DIGITAL ENGINEERING SUMMIT

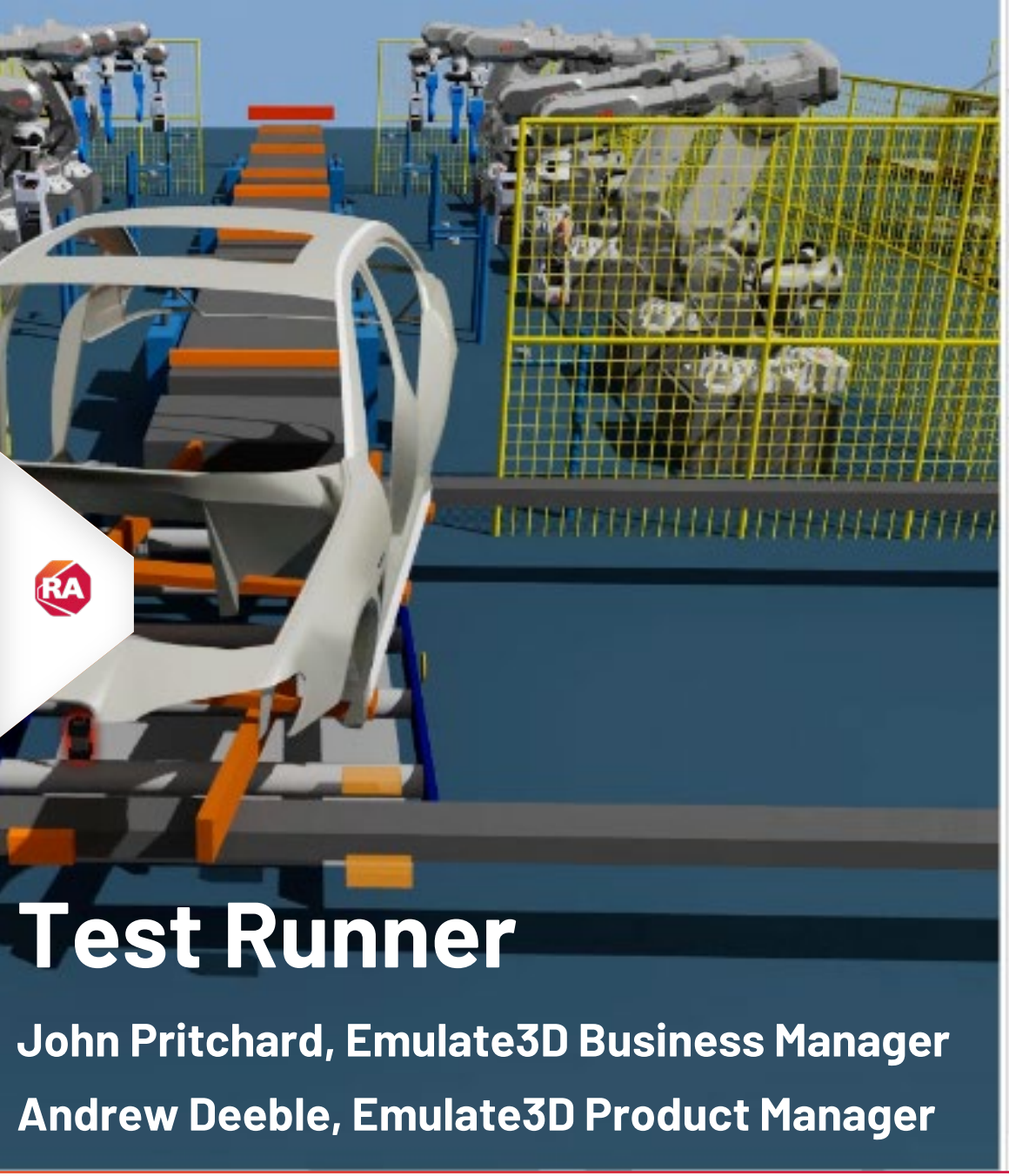
16-17 June 2025

Rome Marriott Park Hotel, Italy

EMULATE3D by ROCKWELL AUTOMATION

ROK LIVE EMEA

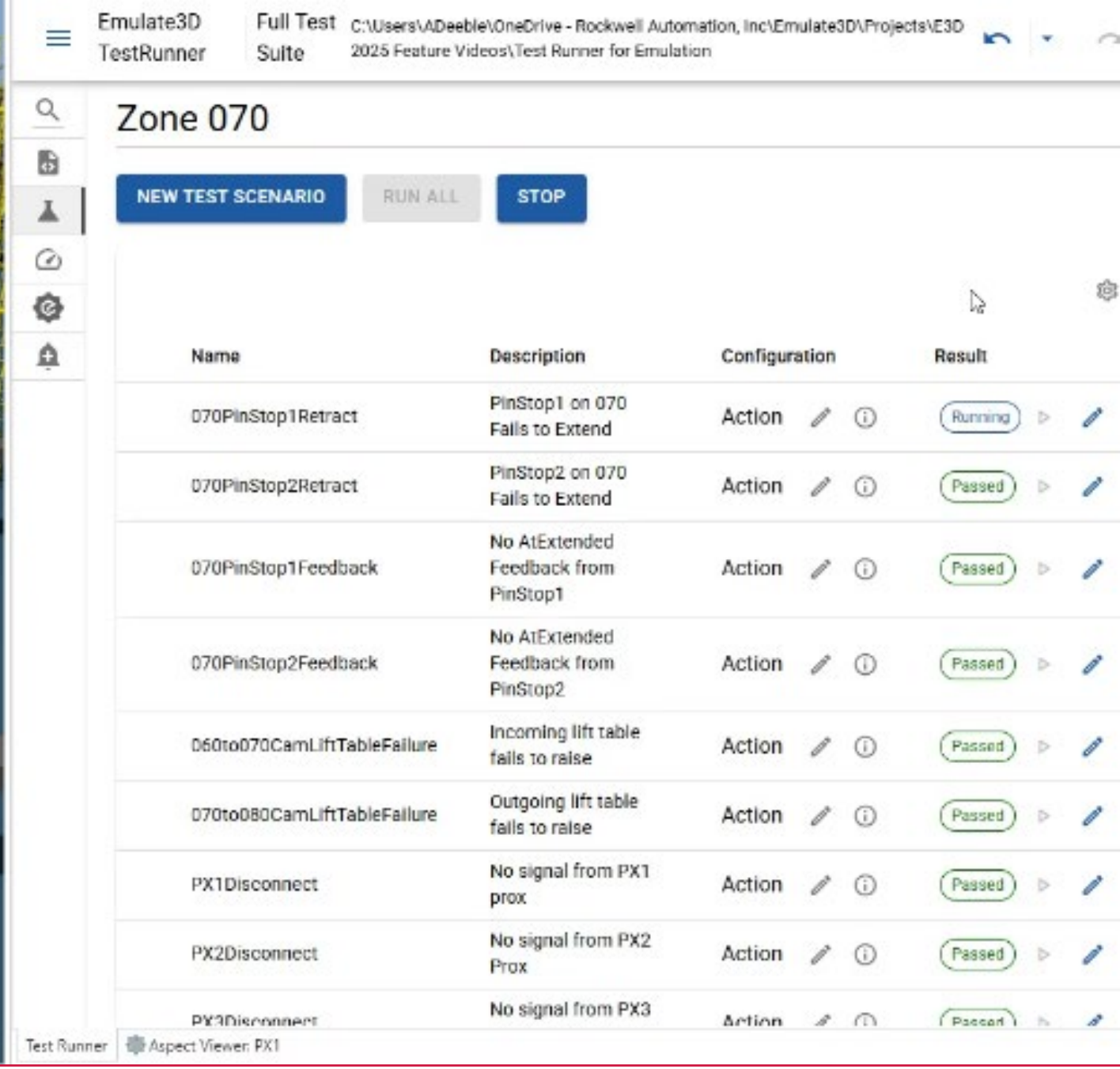
Creating the Future of INDUSTRIAL OPERATIONS



Test Runner

John Pritchard, Emulate3D Business Manager

Andrew Deeble, Emulate3D Product Manager



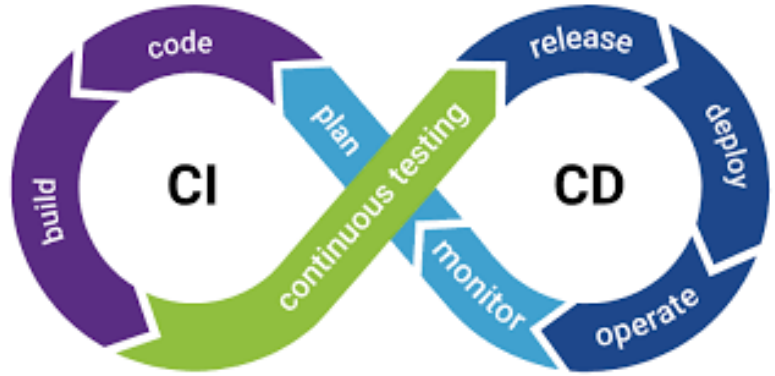
What is Test Runner?

Built on top of NUnit Testing Framework

User Interface which generates NUnit Tests
...or create your own scripted tests

Application runs externally, launching Emulate3D from the command line
...or use Test Runner inside Emulate3D

Enables powerful CI/CD workflows!



Why use Test Runner

Single test framework for Simulation, Controls Testing, and Catalog Development

Simulation	Emulation	Catalog Development
Compare models and layouts against different scenarios	Create and automatically run emulation regression tests	Create and automatically run script and catalog unit tests
Metaheuristic Algorithms for model optimization	Run test scenarios with each controls code change	Run test scenarios for each commit… more on that later!
Compatibility with existing data collection methods	Perform fault simulation with fault framework	Make full use of NUnit through scripting tests
Experiments on steroids!	**CT Regression Test Dashboard!**	**Facilitates CI/CD workflows!**

Example - Test Runner for Simulation

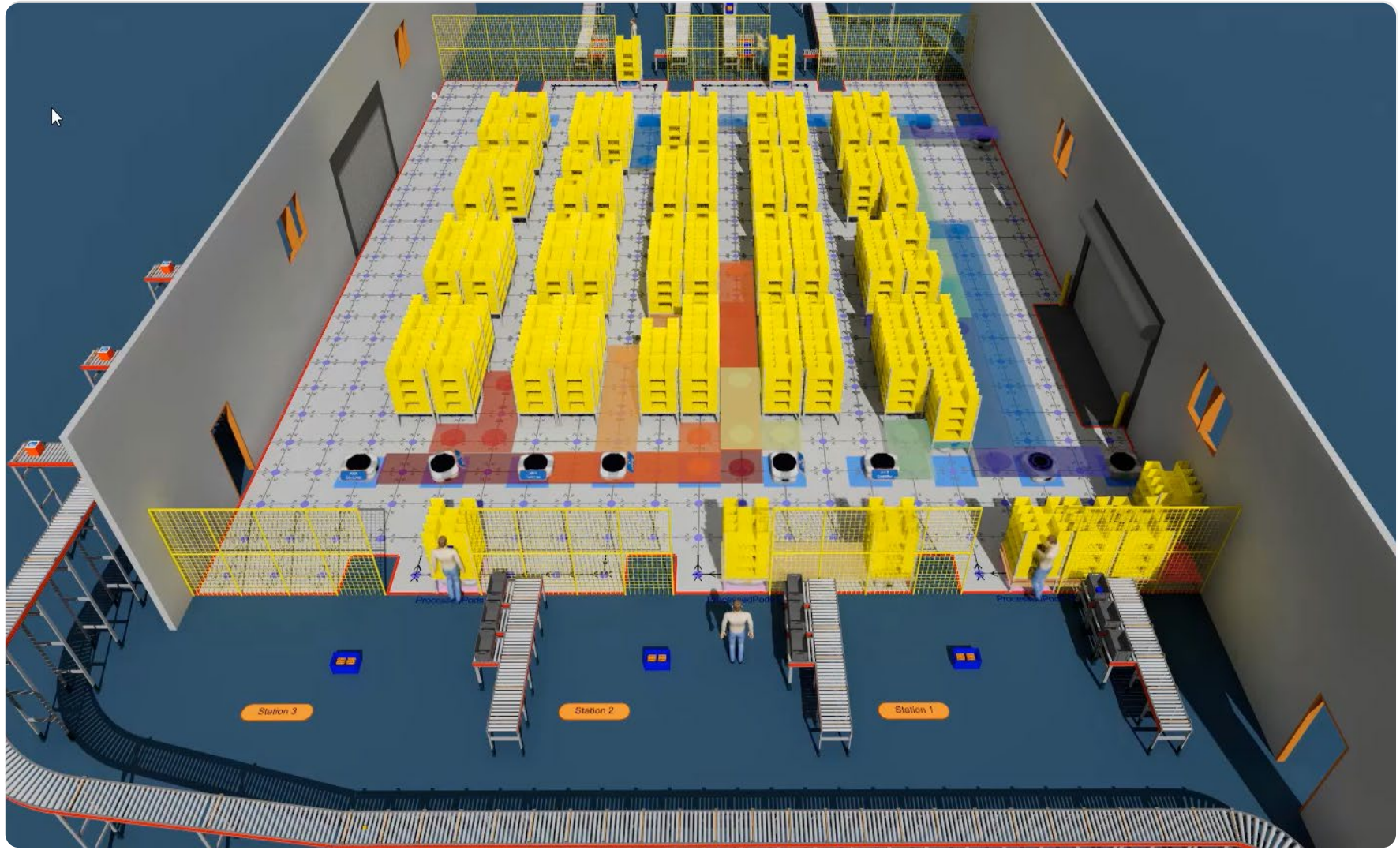
Test Runner as a successor to the Experiments feature

Use Test Runner to:

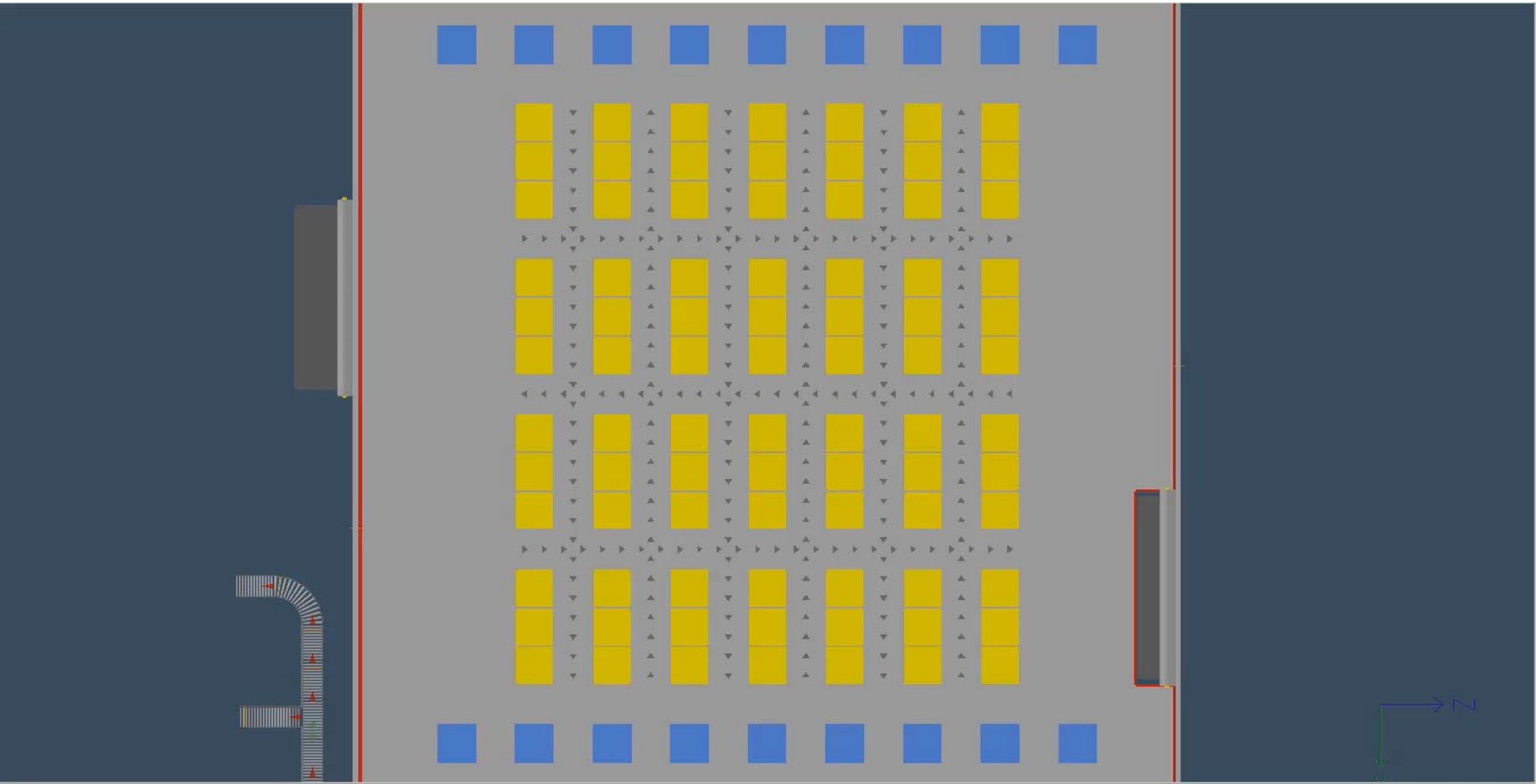
-Compare designs
such as different pod and aisle arrangements

-Run “What If” scenarios,
such as a Picking Station going down

-Vary model parameters to optimize a system,
such as AMR fleet size



Test Scenarios – Comparing Model Variants



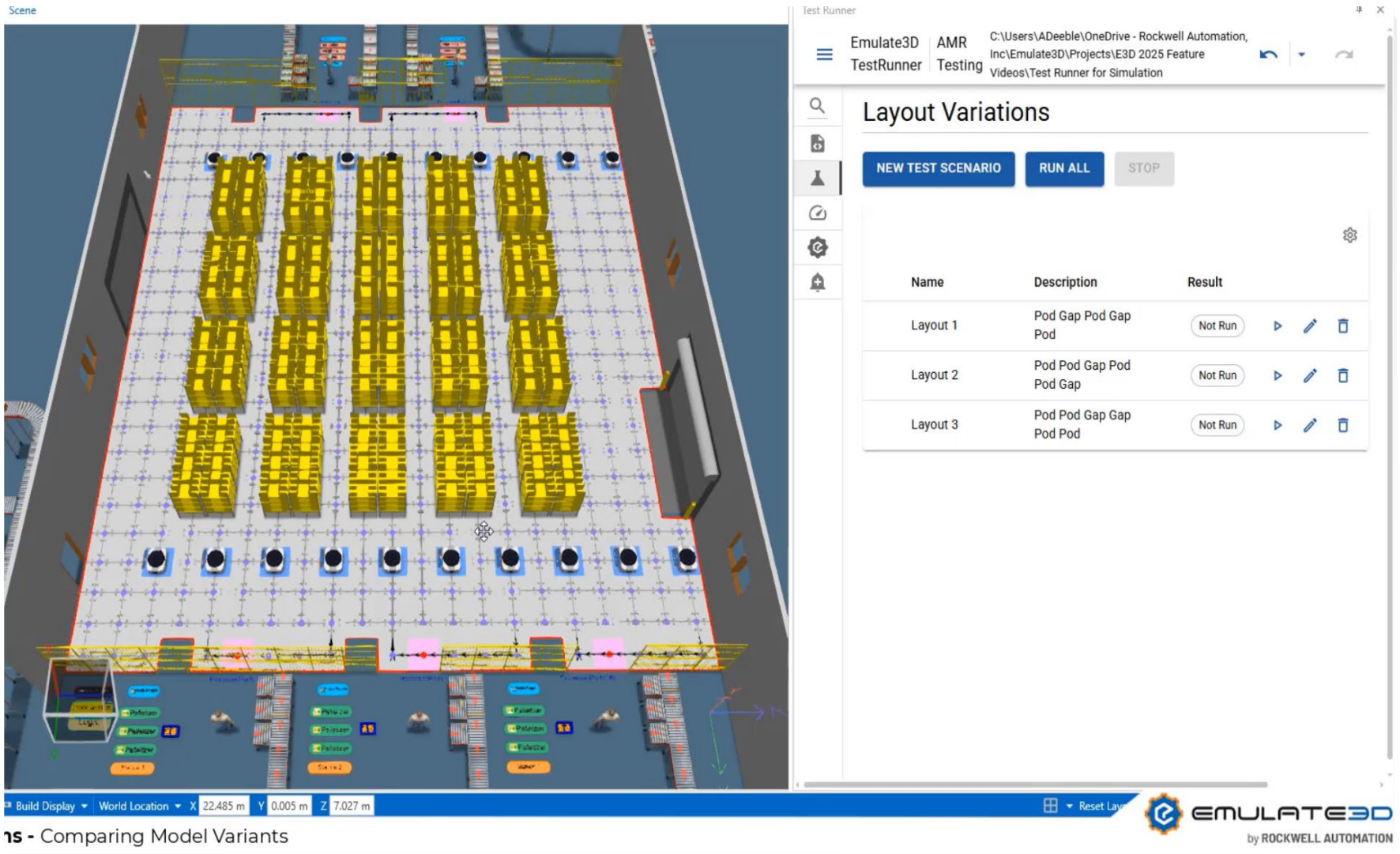
Creating Scenarios

Run a **Model** against **Actions**, and report on the results

Test Runner can be opened inside E3D, or as a separate application.

A Test Project can contain multiple models. These can be run against different scenarios.

Full data is collected in results folders.
Summary KPIs can be seen inside Test Runner



What's Planned – Replications and Reporting

Test Runner is being updated in each minor release, with a roadmap here.

Easily create replications for each scenario.

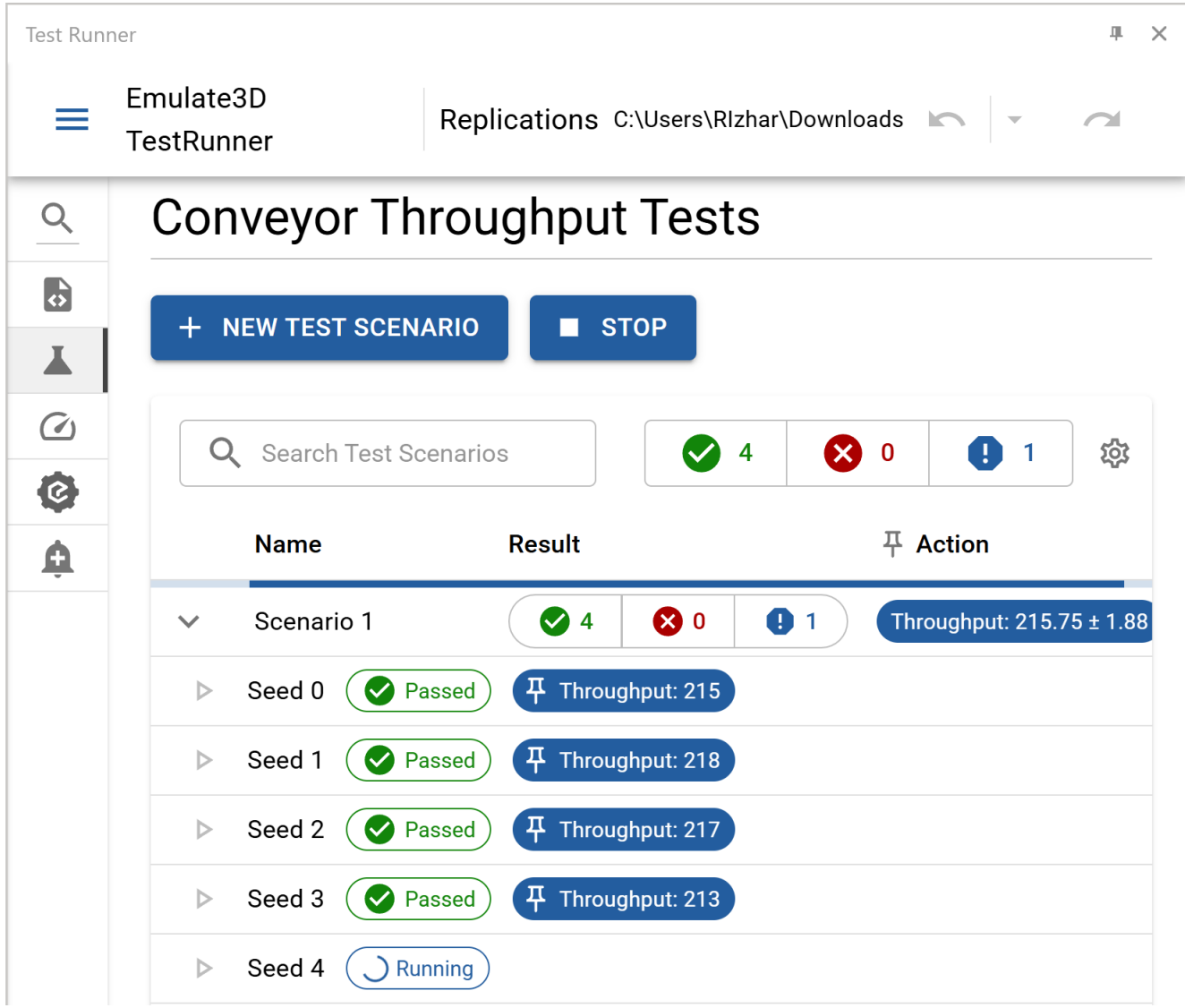
Run a test multiple times with different random number generation.

Confidence intervals presented for KPIs.

Show the pinned KPI in the dashboard view.

Improved summary data outputted as csv.

Filtering and sorting of test results.



Creating Actions

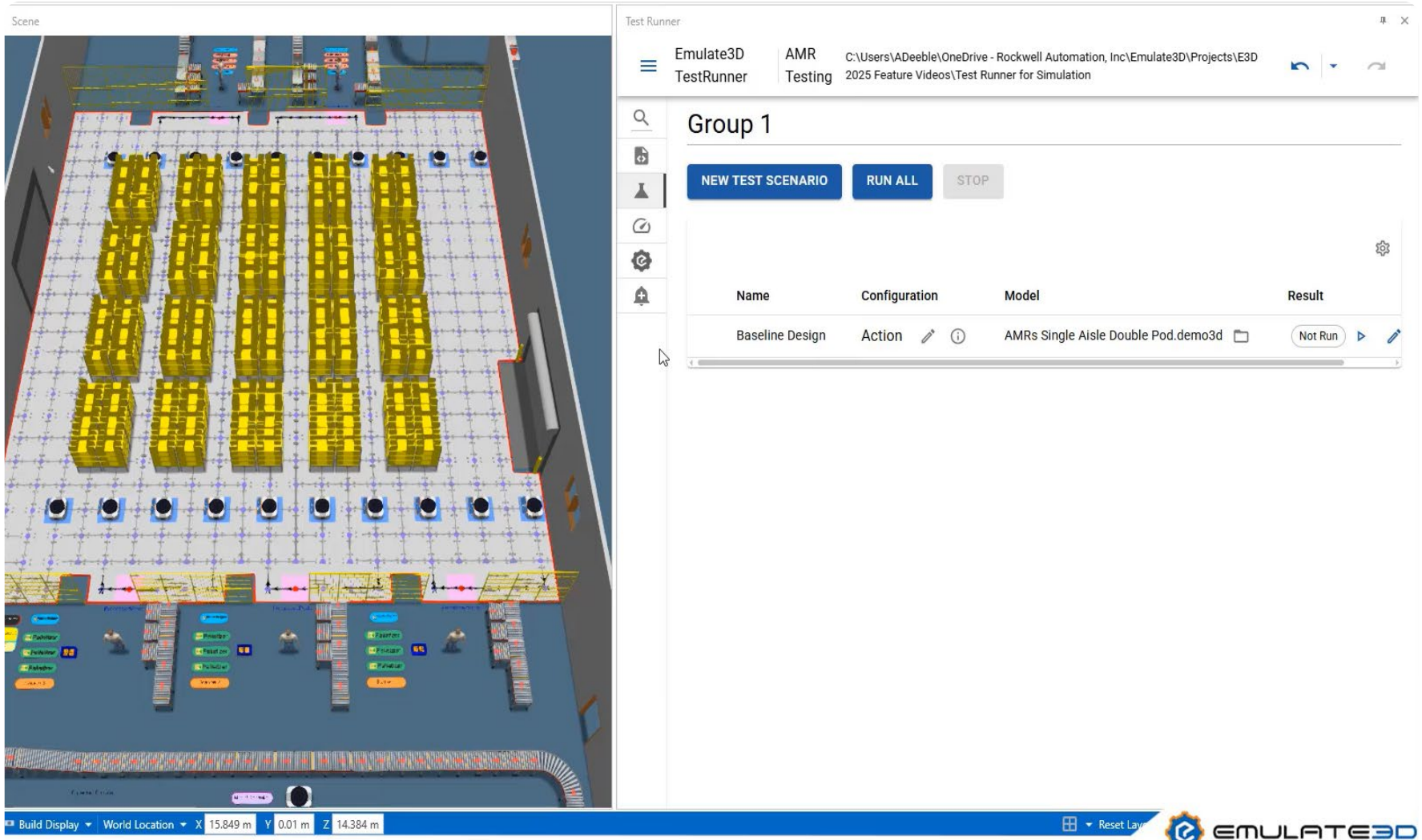
Create complex and flexible “What If” scenarios

Action schedules can be applied to a Scenario:

-Property setters change any property, aspect, or enable Fauts.

-Property recorders capture headline KPIs, alongside data collectors

-Assertions check model behaviour, and pass/fail.



ıs - Creating a Scenario

| Tips and Tricks – Generating Many Tests

Test Runner is being updated in each minor release, with a roadmap here.

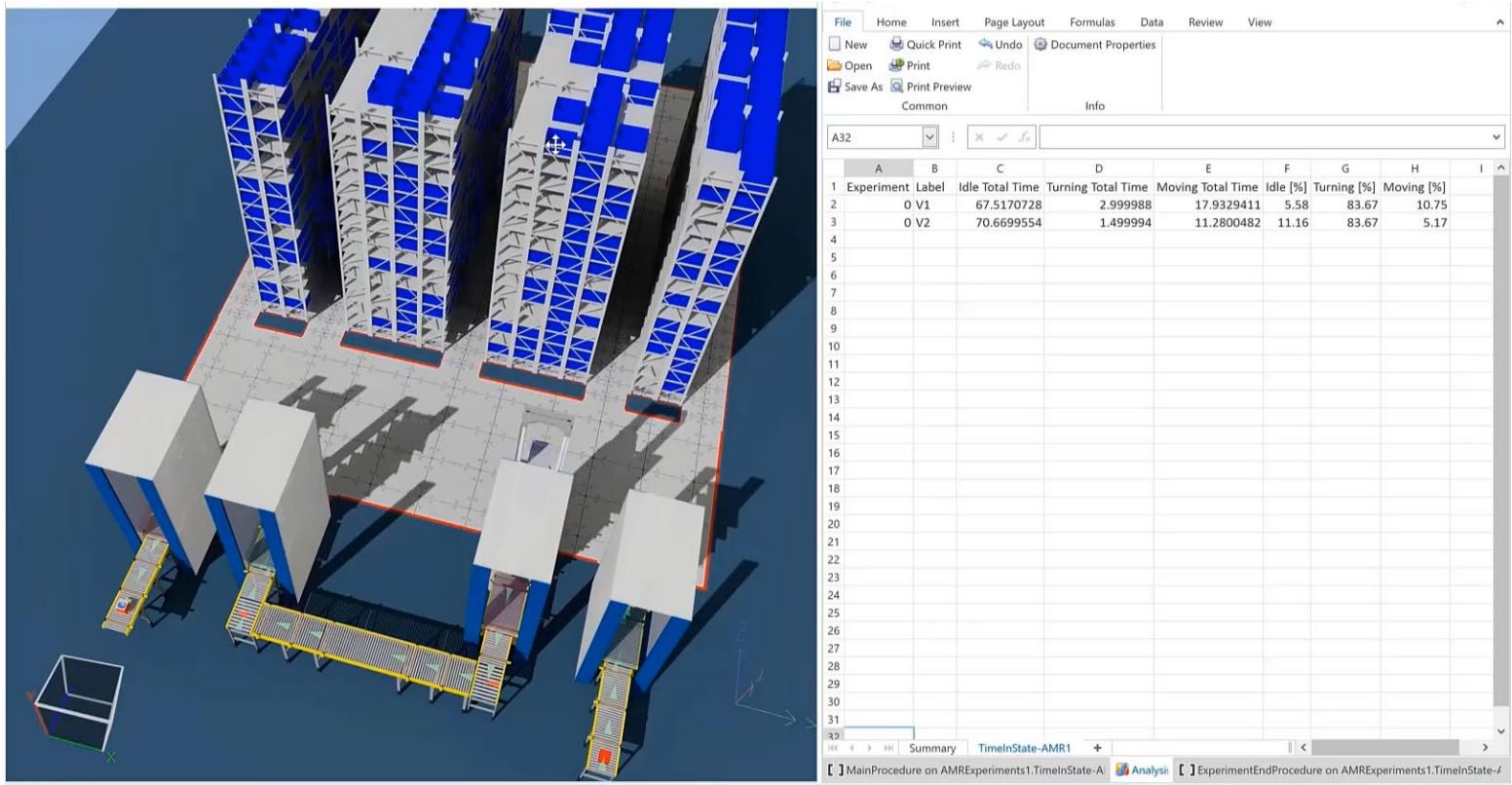
For example, sizing a fleet of AMRs. How can we run a test for 20, 21, 22, 23..... 39, 40 AMRs?

Note that the AMR framework has been updated for improved experimentation!

Use the Optimizers. The Brute Force optimizer is ideal for ranges of tests!

Use "Based On" to make one set of Actions inherit from other Actions.
Coming soon – actions triggered by events / delays from preceding actions

Customise Actions – all editable JSON
Coming soon – import actions from CSV



Model Optimization

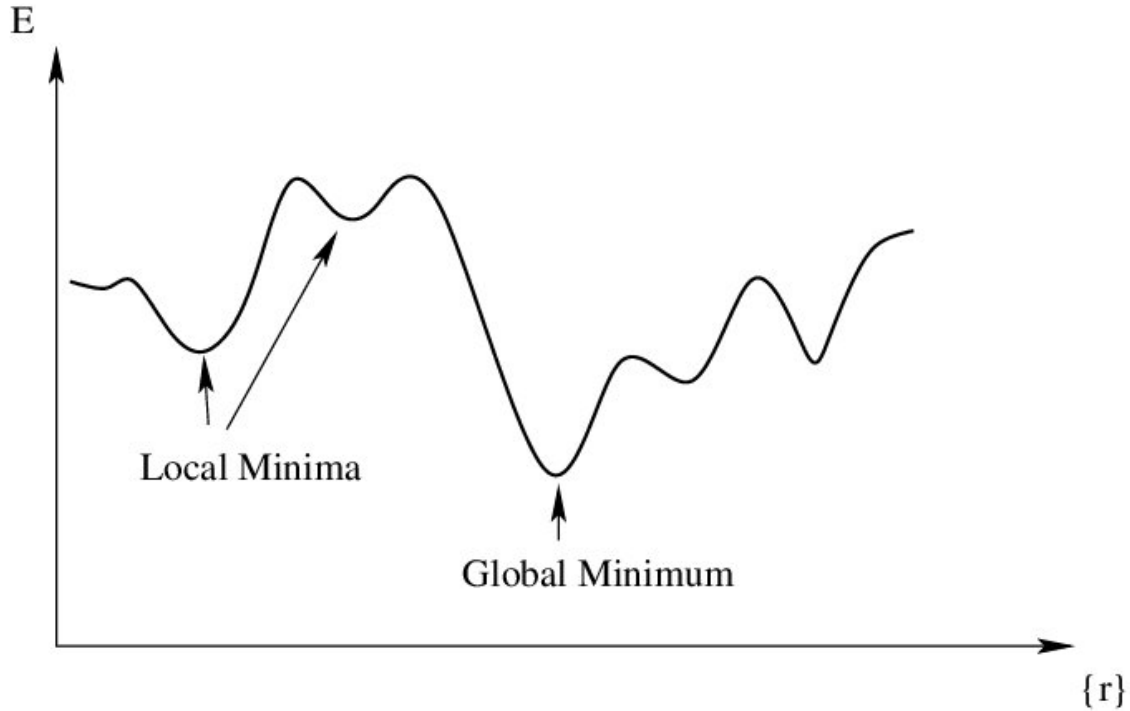
Tools to automatically vary parameters to maximize a chosen goal

Find combination of parameters that maximizes some goal(s).

Parameters are user-specified:
- Number of vehicles, size of buffers or storage…
- Any numeric property (continuous or discrete)

Goals are user-specified
- Throughput, Utilization, Operating cost
- Any function that returns a numeric value

The gradient based methods you learned at school are not enough…



Metaheuristic Algorithms Inside Test Runner

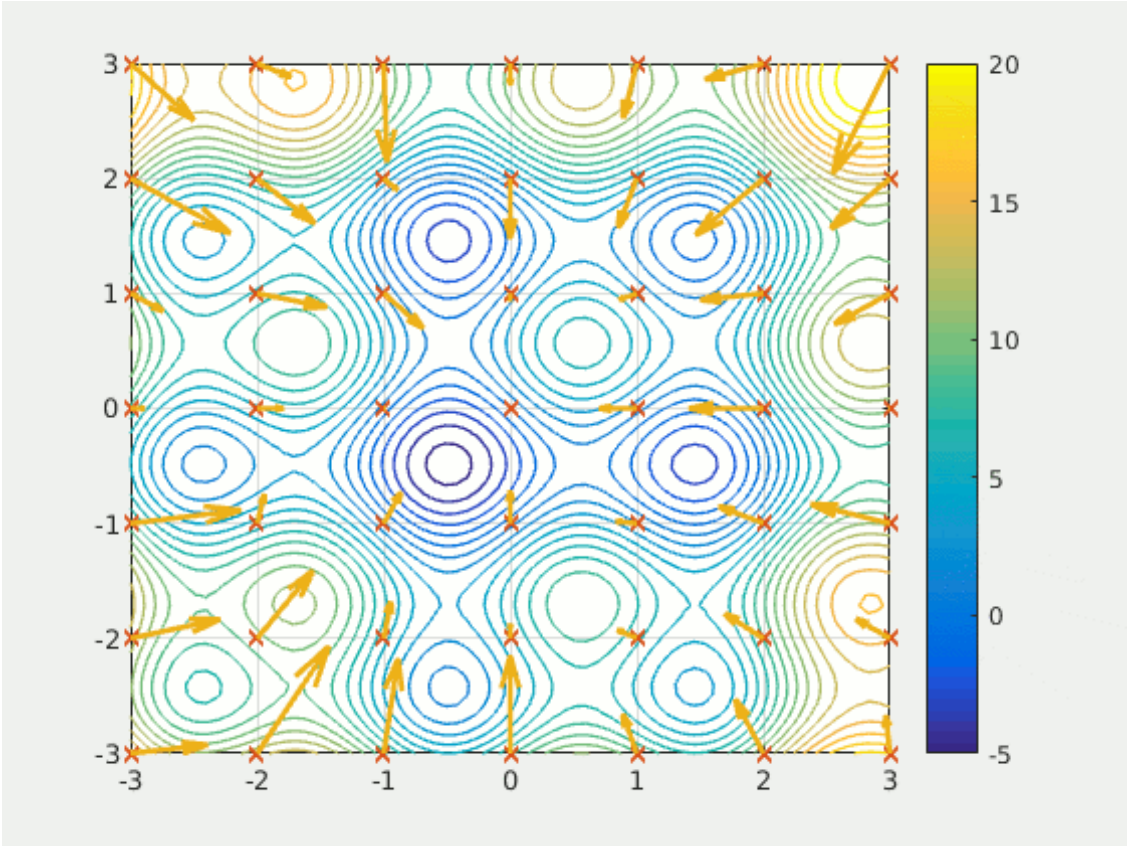
Metaheuristics are algorithmic frameworks that help find solutions to complex optimization problems that are difficult to solve with traditional methods

Major advantages for Emulate3D:
- Less likely to get trapped in a local minima
- Can be stopped at any time
- Can be easily parallelized

A balance between **Exploration** and **Exploitation**

Exploration: Venture into new areas of the search space

Exploitation: Refine promising solutions



Metaheuristic Algorithm Options in Test Runner

Many options offered out of the box, suitable for different applications

Algorithm	Description	Advantage
Particle Swarm Optimization	Particles gravitate towards their historic best position and the global best position	Strong exploitation to refine best solution
Genetic Algorithm	Evolutionary approach, the best parents produce children with random mutations	Strong exploration to find novel solutions
What's Planned – Tabu Search	*Intelligently avoids searching previously explored regions of the search space*	*Reaches a conclusion with fewer tests*

Optimizers

Automatically generate tests, learn from the results, and repeat to find an optimum solution

Specify parameters to vary and their range:
AMRs, 10 – 50,
Merge Delay, 0 – 0.4s

Specify objective function, the target goal e.g Throughput

Generates and runs tests
Reads results, generates new batch of tests
Reports results the user via dashboard

Coming Soon – live data reporting as tests run

Stages of Virtual Commissioning

Can we connect?

Simply connecting to the model finds issues:

Map IO to virtual equipment

Connect the HMIs

Initialize and reset alarms

Check safety feedback

Can we run in manual?

Begin testing system by forcing values:

Verify sensor feedback

Dry run motors

Test manual functions

Step through the sequence

Can we run in auto?

Create product and see how our system responds:

Run idealized cycle on auto

Reposition field sensors

Verify predicted throughput

Test safety, stops, restarts

Ready to go on site?

Wait there is more!

...Maximising Value from Virtual Commissioning

Can we connect?

Can we run in manual?

Can we run in auto?

Advanced testing

Testing in the virtual world is easier than in the real:

Inject device faults

Check alarms & diagnostics

Run varied load schedules

Stress test the system

Operator Training

Familiarize and train on an accurate model:

Familiarize with HMIs

Run training scenarios

Train on device failures

Grade operator responses

System Upgrades

Reuse the model to test potential changes:

Regression test code changes

Optimize performance

Test hypothetical scenarios

Reproduce issues virtually

Using Test Runner for Emulations

Automated testing of your controls code, enabling CI/CD workflows

Create a suite of test scenarios:

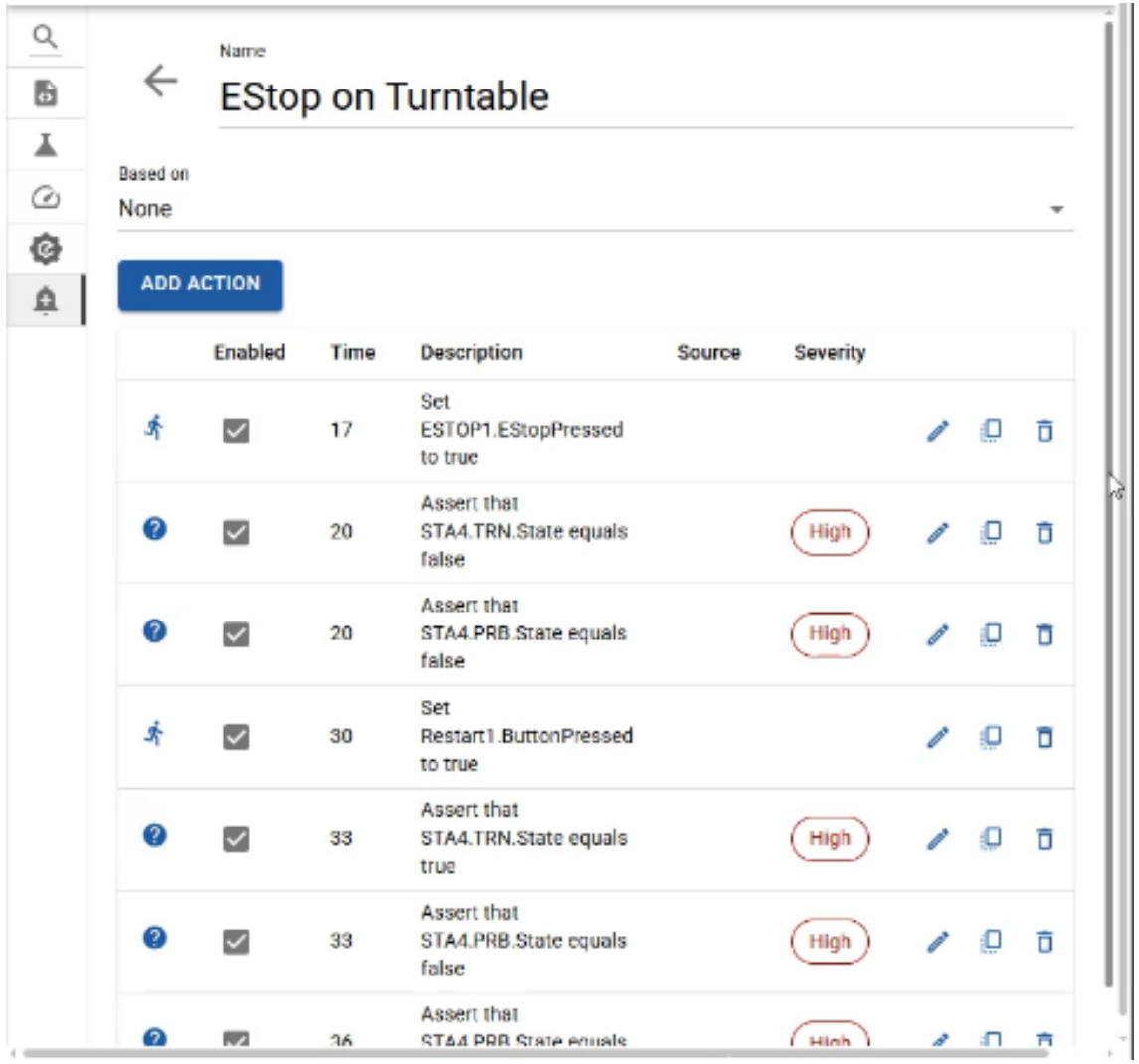
- Inject simulated device faults
- Run different product variations
- Simulate operator interactions
- Stress test the system

Define expected behaviour with Assertions:

- Add to the model or to the scenario
- Define severity level, prerequisite conditions

Run test suite from Test Runner every time you make a PLC code or design change

See a dashboard of test results



Test Runner for Emulation

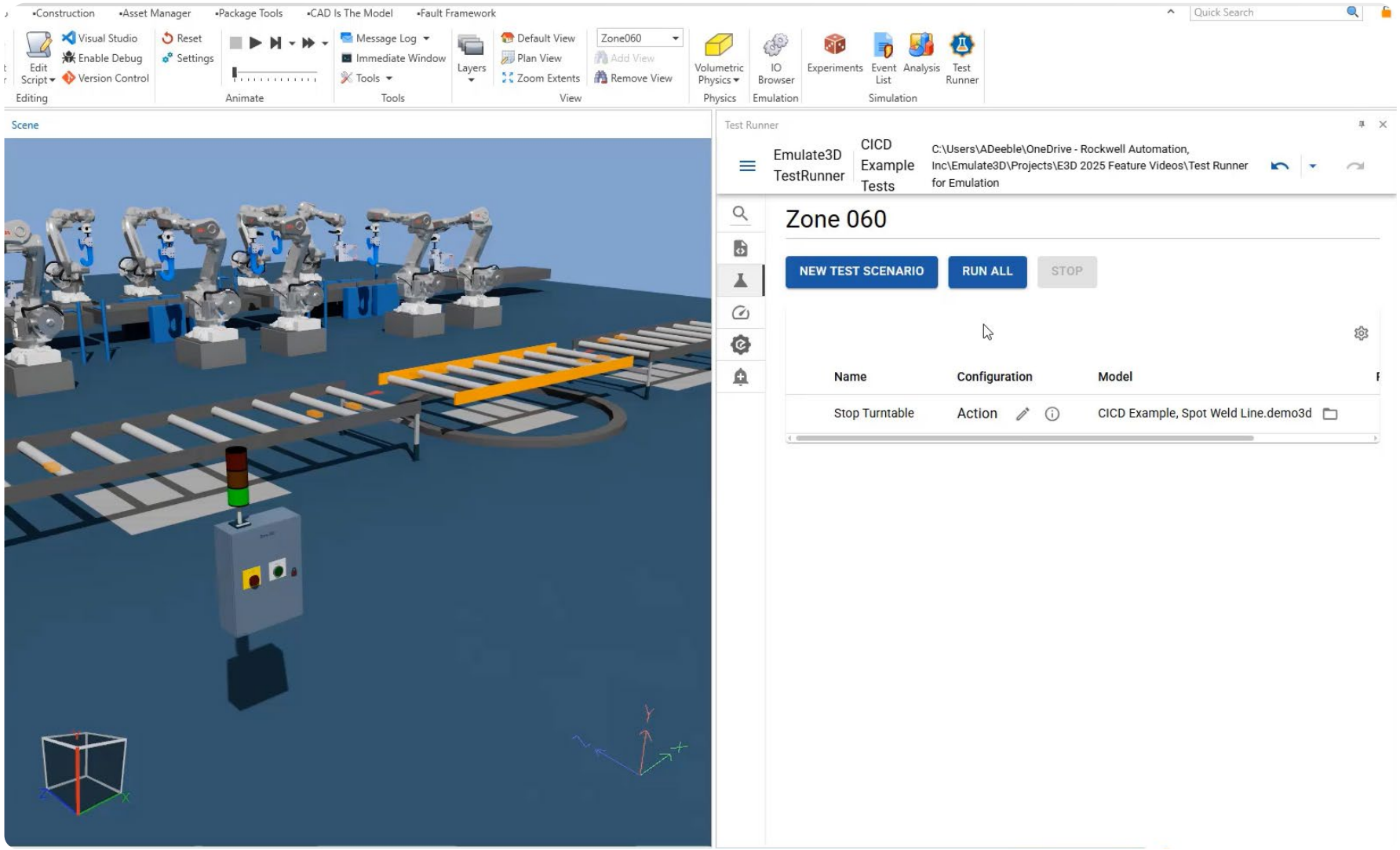
Create regression tests for your controls code

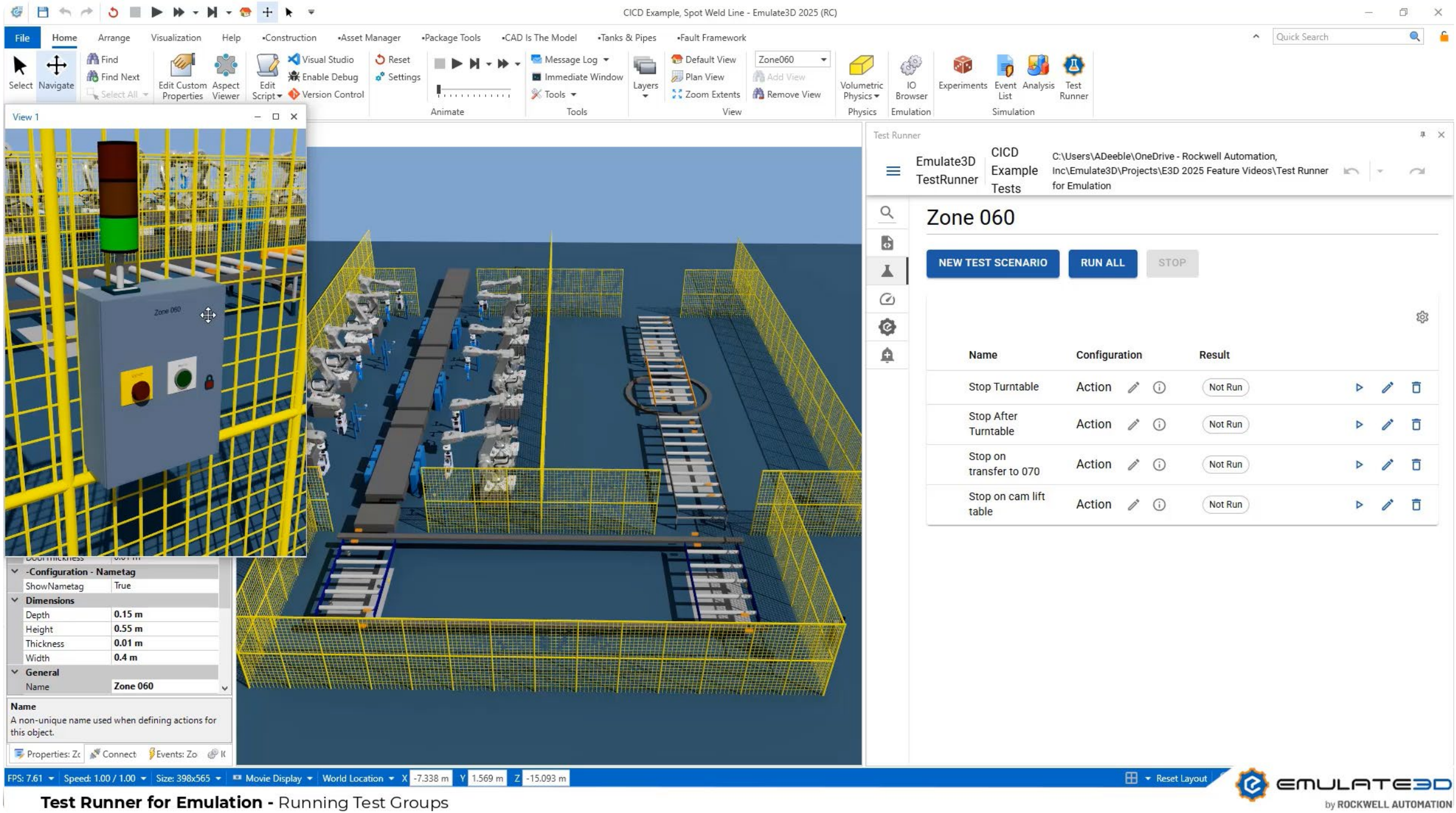
Create a set of repeatable actions which can be run against your PLC code.

Define Assertions in the test, or in the model.

Run the test suite, report on Pass/Fail/Warnings.

Tip – The Emulation State component can help each test automatically connect





Test Runner for Emulation - Running Test Groups

Test Runner with Fault Framework

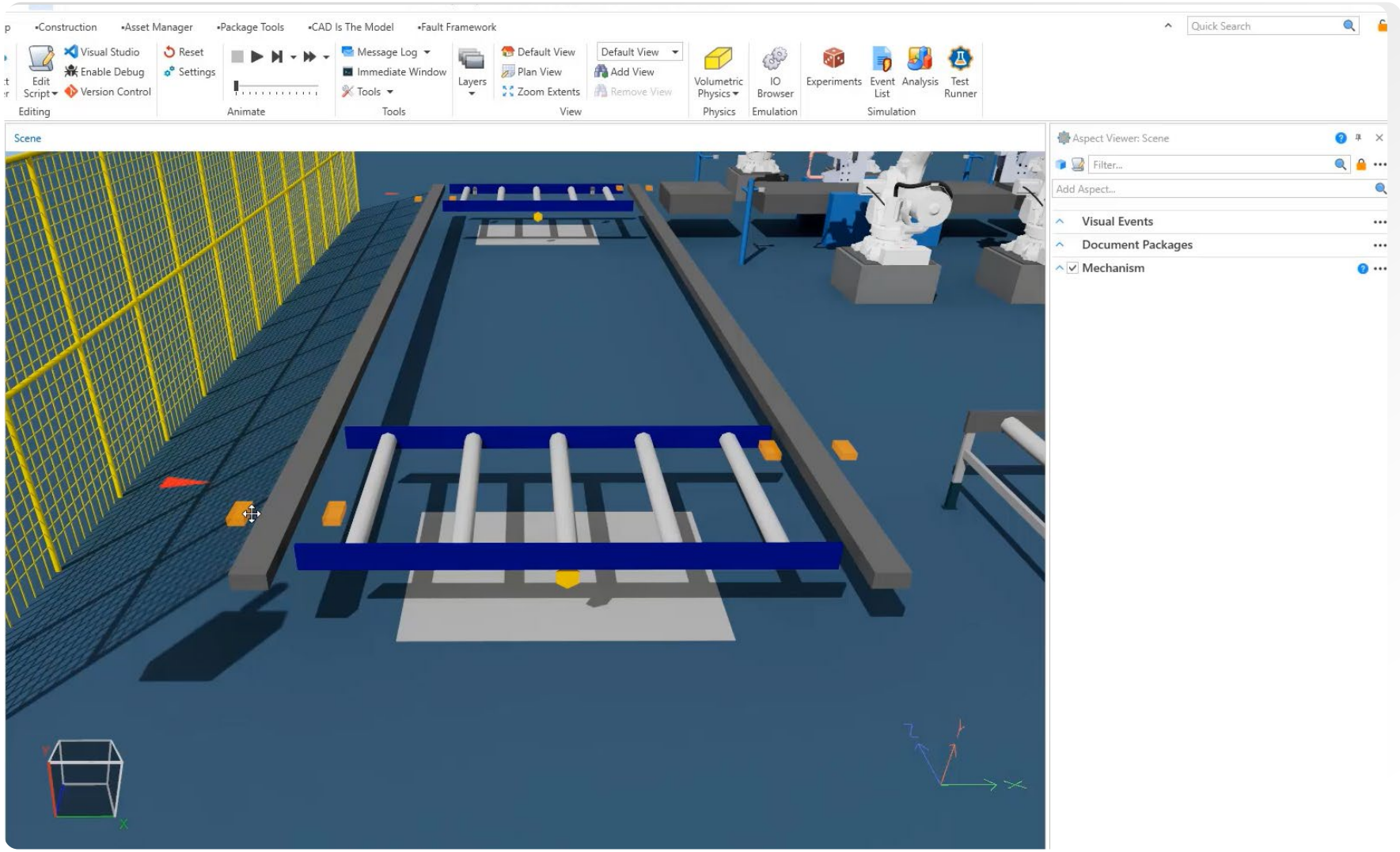
Simulate device faults using the Fault Framework aspects, from within Test Runner

Easily trigger faults from the Test Runner.

Tests adhere to Fault Framework assertions, or to test actions.

Also applicable to Operator Training

Coming Soon – Dedicated Fault action with autofill



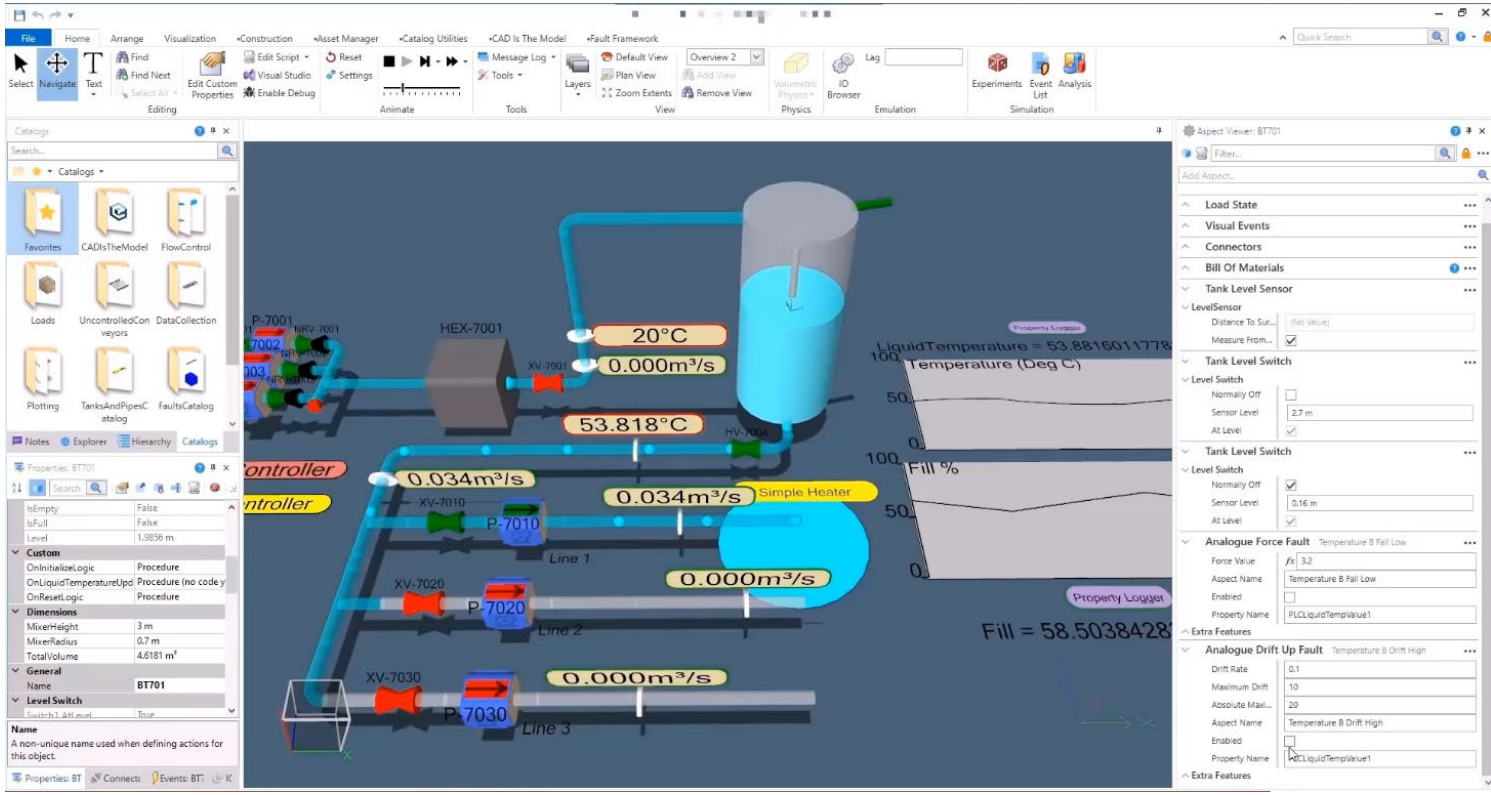
Operator Training

Fault Framework and Test Runner enables Operator Training workflows

Create operator tests with actions and device faults for operators to diagnose

Use assertions to see if the operator correctly resolved the challenge

Automatic log of all interactions, ability for operator to log notes



Test Runner is also for Catalog Development...

... but you'll hear about that in another presentation!

Use actions and assertions for simple tests

NUnit support in custom scripting enables advanced testing

CI/CD for custom catalog development:
- Version control with integrated GIT UI
- Run automated tests with Test Runner
- Deploy to custom package feed

Let's first see GIT version control in detail!

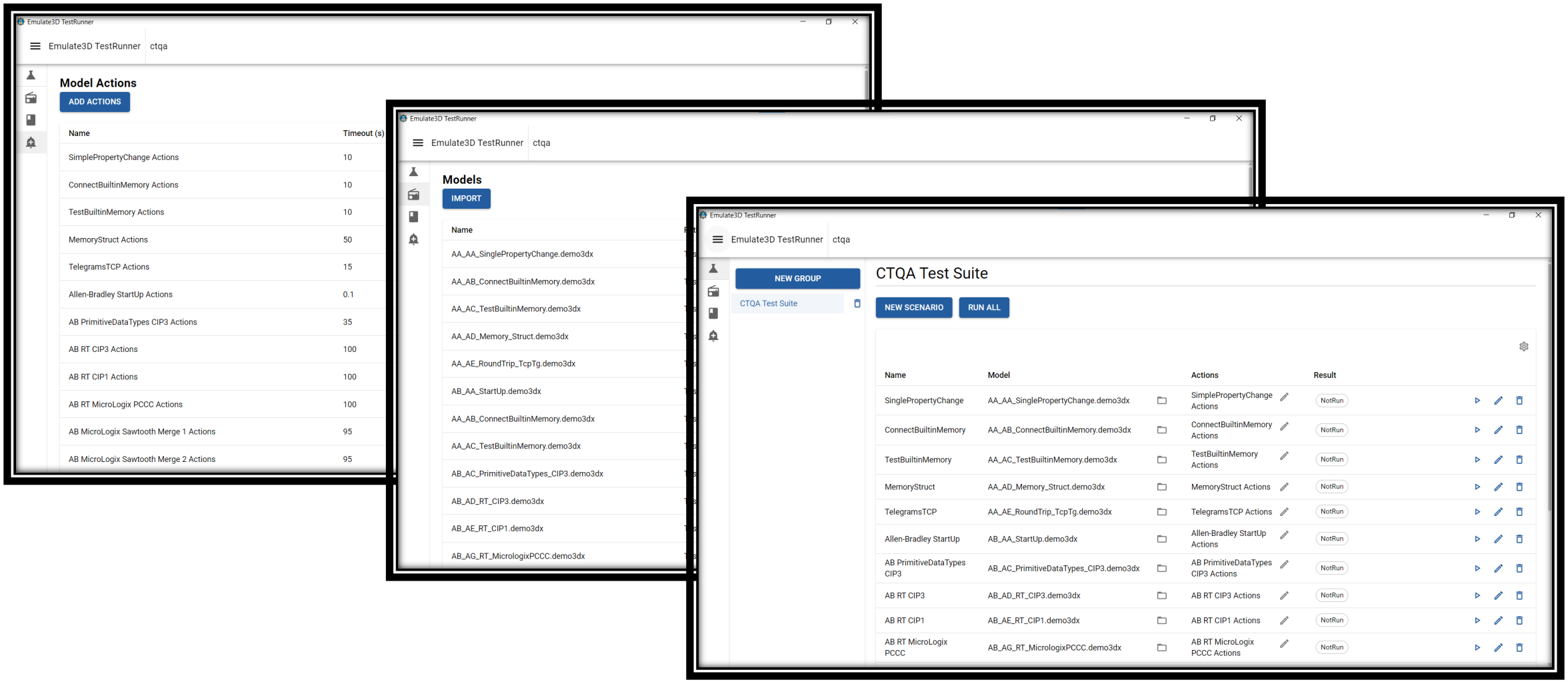
```
[TestFixture]
public class MyTests : Emulate3DTestFixture {
    [Emulate3DTest]
    public async Task Pass() {
        await Task.Delay(2000);
        Assert.Pass();
    }

    [Emulate3DTest]
    public async Task Fail() {
        await Task.Delay(2000);
        Assert.Fail();
    }

    [Emulate3DTest]
    public async Task ConveyerTest() {
        var conv = App.Document.FindVisual("SB1");
        Assert.IsNotNull(conv);
        var lc = conv.FindChild("LoadCreator1");
        Assert.IsNotNull(lc);
        if (lc.Props is not LoadCreatorProperties props) {
            Assert.Fail("Unexpected type");
            return;
        }
        props.DelayRate = 20;
        props.CongestionZone = false;
        App.Invoke(App.Reset);
        RunActionAtTime(5, () => {
            var loadCount = App.Document.PhysicsEngine.CountLoads();
            Assert.That(loadCount, Is.GreaterThanOrEqualTo(5),
                "Not enough loads in the scene, is the LoadCreator1 release rate too low?");
            Assert.That(loadCount, Is.LessThanOrEqualTo(10),
                "Too many loads in the scene, is the LoadCreator1 release rate too high?");
        });
        await RunModelForSecondsAsync(5.1, 2);
    }
}
```

Test Runner is used by the Emulate3D Developers...

...for catalog, regression, communications, and performance testing!



Rockwell Automation

expanding **human possibility**®

Thank you

www.rockwellautomation.com