Bringing CI/CD to Digital Engineering

Unit Testing, Model Assessments, & Build Automation

Robert Peters
Principal Engineer

Catherine Haggerty
Software Engineer

Mark Petrotta
Principal Engineer

Agenda

- About Us
- What We Can Learn From Software
- Applying CI/CD to MBSE
- "Hello World" Example
- Limitations & Future Considerations

About Us



- 40 years combined experience in software, 20 years in MBSE.
- Have collectively seen the benefits of Continuous Integration Continuous Delivery (CI/CD) in the software world. When we were each brought to the MBSE world, we saw that these things were missing.
- We value quality and efficiency. Early detection, yay!

What We Can Learn From Software

Software engineering nearly broke 20 years ago

- Team Structure
- Version Control Systems
- Review Process
- Merge / Integration Issues
- Build & Test Challenges
- Traceability & Requirements
- Tooling Lock-in

1991-2005

decentralized growth without tools

2001–2010s

heavyweight, centralized, contractor-driven processes

Modern era of distributed version control, modular architectures, CI/CD, and DevSecOps

What We Can Learn From Software

What challenges do engineers face and how does software address them?

Common Challenges

- Manual validation leads to inefficiencies and missed errors.
- Slow feedback loops hinder agility and collaboration.
- Testing processes lack standardization and repeatability.
- Development process needs traceability across lifecycle.

The Software CI/CD Solution

- Automated pipelines catch common errors, inefficiencies, and security risks.
- Automation enables quicker feedback loops fostering agility and collaboration.
- Automated pipelines ensure the same tests run on a standardized environment.
- Entire development process is documented, allowing for strong traceability.

Reduced Errors

Faster Feedback

Standardized Process

Enhanced Traceability

The Software CI/CD Process

Software has developed efficient processes for addressing these engineering challenges.

- Feature requests, bugs, change requests, etc. are recorded in an issue tracking tool
- Source code version history is recorded, and branches are created for each issue
- Source code and test code are developed in conjunction, known as test-driven development
- Source code is automatically tested in a common environment using automation tools
- Releases are published to an integrated artifact repository
- Releases are deployed to active environment

Applying CI/CD to MBSE

Applying CI/CD to MBSE

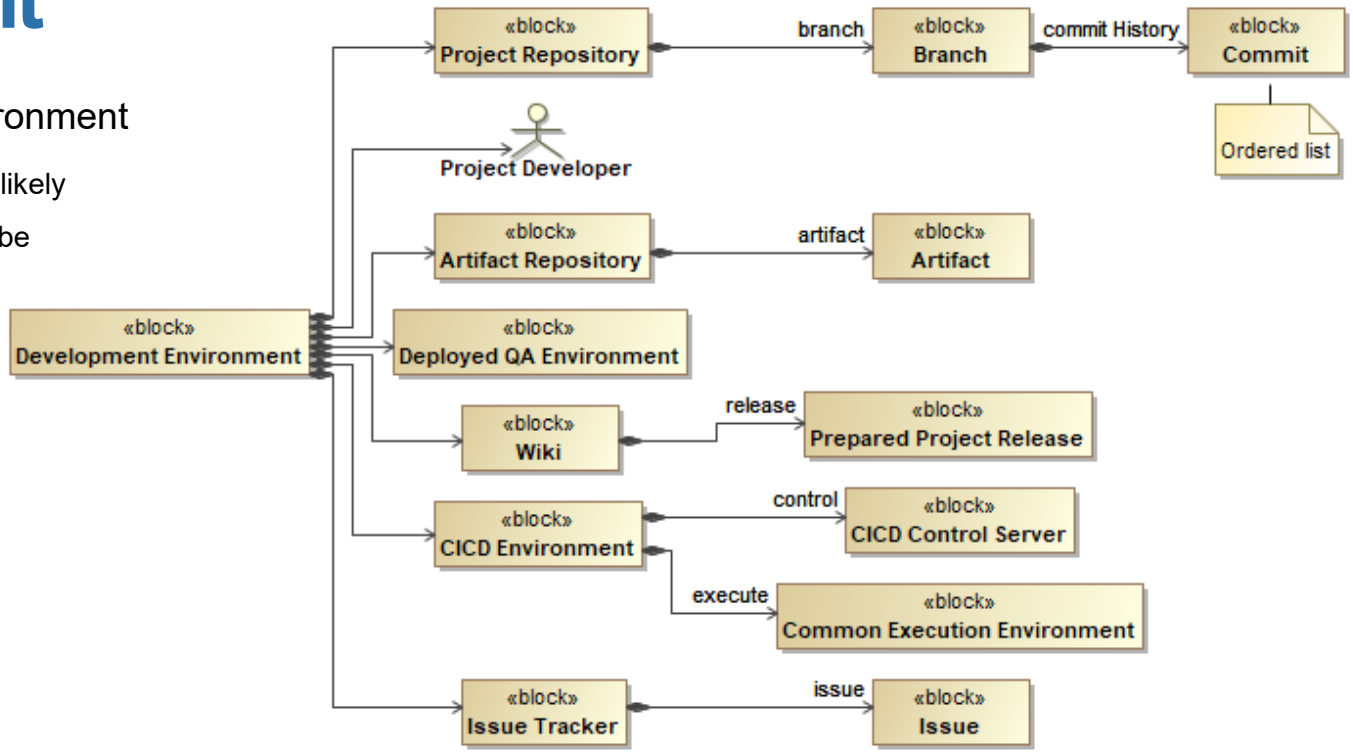
Comparing current software CI/CD solutions with current and future MBSE CI/CD solutions.

Topic	Current Software	Current MBSE	Future MBSE
Issue Tracking	Well-managed	Less formal	Improved traceability
Version Control	Feature-based branching	Minimal branching	Issue-based branching
Test-Driven Development	Unit tests for each method	Validation suites and simulation	Create model with testing in mind
Automated Testing	Runs for every push or pull request, blocks merge	Not enforced	Expanded testing, blocks merge
Integrated Artifact Repository	Build images are stored with strong traceability	Models are stored with limited traceability	MBSE artifacts are stored with strong traceability

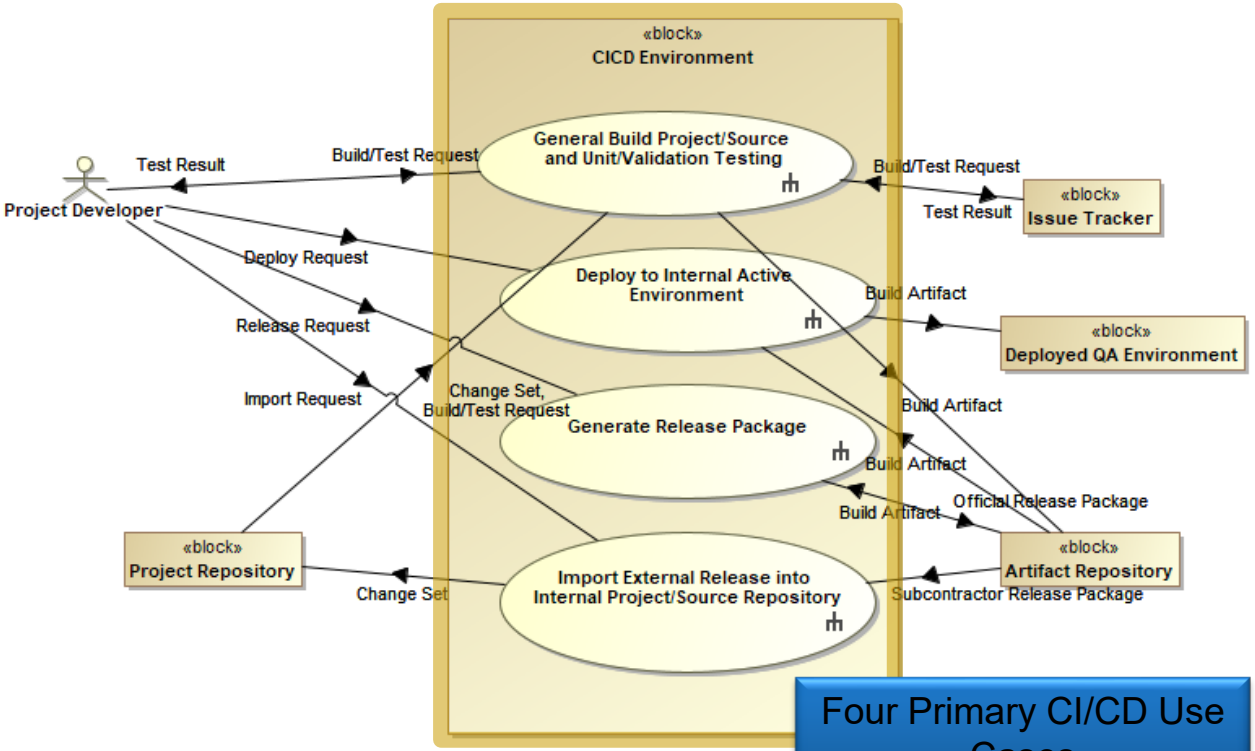
Common Development Environment

Focus on the CI/CD Environment

- Every software or MBSE team likely has each of these, except maybe the CICD Environment

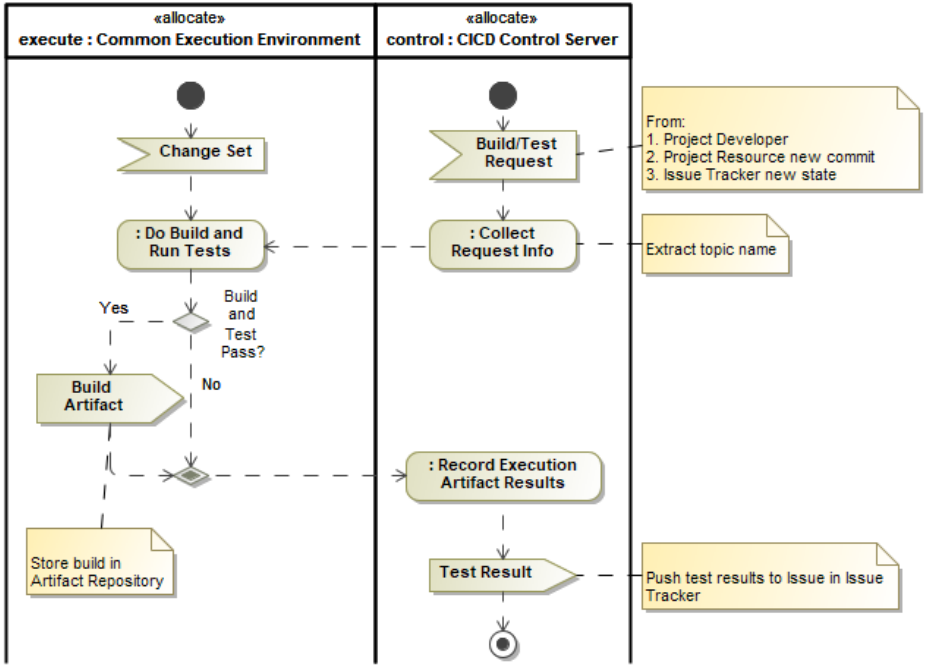


Four Common CI/CD Use Cases for Software and MBSE

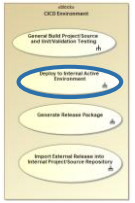


General Build Project/Source and Unit/Validation Testing

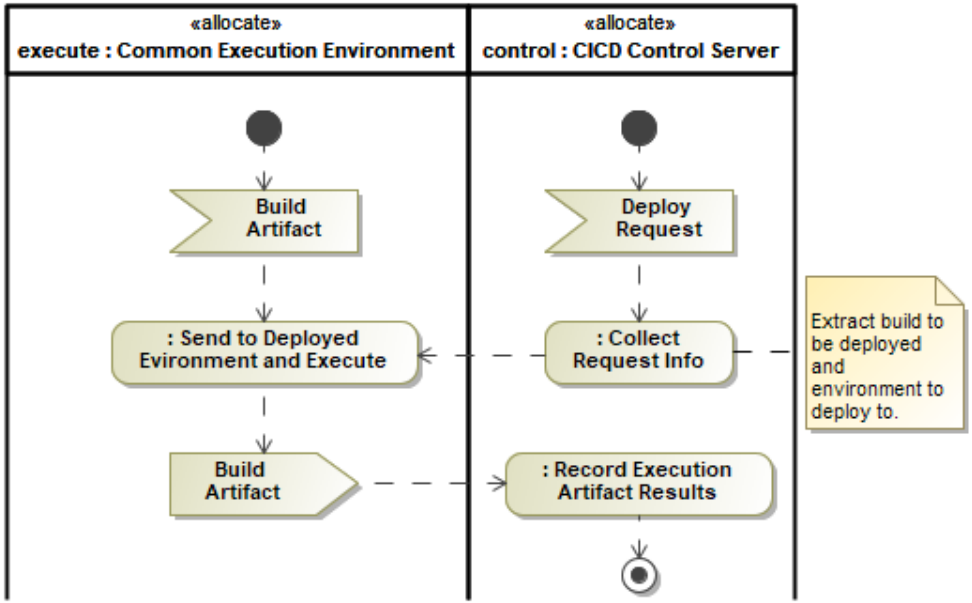
- Software Examples
  - Build SW, check for build errors
  - Run Unit Test, check for logic errors
- MBSE Examples
  - Run simulation
  - Run automatic validation checks
- Hybrid Example (Model + Code)
  - Build code that interacts with model
  - Check against test model



Deploy to Internal Active Environment

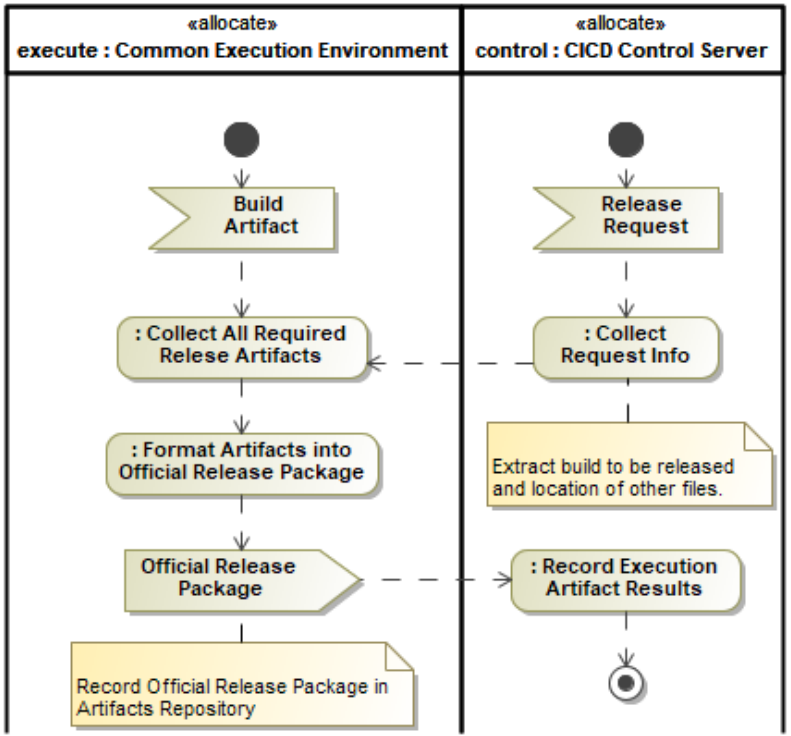


- Makes the latest changes accessible to the whole team
- In software, this may be pushing latest code on the develop branch to the dev runtime environment
- In MBSE, this may be publishing to Cameo Collaborator



Generate Release Package

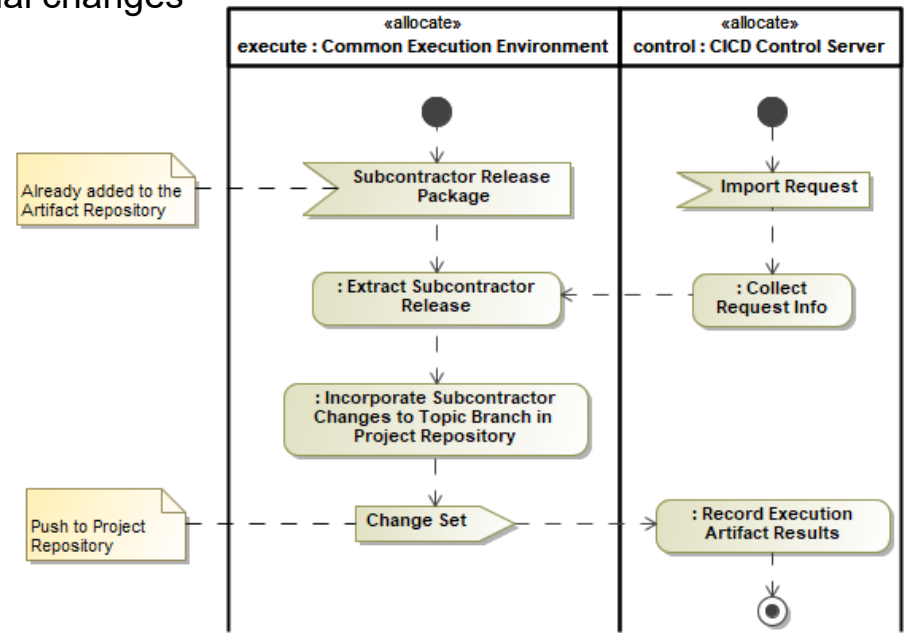
- Collect from project repository and other artifacts to create a consistent release package that is traceable to the commit history, issue history, tests, etc. and accessible via standard artifact repository
- Generate documentation for release or review
  - Extract diagrams and documentation from the model
  - Create list of change sets
  - Create list of closed issues
  - Generate the diff between two versions
- Alternate:
  - Generate metrics
  - Extract detailed custom information via scripts



Import External Release into Internal Project/Source Repository

Process for receiving and accepting external changes

- External suppliers may not have direct access to the Project Repository.
- This process is to integrate the release into a change set that is part of the Project Repository that can be handled like any other change.
- Process may also run automated acceptance tests including RoC.

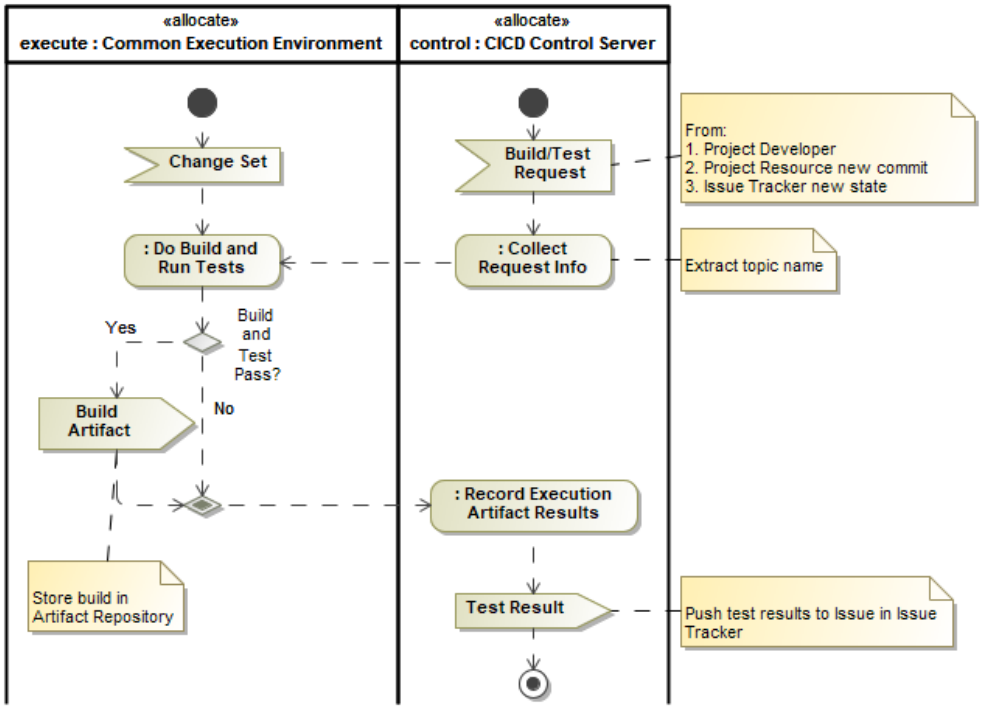


“Hello World” Example of CI/CD for MBSE

“Hello World” Example Overview

Doing a basic test of a model

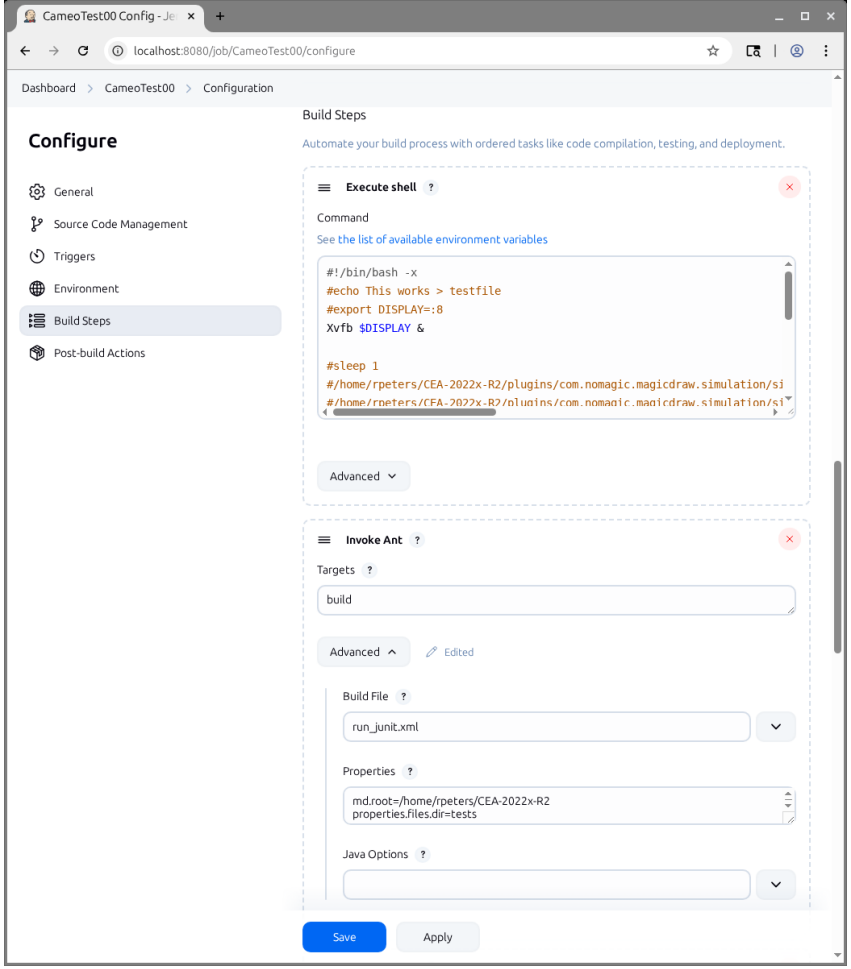
- Follows the behavior for use case “General Build Project/Source and Unit/Validation Testing”
- This test will use Jenkins as the CI/CD control server to kick off a Cameo simulation test from the command line and return the results to Jenkins



Jenkins Project Configuration 1/2

Shows the unique configuration

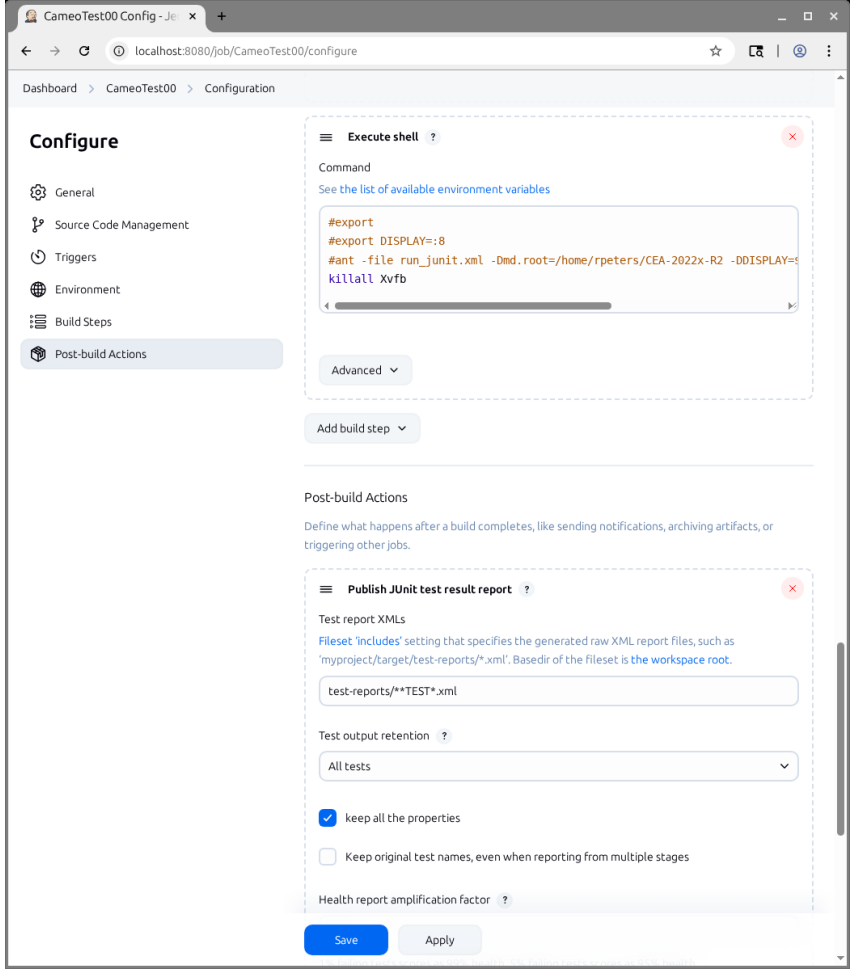
- Xvfb is needed to run Cameo headless (in the background)
  - Xvfb: X virtual framebuffer
- The DISPLAY environment variable is configured on the Node
- The Ant scripts are on the local filesystem, but they would eventually be in git (or similar text-based project repository)



Jenkins Project Configuration 2/2

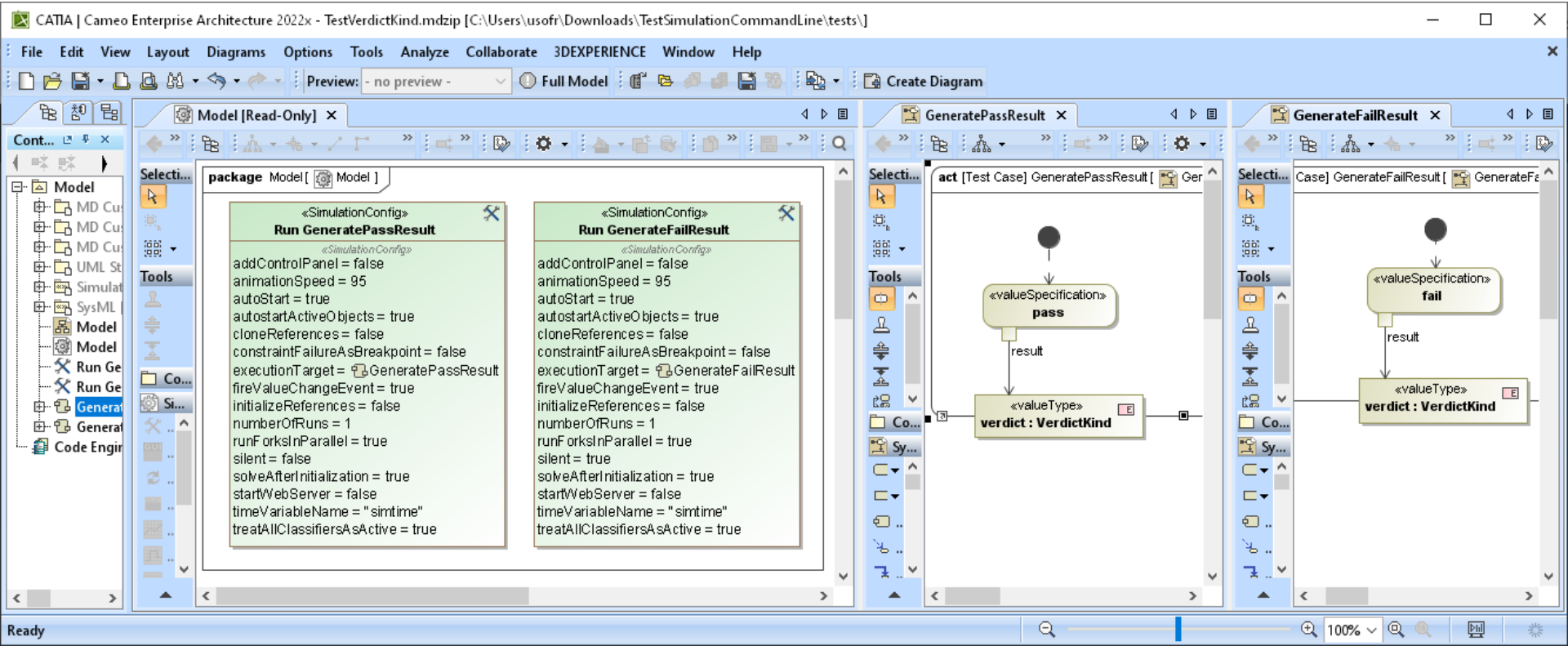
Shows the unique configuration

- Kill Xvfb to cleanup for next time
- Collect the test results as build artifacts



Example Cameo Project

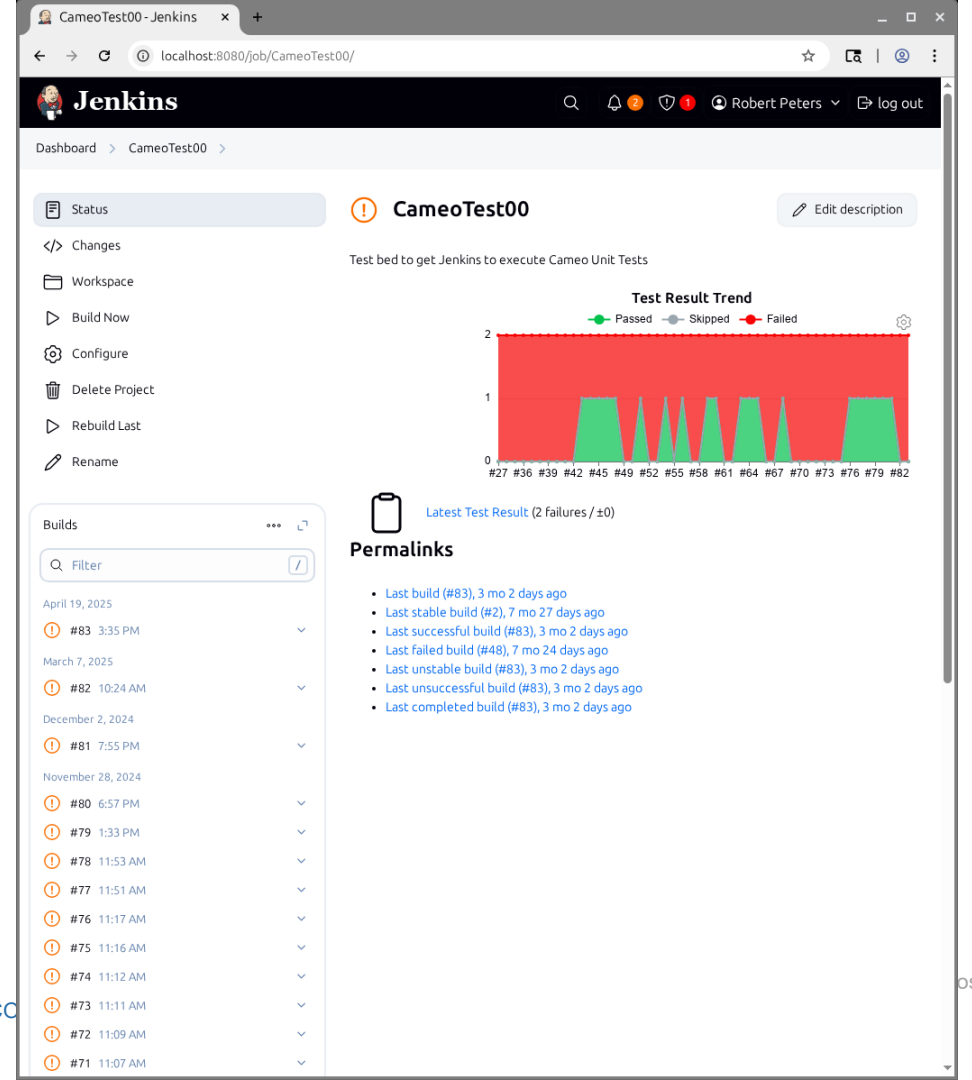
The Cameo Simulation Toolkit example model.



Jenkins Project Results 1/5

Test Result History

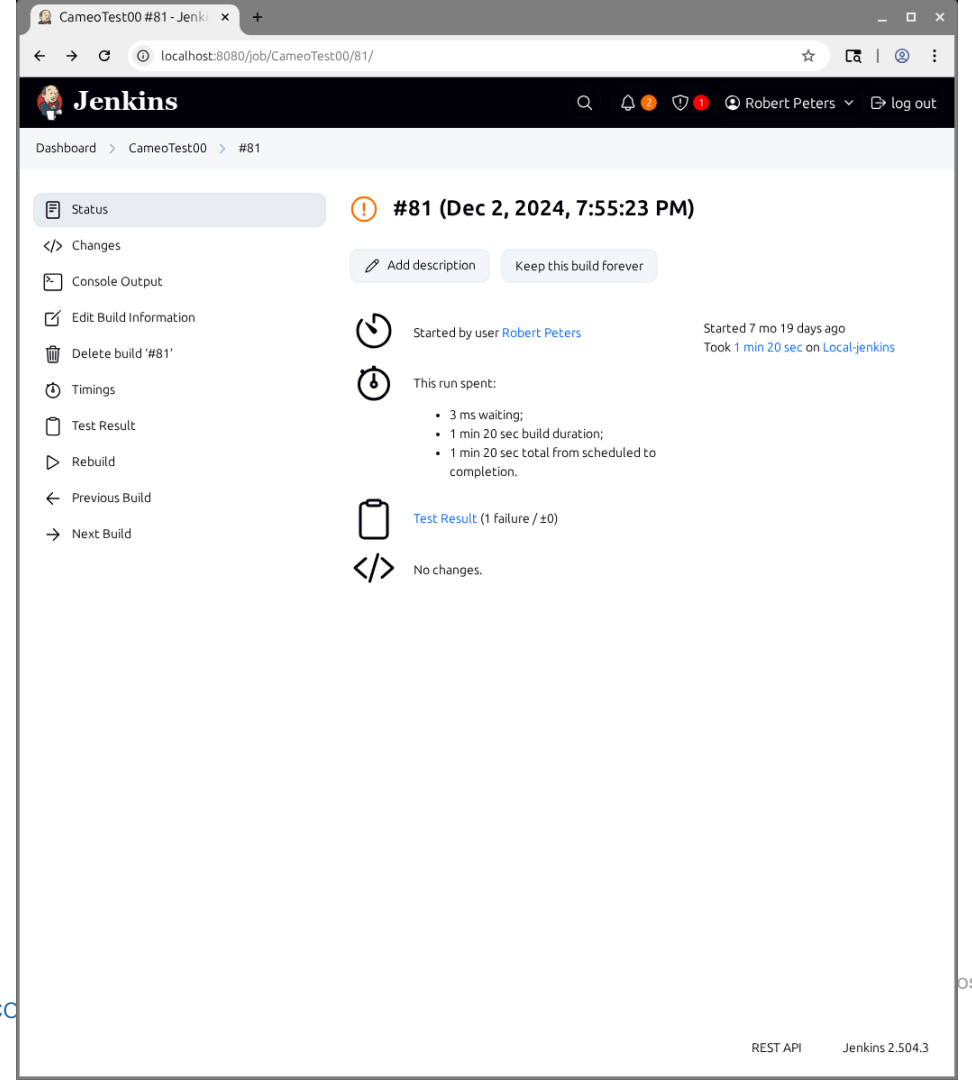
- When the build is running correctly, 1 will pass and 1 will fail.
- 2 fails is when the build isn't running correctly



Jenkins Project Results 2/5

Single build result

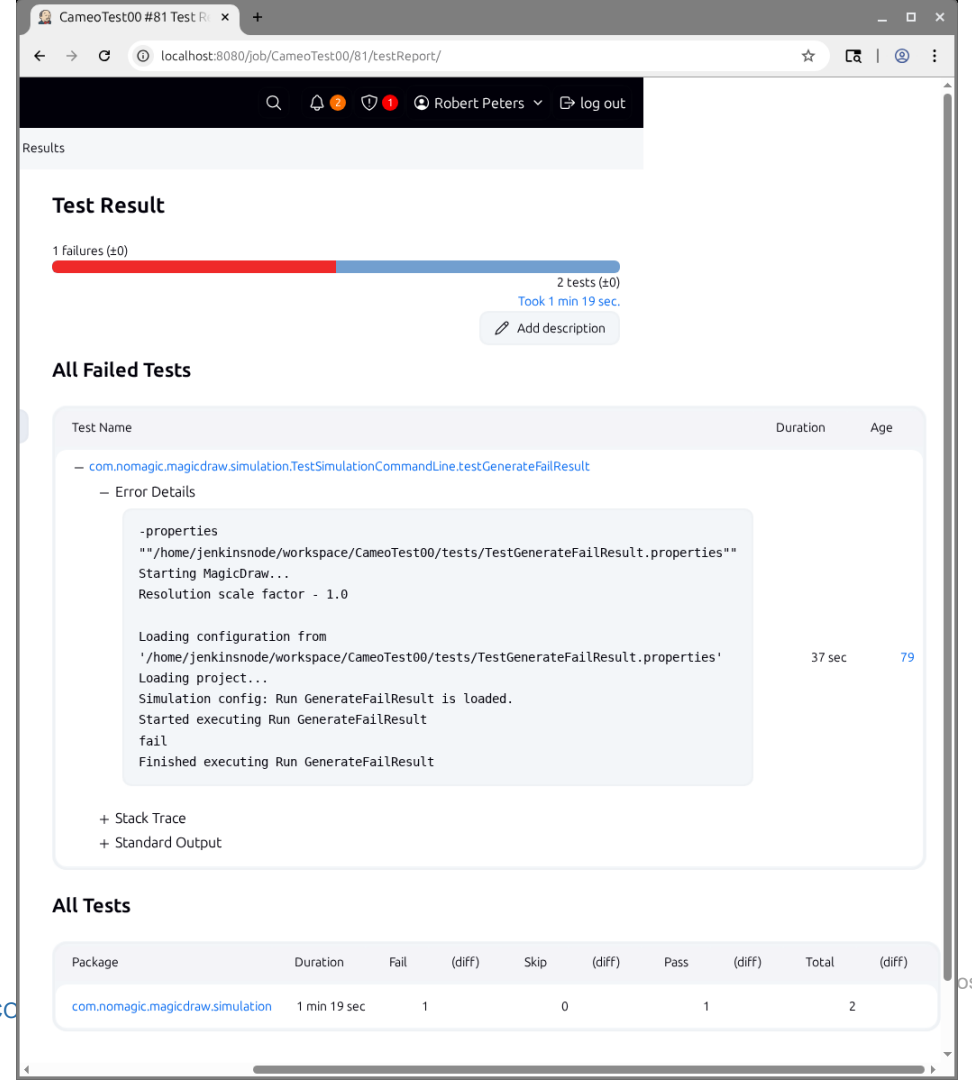
- Shows build time
- Has link to Test Result



Jenkins Project Results 3/5

Test Result Details

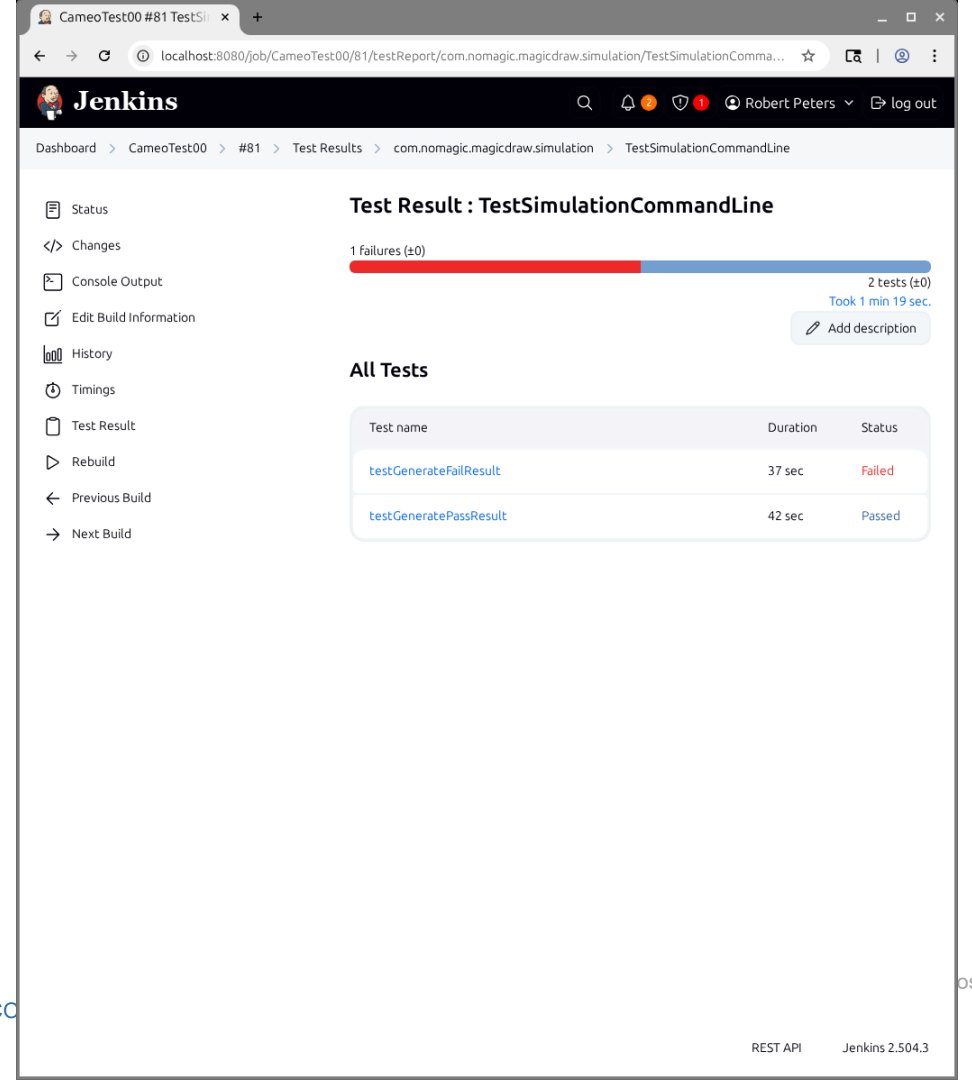
- Expandable details about
  - Error Details
  - Stack Trace
  - Standard Output



Jenkins Project Results 4/5

All Tests

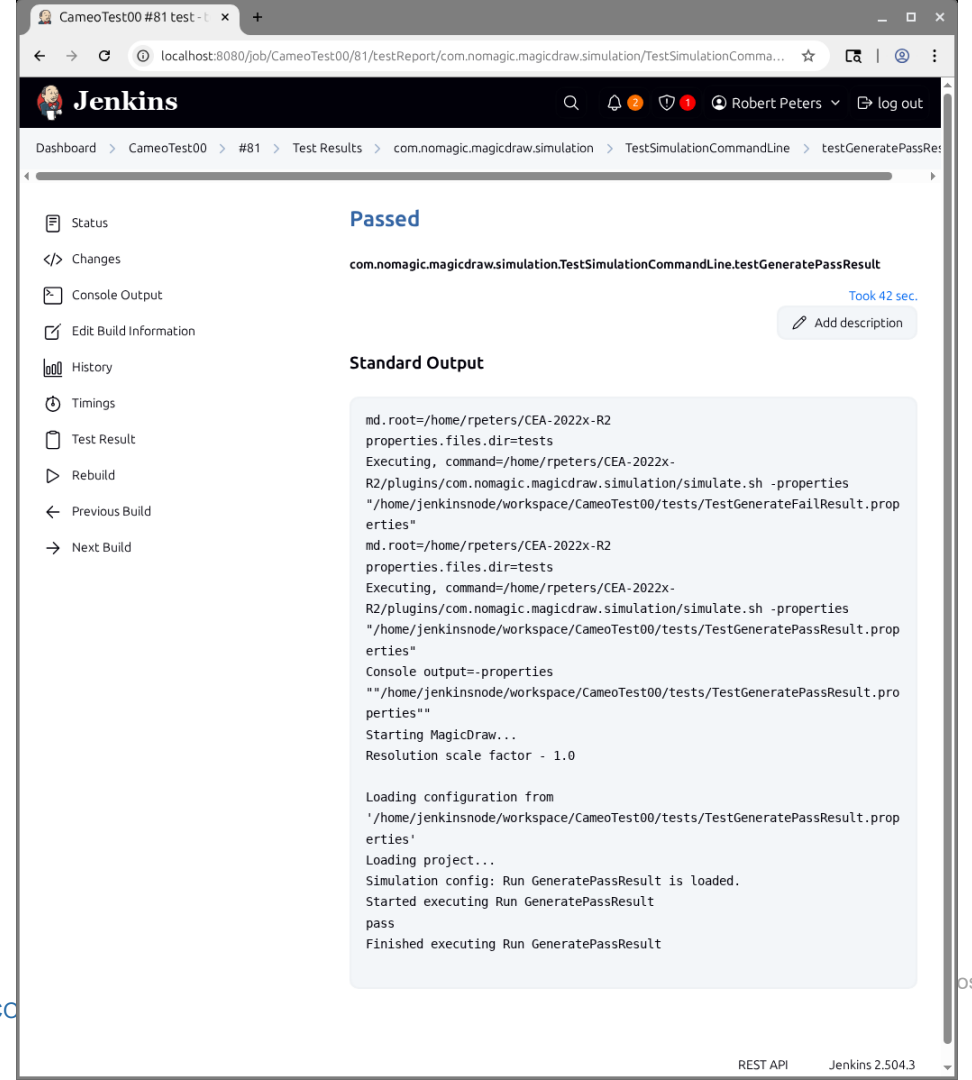
- See links for the two tests
- Time Duration
- Pass/Fail Status



Jenkins Project Results 5/5

Pass Test

- Standard Output
- Note the “pass” on the line second from the bottom



Conclusion

Conclusion - CI/CD to MBSE with Example

Comparing future MBSE goals to our “Hello World” example.

Topic	Future MBSE	What We Showed
Issue Tracking	Improved traceability	NA
Version Control	Issue-based branching	NA
Test-Driven Development	Create model with testing in mind	Check
Automated Testing	Expanded testing, blocks merge	Check
Integrated Artifact Repository	MBSE artifacts are stored with strong traceability	Partial (Jenkins)

Conclusion

There are a lot of opportunities, but still some challenges to implementing this in your environment.

- Implementing CI/CD offers the ability to manage multiple concurrent development lifecycles across a team of developers/modelers.
- Test cases are a powerful tool for enabling quality and conformance in software and are now being realized for MBSE.
- MBSE has broader use cases. The solutions for each program using models may be different and hard to reuse as a community but should be reusable within a single company.
- The current supported mechanisms for executing MagicDraw from the command line are limited, particularly when using a CAC for login.
- SysML v2 textual notation will likely bring CI/CD for MBSE closer to software.

Scalability

Quality & Conformance

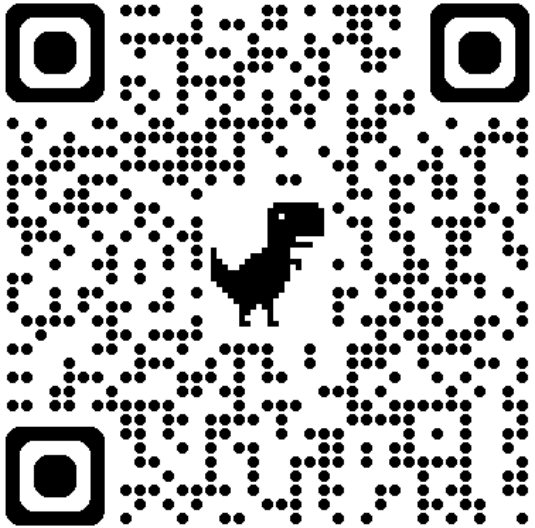
Highly Configurable

Technical Challenges

Interdisciplinary Convergence

Summary

CI/CD is great! Let's all start using it to automate repeated quality checks.



← QR Code to this presentation and additional materials (Will be available next week.)