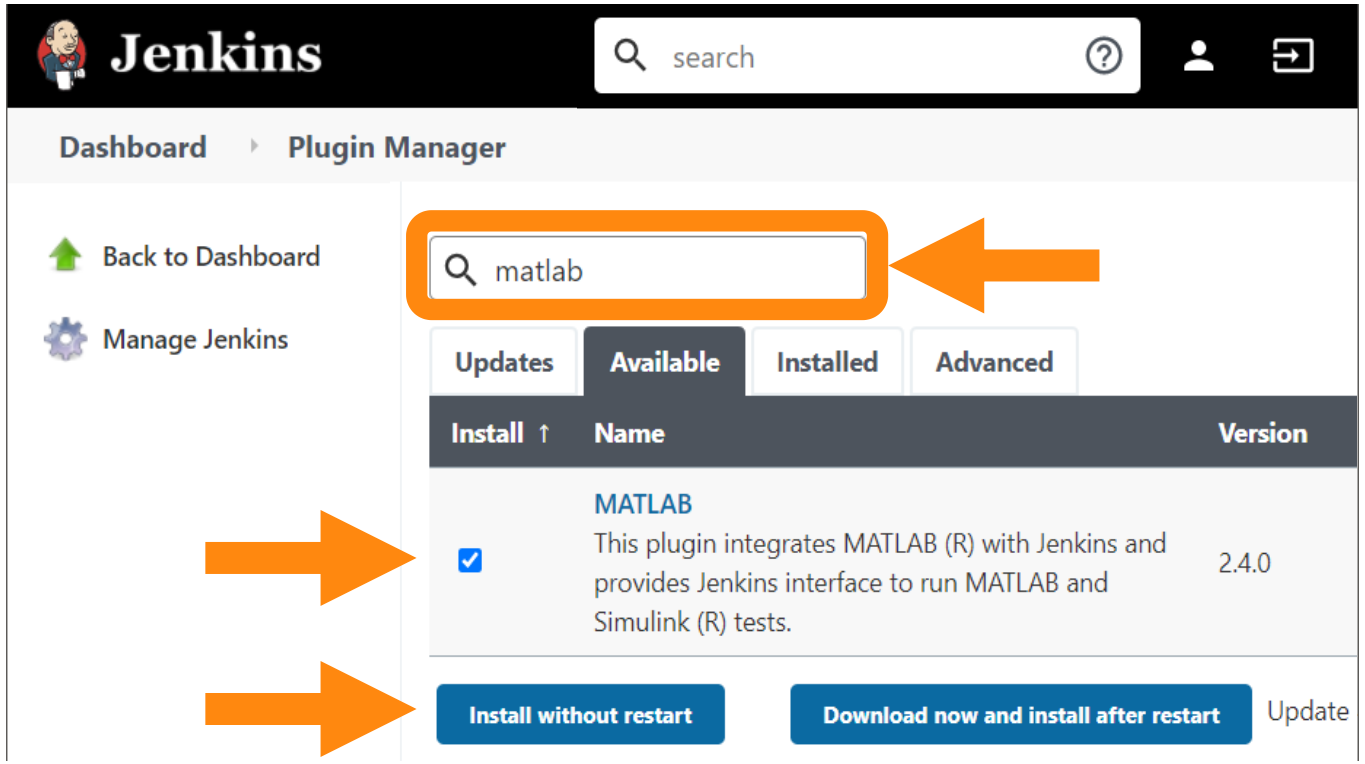
2 **Continuous Integration**



MATLAB Jenkins Plugin

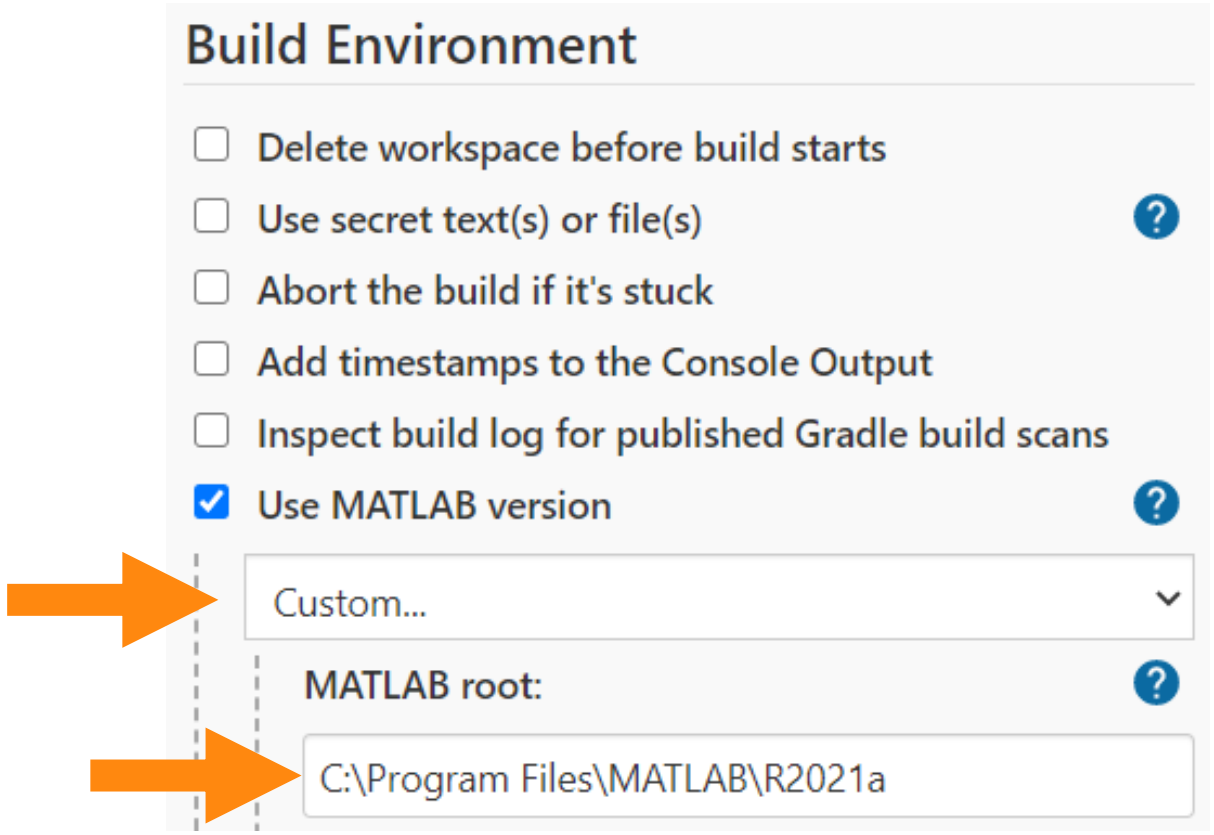
Using the MATLAB plugin for Jenkins

- Install the MATLAB plugin



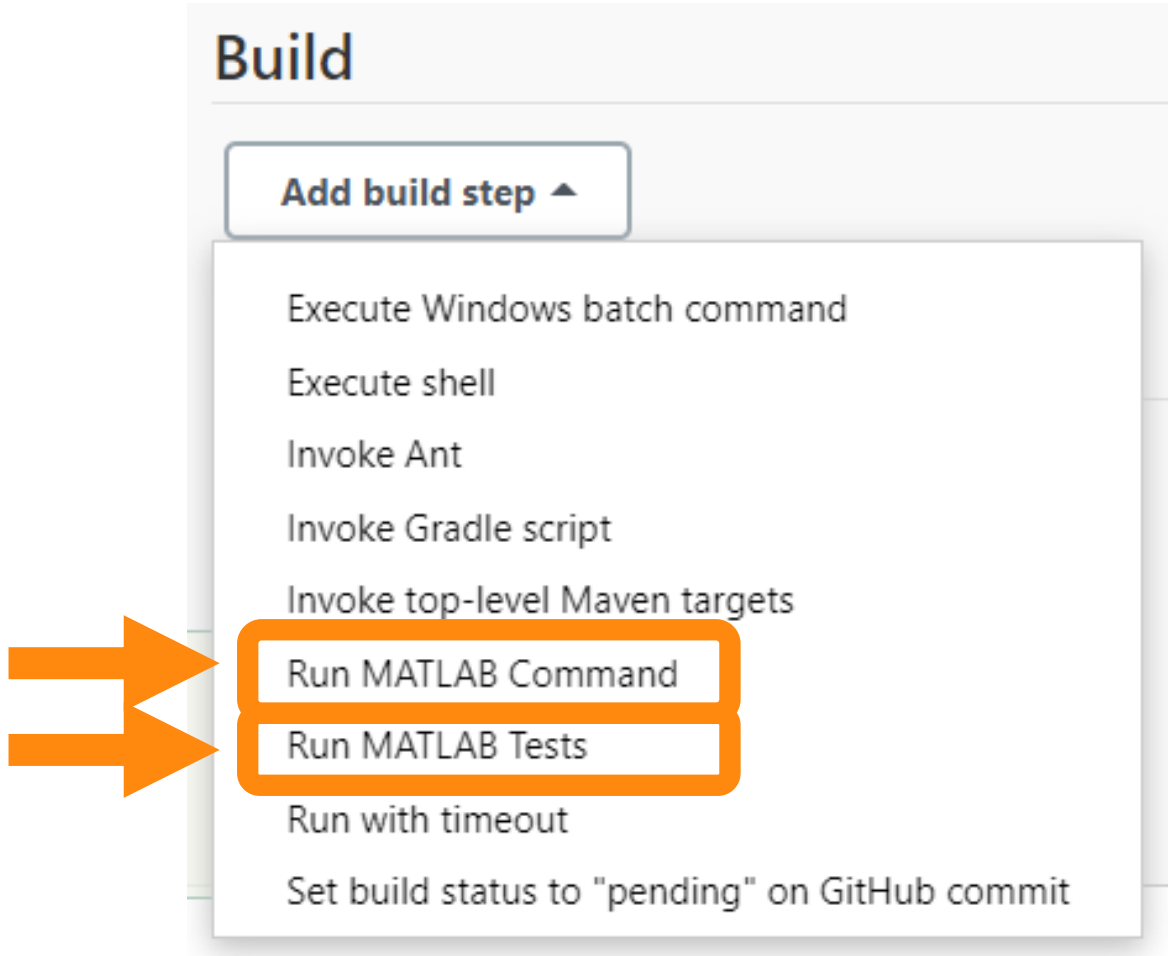
Using the MATLAB plugin for Jenkins

- Install the MATLAB plugin
- Tell Jenkins where to find MATLAB



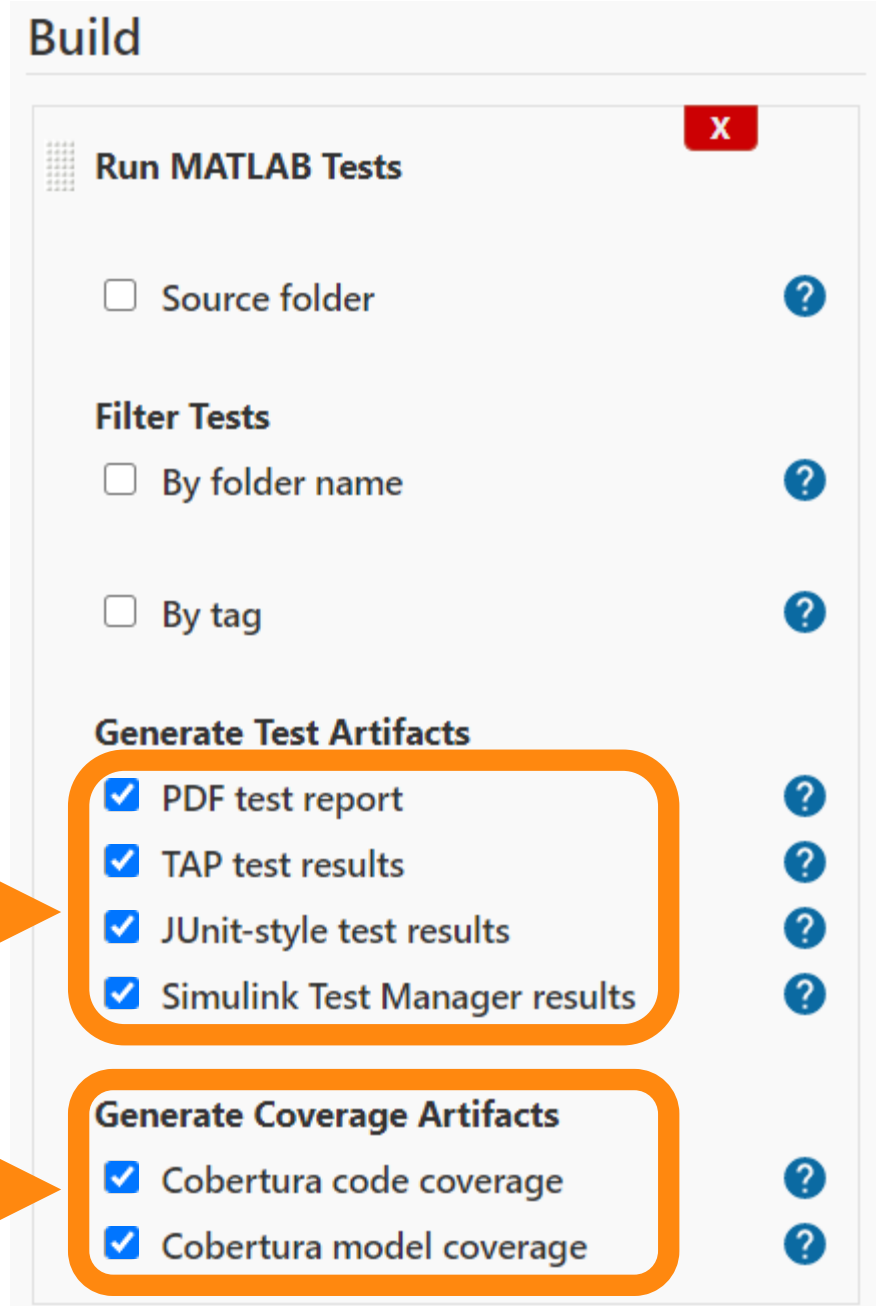
Using the MATLAB plugin for Jenkins

- Install the MATLAB plugin
- Tell Jenkins where to find MATLAB
- Tell Jenkins what to do
  - Run all MATLAB tests
  - Run a custom MATLAB command

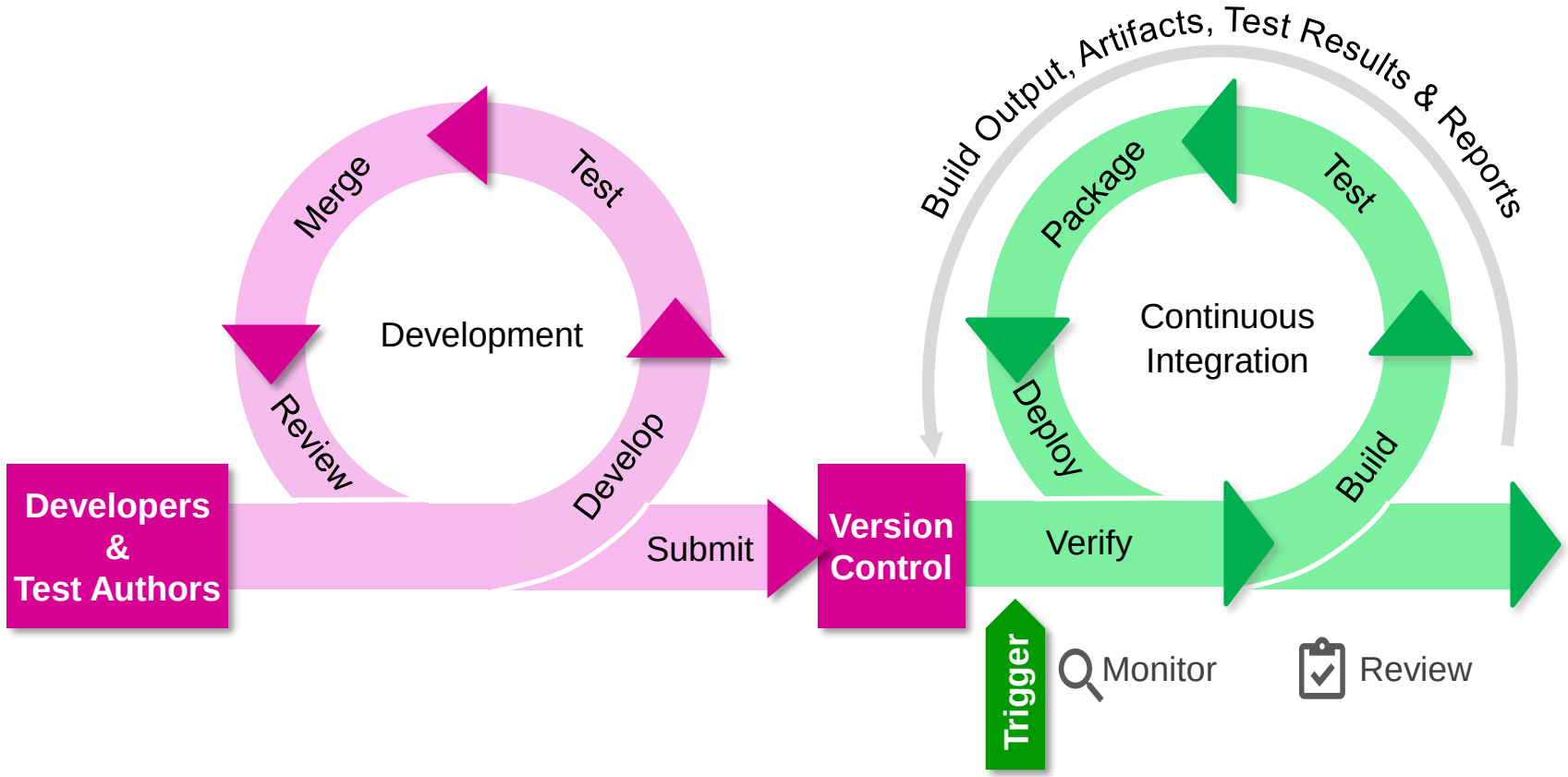


Using the MATLAB plugin for Jenkins

- Install the MATLAB plugin
- Tell Jenkins where to find MATLAB
- Tell Jenkins what to do
  - Run all MATLAB tests
  - Run a custom MATLAB command
- Choose which test results you want to see

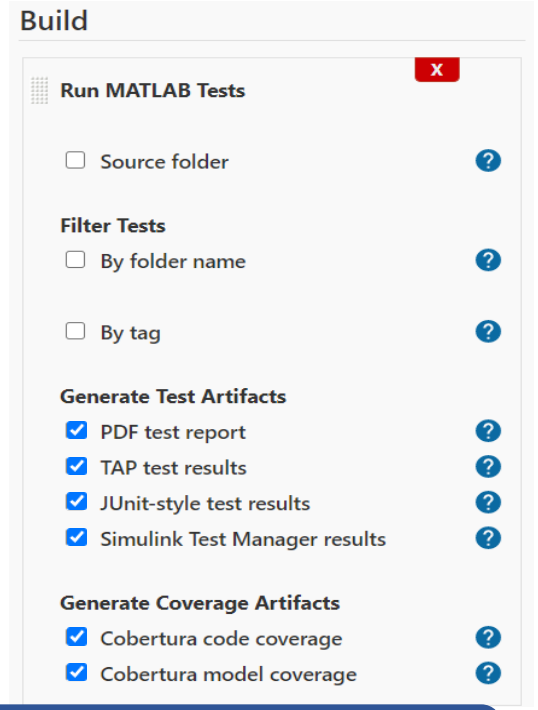


Continuous Integration Workflow with Model-Based Design



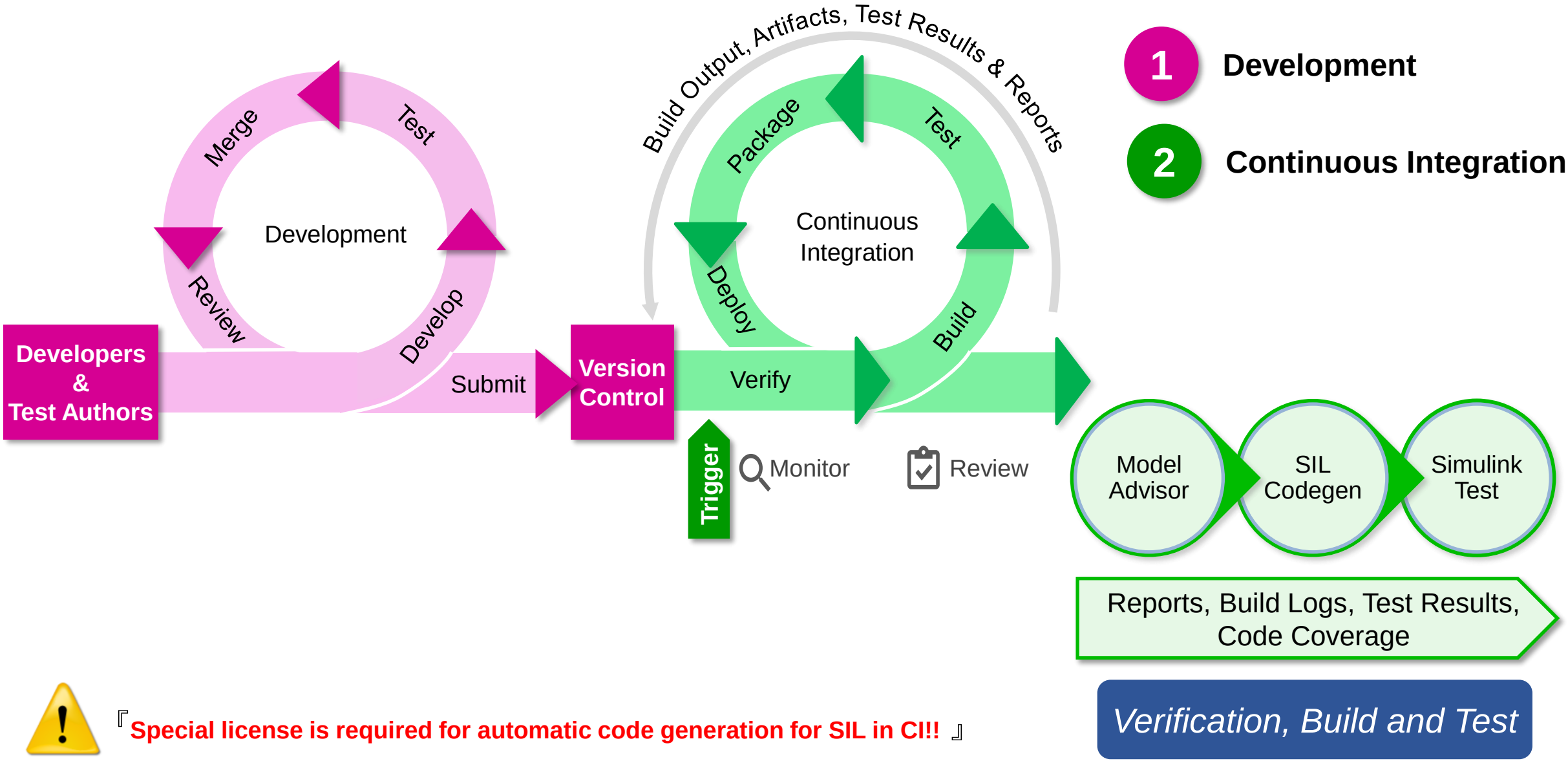
1 **Development**

2 **Continuous Integration**

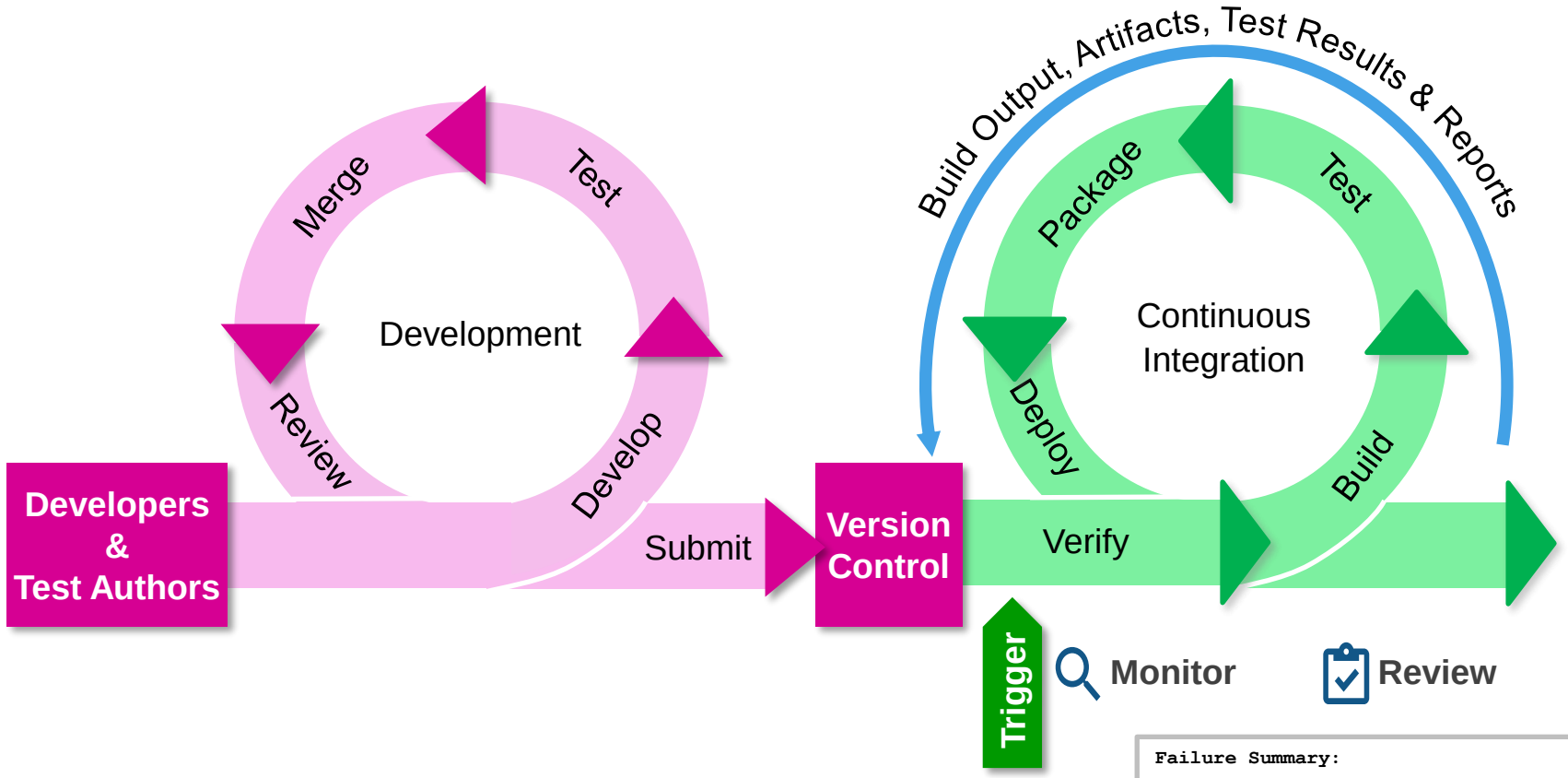


Verification, Build and Test

Continuous Integration Workflow with Model-Based Design



Continuous Integration Workflow with Model-Based Design

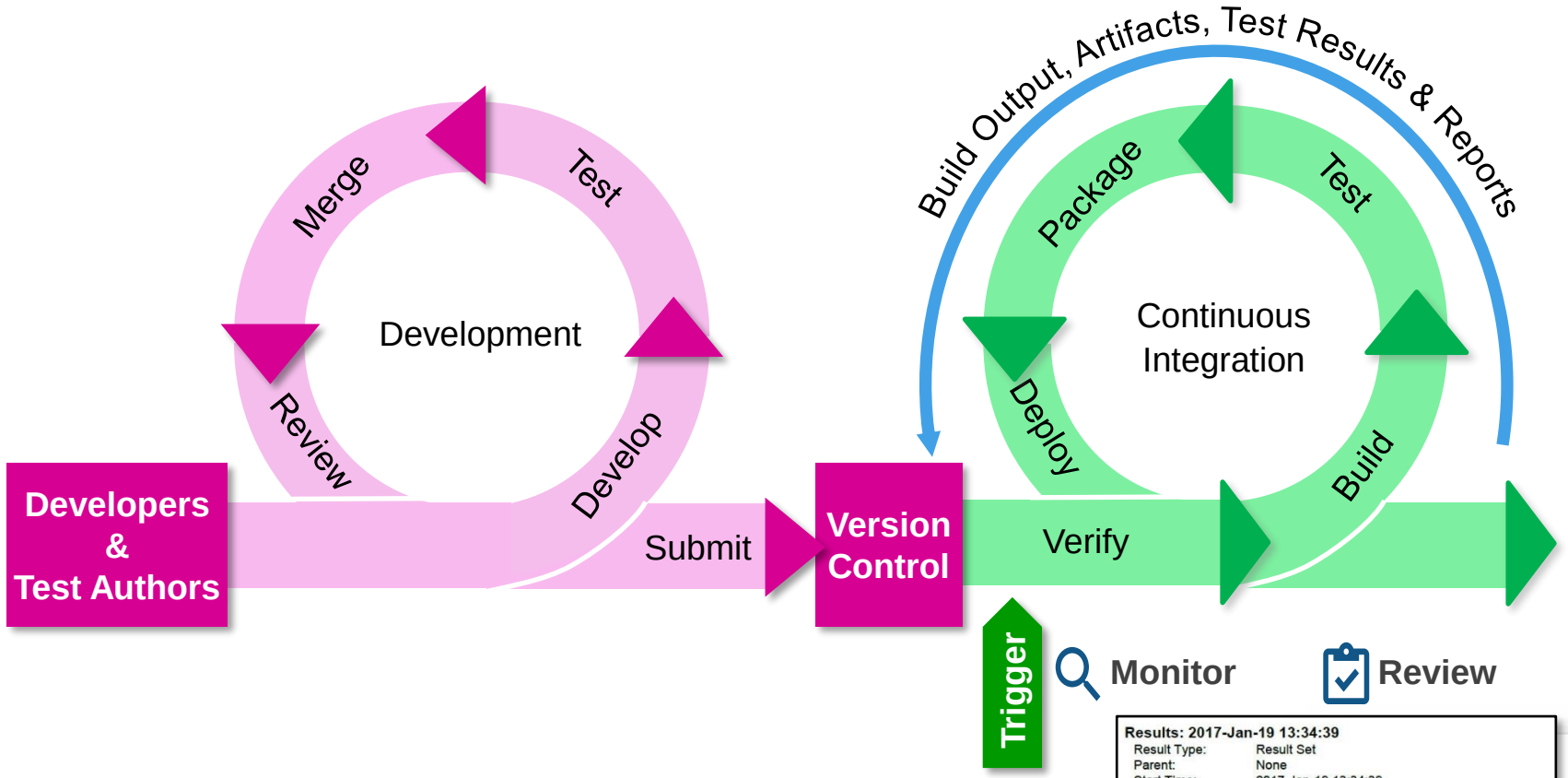


1. **Development**
2. **Continuous Integration**
3. **Results Monitor and Review**

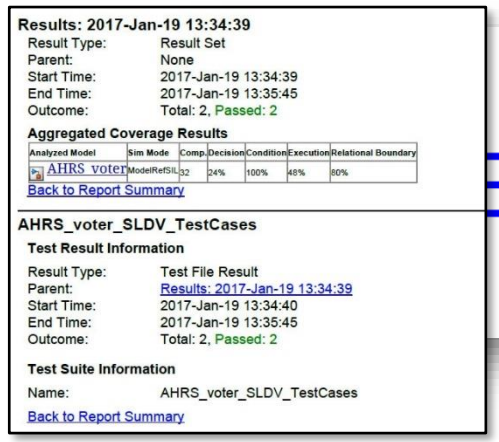
```
Failure Summary:

     Name                                                                          Failed
================================================================================================
     LaneFollowingTestScenarios > Scenarios/LFACC_Curve_CutInOut_TooClose     X
ERROR: MATLAB error Exit Status: 0x00000001
Build step 'Run MATLAB Tests' changed build result to FAILURE
Finished: FAILURE
```

Text Output

Continuous Integration Workflow with Model-Based Design



1. **Development**
2. **Continuous Integration**
3. **Results Monitor and Review**


Results: 2017-Jan-19 13:34:39
Result Type: Result Set
Parent: None
Start Time: 2017-Jan-19 13:34:39
End Time: 2017-Jan-19 13:35:45
Outcome: Total: 2, Passed: 2

Aggregated Coverage Results

Analyzed Model	Sim Mode	Comp.	Decision	Condition	Execution	Relational Boundary
AHRS_voter	ModelRefSIL	32	24%	100%	48%	80%

Back to Report Summary

AHRS_voter_SLDV_TestCases

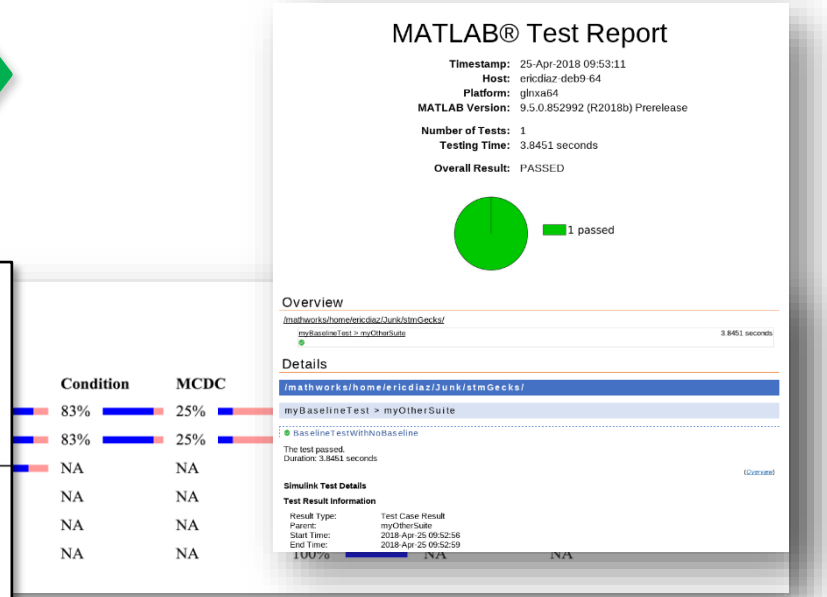
Test Result Information

Result Type: Test File Result
Parent: Results: 2017-Jan-19 13:34:39
Start Time: 2017-Jan-19 13:34:40
End Time: 2017-Jan-19 13:35:45
Outcome: Total: 2, Passed: 2

Test Suite Information

Name: AHRS_voter_SLDV_TestCases

Back to Report Summary


MATLAB® Test Report

Timestamp: 25-Apr-2018 09:53:11
Host: ericdiaz-deb9-64
Platform: glnxa64
MATLAB Version: 9.5.0.852992 (R2018b) Prerelease

Number of Tests: 1
Testing Time: 3.8451 seconds

Overall Result: PASSED

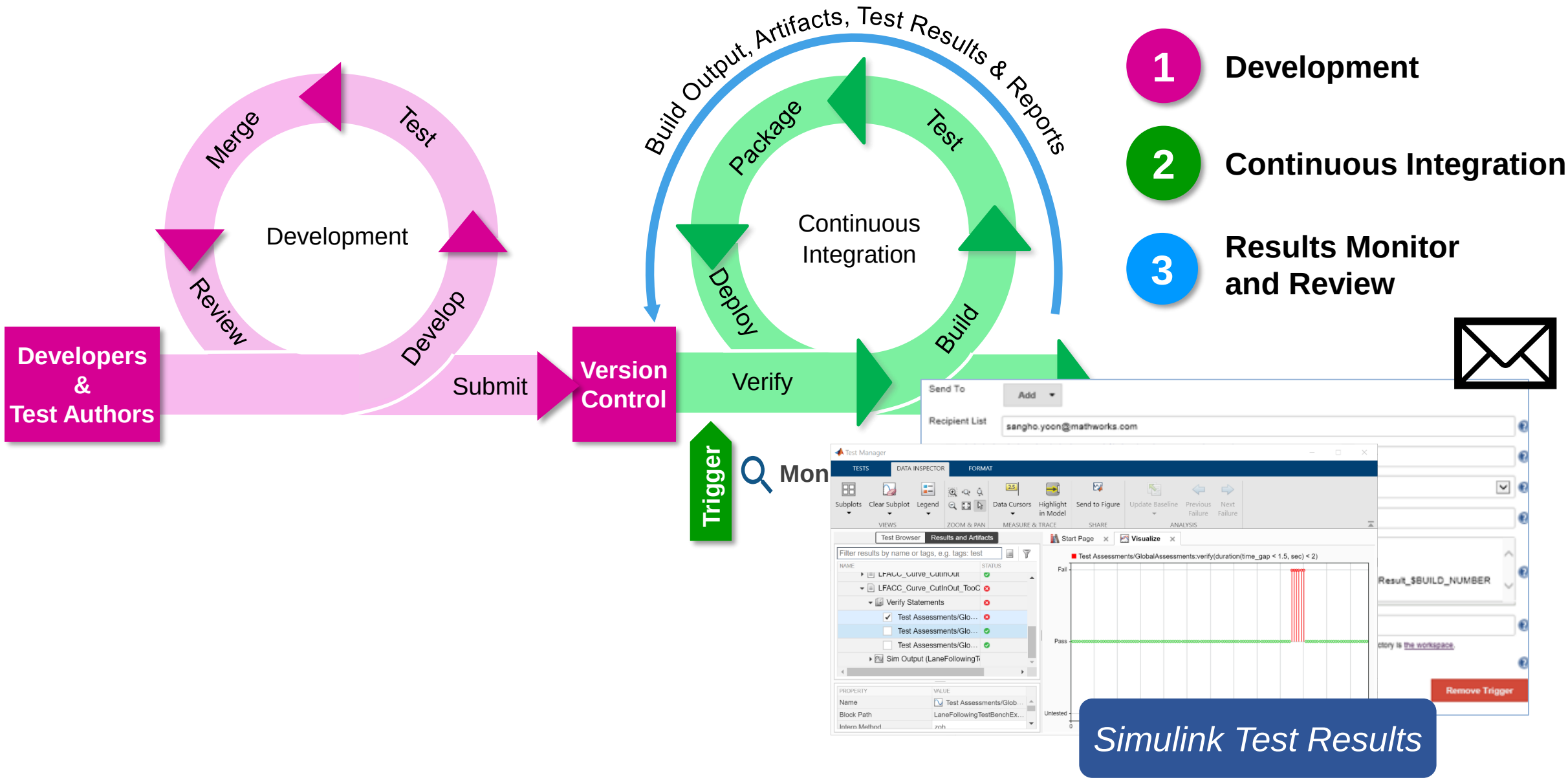
1 passed

Overview

Details

Condition MCDC
83% 25%
83% 25%
NA NA
NA NA
NA NA
NA NA

Continuous Integration Workflow with Model-Based Design



What's the easiest way to get started with CI?

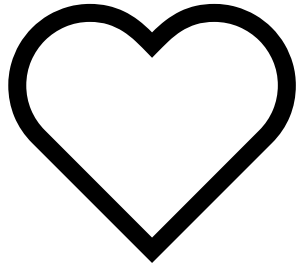
- Use our plugins to streamline CI job setup for MATLAB and Simulink
  - Azure DevOps
  - Jenkins
  - CircleCI
  - Travis CI
  - GitHub Actions
- Don't worry, you can still use MATLAB and Simulink with other CI platforms!
  - The plugins just make it *easier*
- Reference architectures to get you started with cloud-based hosts
  - AWS, Azure, and Google Cloud Platform (GCP)
  - https://github.com/mathworks-ref-arch

Free MATLAB CI support for public GitHub repositories

- MathWorks now provides free CI support for MATLAB and Simulink if:
  1. your code and models are publicly available on GitHub
  2. your CI build is publicly available on a supported CI platform
  3. you don't use any excluded products (e.g., MATLAB Compiler, MATLAB Coder)

- Currently supported CI platforms:
  - Azure DevOps
  - CircleCI
  - GitHub Actions
  - Travis CI

MATLAB



GitHub
Community

Can commercial customers use the free CI service?

- Yes and no…
- Why no?
  - Free CI service requires you share your code publicly
  - You don't want to give away your intellectual property
- Why yes?
  - Use "toy" examples to explore potential future cloud migration
- Remember, our plugins support on-premise CI servers and private agents

This sounds great, but how do I get started?

- Check out our example GitHub repository!
  - http://github.com/mathworks/ci-configuration-examples

- The example repository provides:
  - A quick start guide
  - Ready-to-use CI configuration files
  - Example MATLAB code and tests
  - CI badge examples, with helpful links

MATLAB CI Configuration Examples

This repository shows how to run MATLAB tests with a variety of continuous integration systems.

CI Platform	Badges	Badge Help
Azure DevOps	Azure Pipelines succeeded coverage 80%	Blog with helpful information for setting up Azure DevOps badges
CircleCI	circleci passing	CircleCI documentation for setting up badges
GitHub Actions	MATLAB passing	GitHub Actions documentation for setting up badges
Travis CI	build passing	Travis CI documentation for setting up badges

CI Servers vs CI Services

	CI Servers (Local CI)
Setup	Primary + Agents on your hardware
Pros	▪ No service charge ▪ Code privacy
Cons	▪ Install + maintenance ▪ Harder to scale
Ex	▪ Jenkins ▪ Bamboo ▪ GitLab CI ▪ TeamCity

	CI Services (Cloud CI)
Setup	Primary + Agents on cloud hardware
Pros	▪ No installation + maintenance ▪ Easy to scale (↓ complexity) ▪ Service support
Cons	▪ Service costs ▪ Code privacy issues
Ex	▪ Azure DevOps ▪ GitHub Actions ▪ CircleCI ▪ Travis CI

CI Servers vs CI Services

	CI Services (Cloud CI)
Setup	Primary + Agents on cloud hardware
Pros	▪ No installation + maintenance ▪ Easy to scale (↓ complexity) ▪ Service support
Cons	▪ Service costs ▪ Code privacy issues
Ex	▪ Azure DevOps ▪ GitHub Actions ▪ CircleCI ▪ Travis CI

GitHub Actions Demo!

Example CI configuration file

```yaml
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest

    # Steps represent a sequence of tasks that will be executed as part of the job
    steps:
    # Checks-out your repository under $GITHUB_WORKSPACE, s
    - uses: actions/checkout@v2

    # Sets up MATLAB on the GitHub Actions runner
    - name: Setup MATLAB
      uses: matlab-actions/setup-matlab@v0

    # Runs a set of commands using the runners shell
    - name: Run all tests
      uses: matlab-actions/run-tests@v0
      with:
        source-folder: main
```

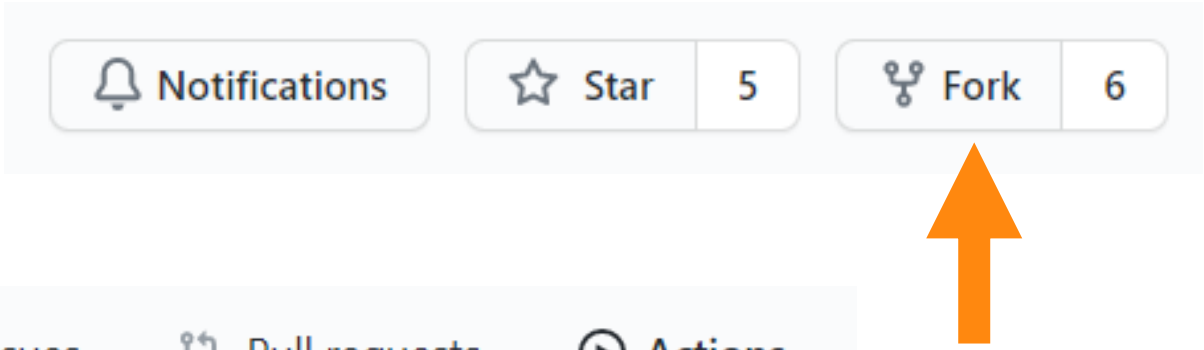
1) Check out your code

2) Setup MATLAB

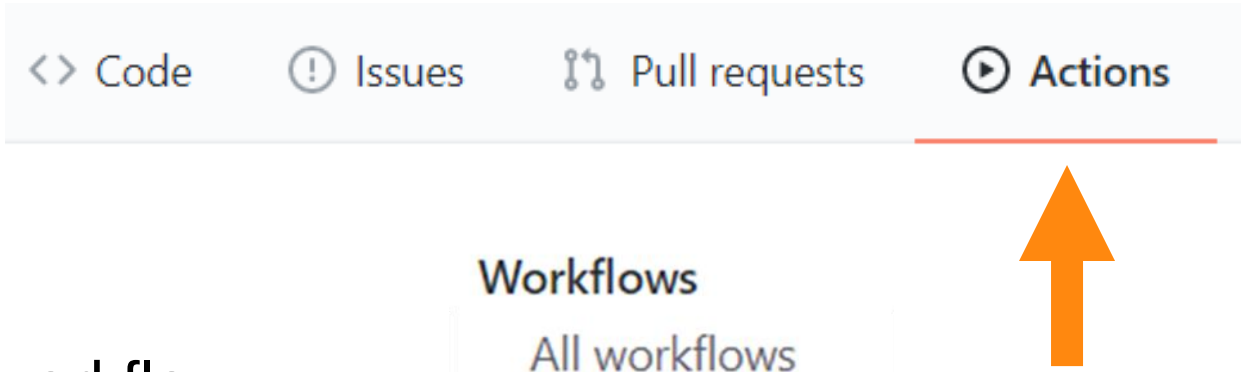
3) Run all your tests

GitHub Actions Quick Start Guide

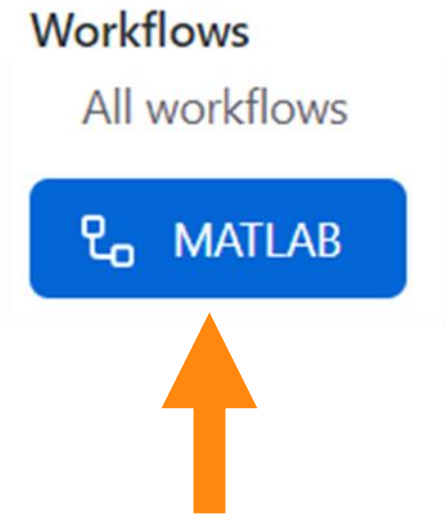
1. Fork our example repository



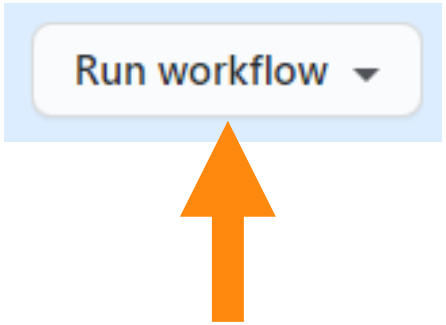
2. Go to Actions

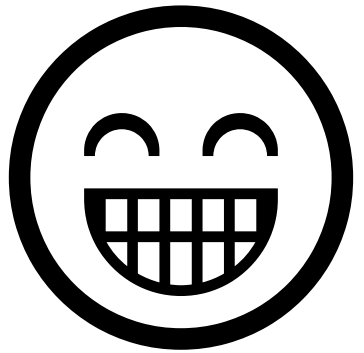


3. Choose the "MATLAB" workflow



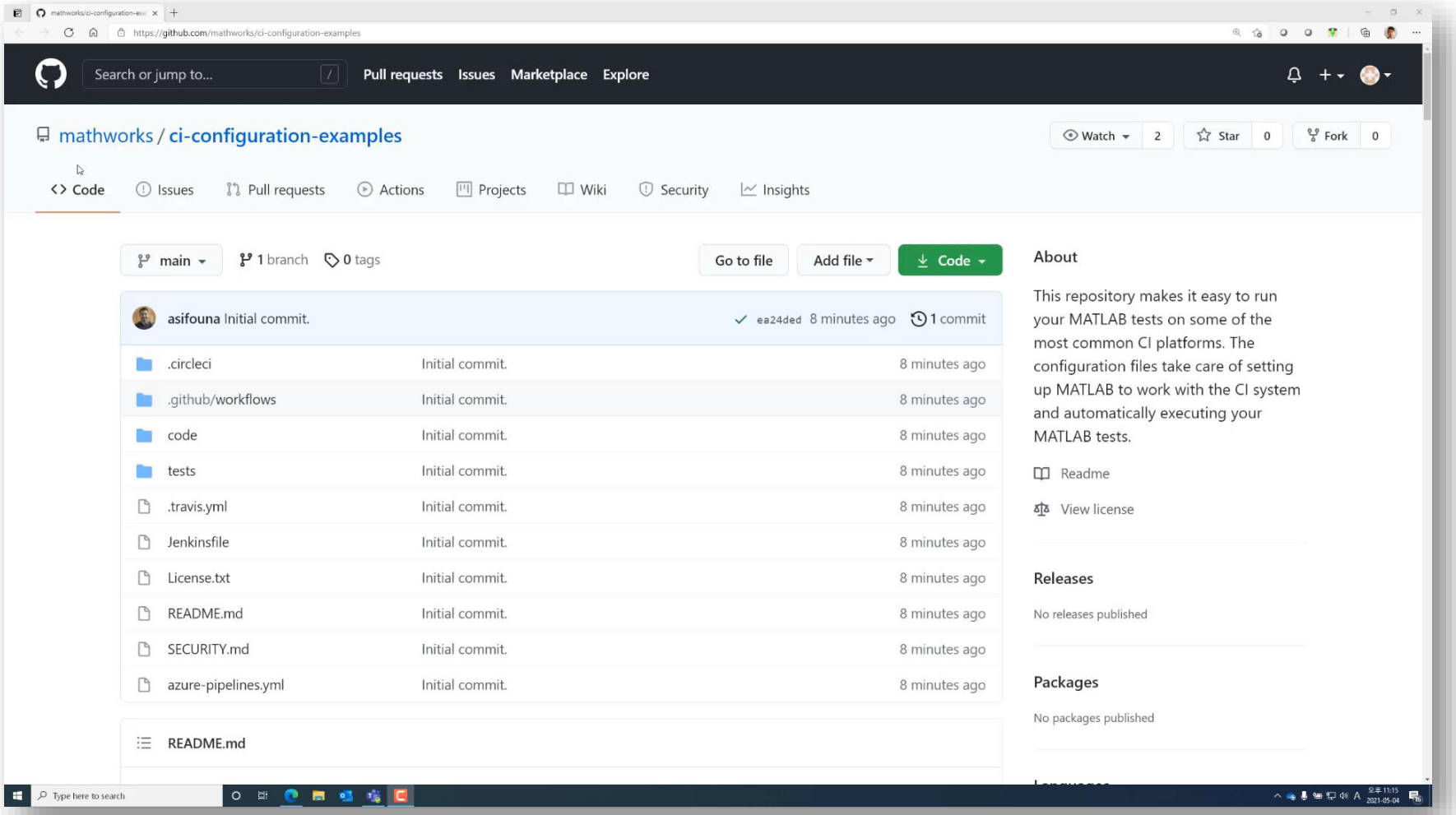
4. Run workflow!



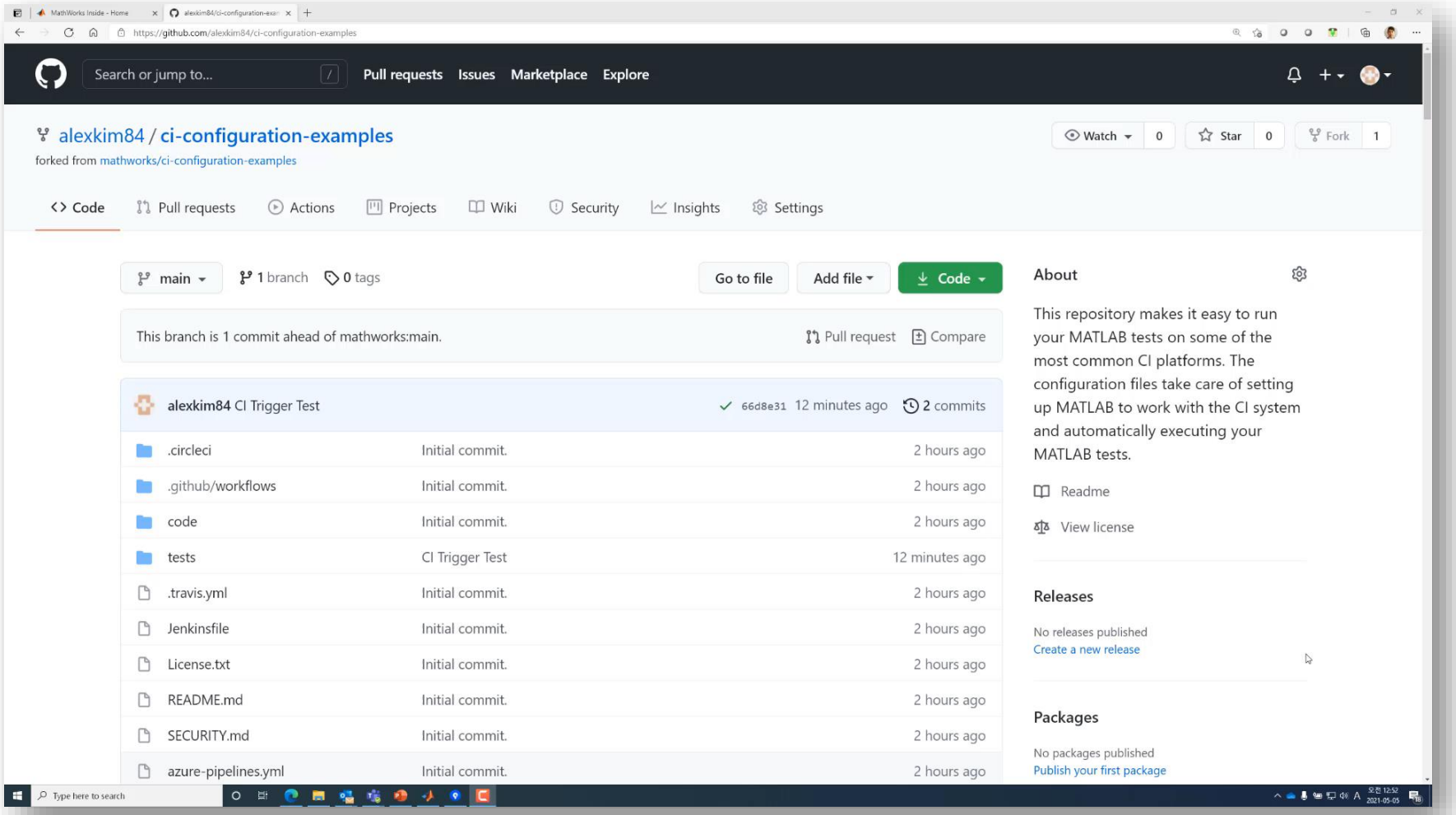


Totally easy, right?

DEMO: GitHub Actions Quick Start Guide (1)



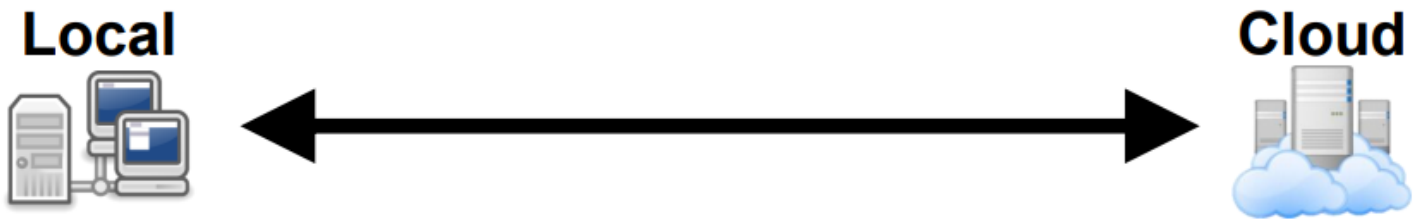
DEMO: GitHub Actions quick start guide (2)



Extending our example to your code and models

- This repository was designed to be easily extendable
  - Replace our code and tests with your code and tests
  - Commit
  - Push
- No need to modify any configuration files!
- CI job automatically triggered by changes to your GitHub repository

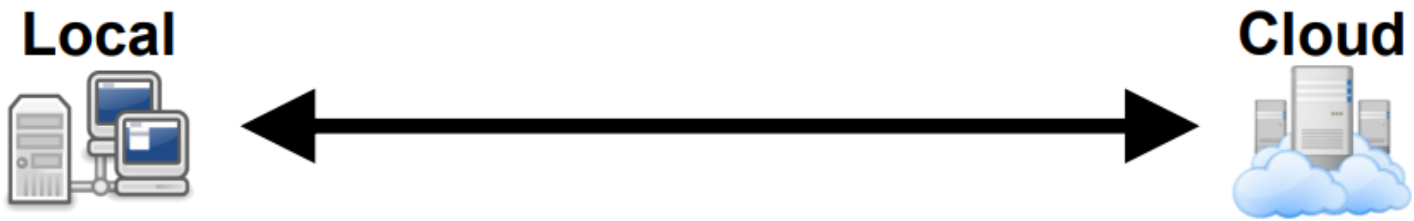
CI Servers vs CI Services



	CI Servers (Local CI)
Setup	Primary + Agents on your hardware
Pros	▪ No service charge ▪ Code privacy
Cons	▪ Install + maintenance ▪ Harder to scale
Ex	▪ Jenkins ▪ Bamboo ▪ GitLab CI ▪ TeamCity

	CI Services (Cloud CI)
Setup	Primary + Agents on cloud hardware
Pros	▪ No installation + maintenance ▪ Easy to scale (↓ complexity) ▪ Service support
Cons	▪ Service costs ▪ Code privacy issues
Ex	▪ Azure DevOps ▪ GitHub Actions ▪ CircleCI ▪ Travis CI

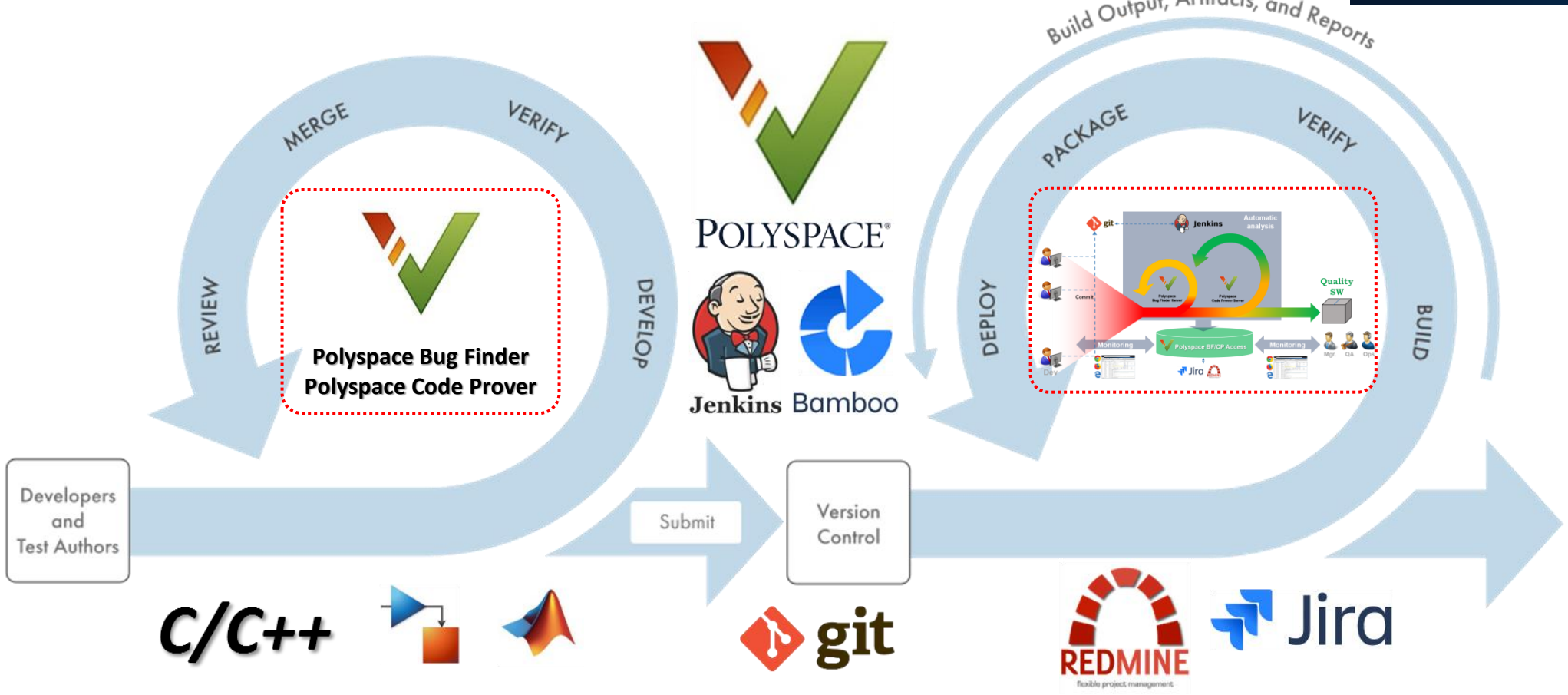
CI Servers vs CI Services



	Cloud CI w/ local agents
Setup	Primary on cloud hardware Agents on local hardware
Pros	▪ Easier to scale (less complexity) ▪ Service support ▪ Code privacy
Cons	▪ Hardware setup + maintenance ▪ Expensive to scale (hardware)
Ex	▪ GitHub Actions ▪ Azure DevOps ▪ CircleCI

Support *MATLAB Plugin*

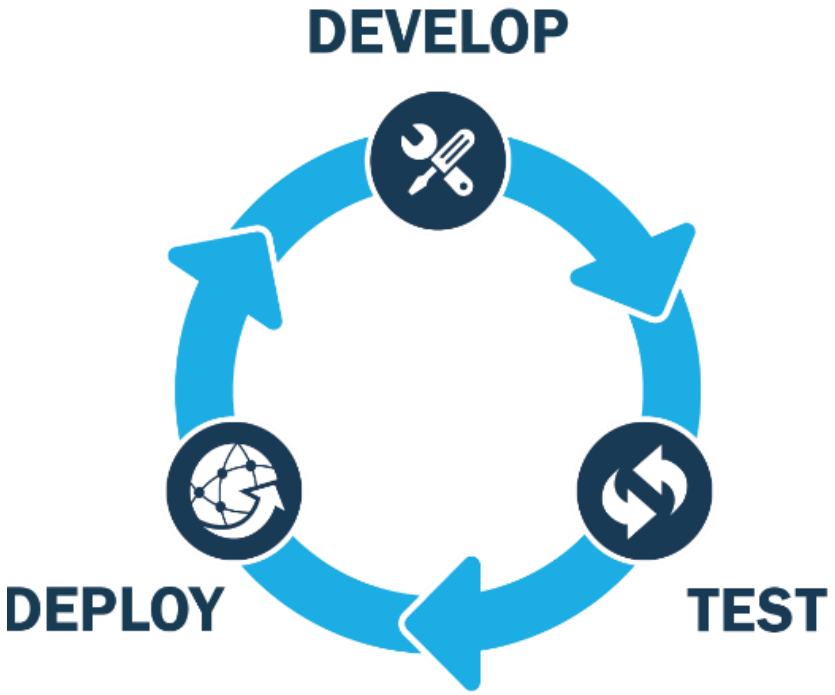
CI Workflow for Static Analysis with Polyspace



Summary and Resources

Key Takeaways

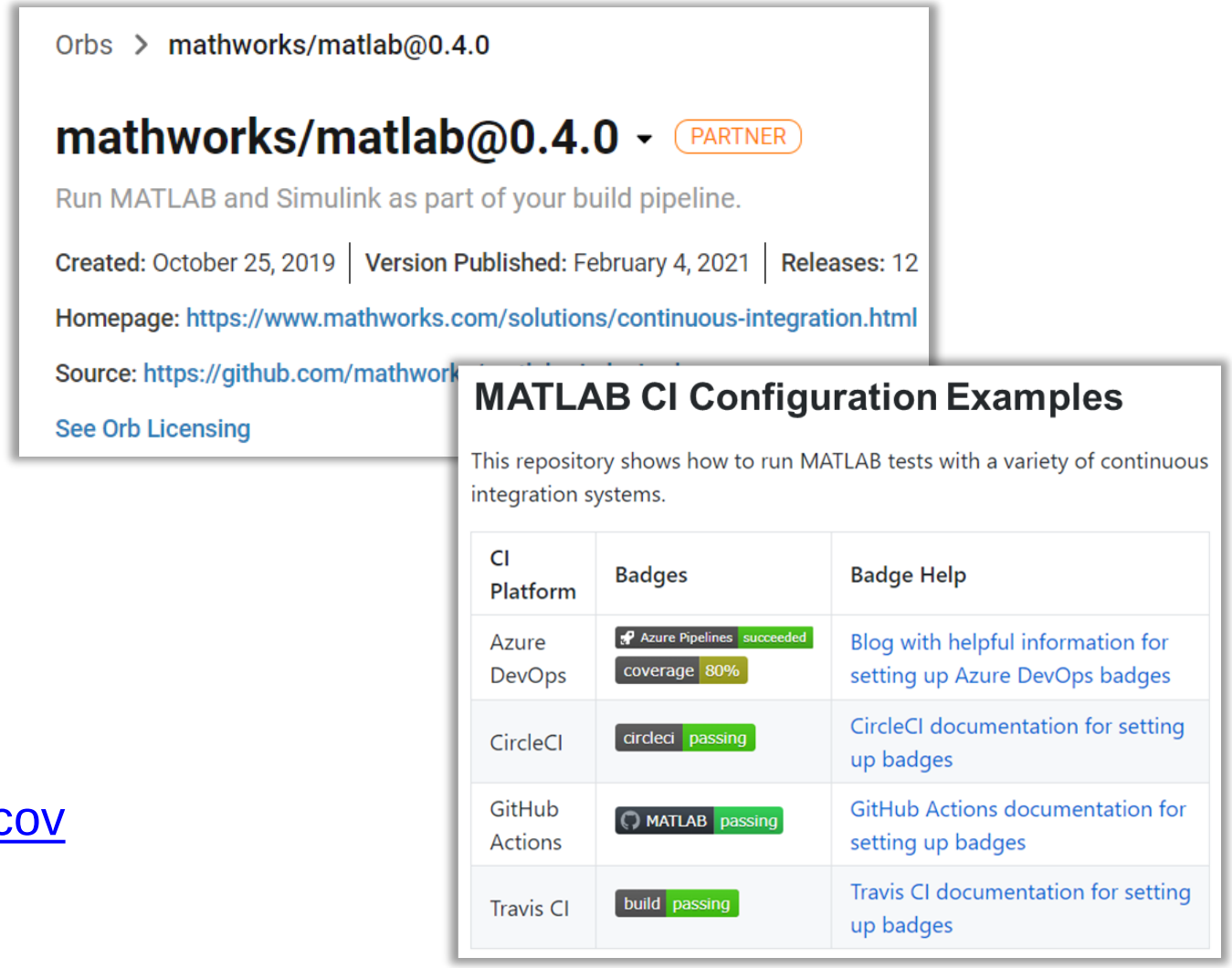
- Continuous integration helps you develop high quality software, faster!
- MATLAB and Simulink support you through all stages of CI
- Getting started with CI is easy with our plugins and example codes
- Free CI support for public MATLAB and Simulink GitHub repositories!



Getting Started: CI plugins and code examples

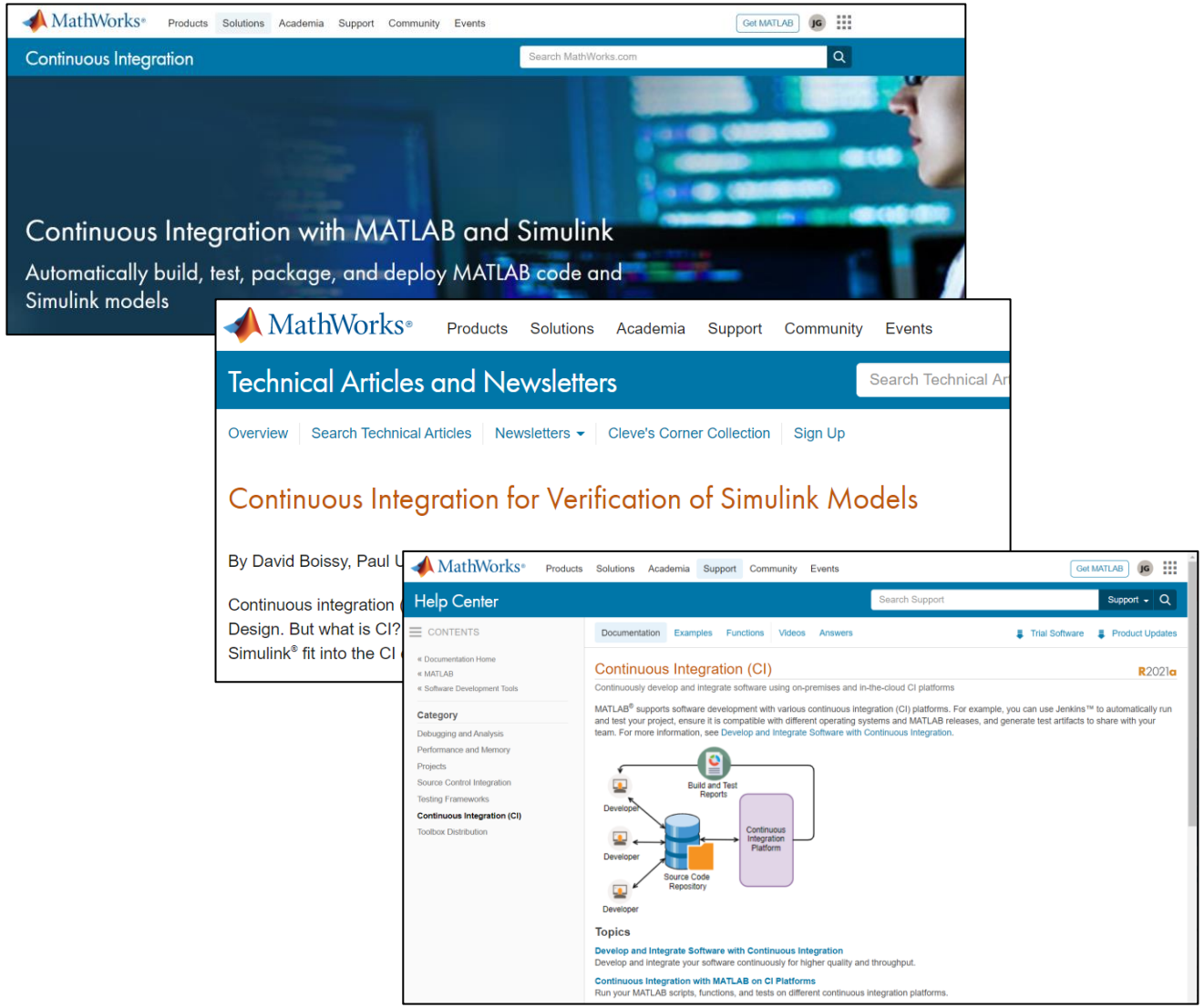
- CI plugins
  - Azure DevOps
  - CircleCI
  - GitHub Actions
  - Jenkins
  - Travis CI

- Code examples
  - CI configuration examples
  - CI with Simulink
  - Code coverage using Codecov



Learning More: Continuous Integration with MATLAB and Simulink

- Solutions Page:
  - Continuous Integration Solution Page
- White Papers:
  - Continuous Integration for Verification of Simulink Models
  - Agile Model-Based Design: Accelerating Simulink Simulations in CI Workflows
- Documentation and Blogs:
  - Continuous Integration Documentation Hub
  - Developer Zone: Continuous Integration
  - CI with Projects and Simulink Test



MATLAB EXPO 2021

Thank you