

MathWorks
**AUTOMOTIVE
CONFERENCE 2023**
India

Accelerating Development for Software-Defined Vehicles Using CI/CD

Nukul Sehgal, MathWorks



Vamshi Kumbham, MathWorks



Rajat Arora, MathWorks



Software Defined Vehicles

- Automotive industry is embracing Service-Oriented Architectures (SOA) as a new paradigm to design modern applications like Software-Defined Vehicles (SDVs)

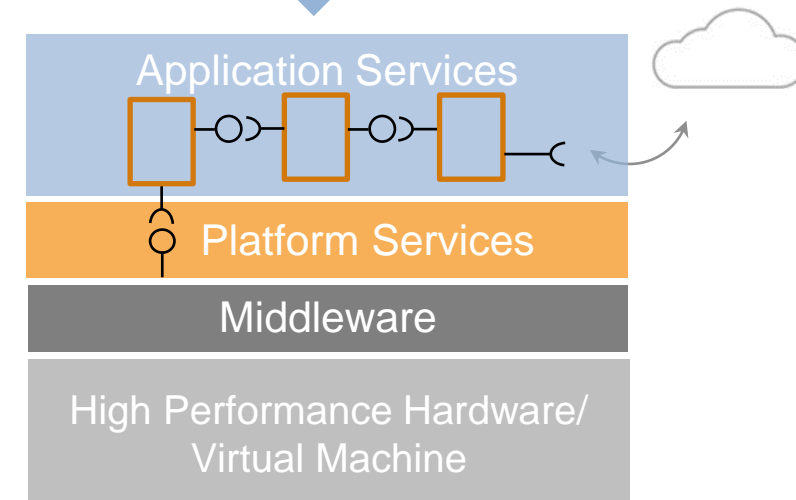


100110
001010
1001100010
001010
100110
001010
010010



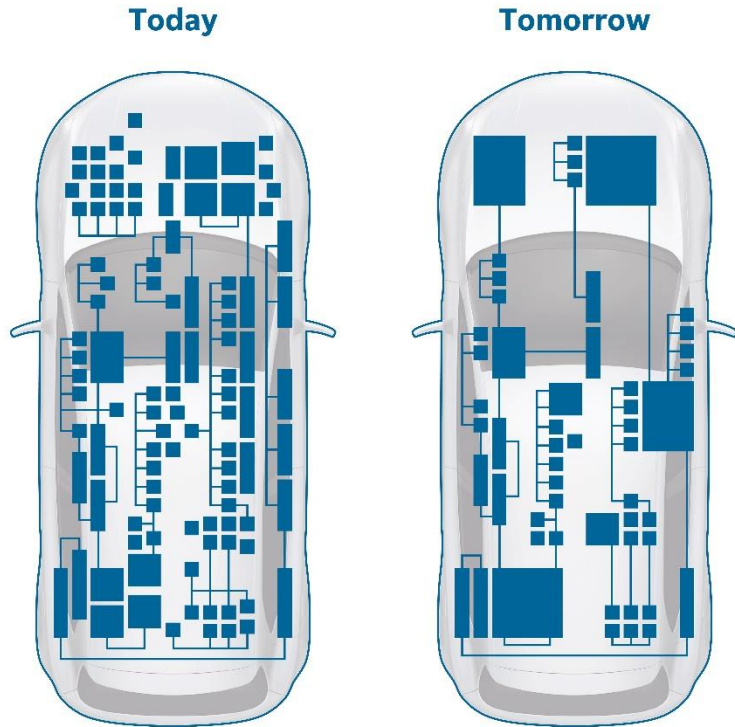
SW updates

- Frequent
- Selective
- Over-the-air

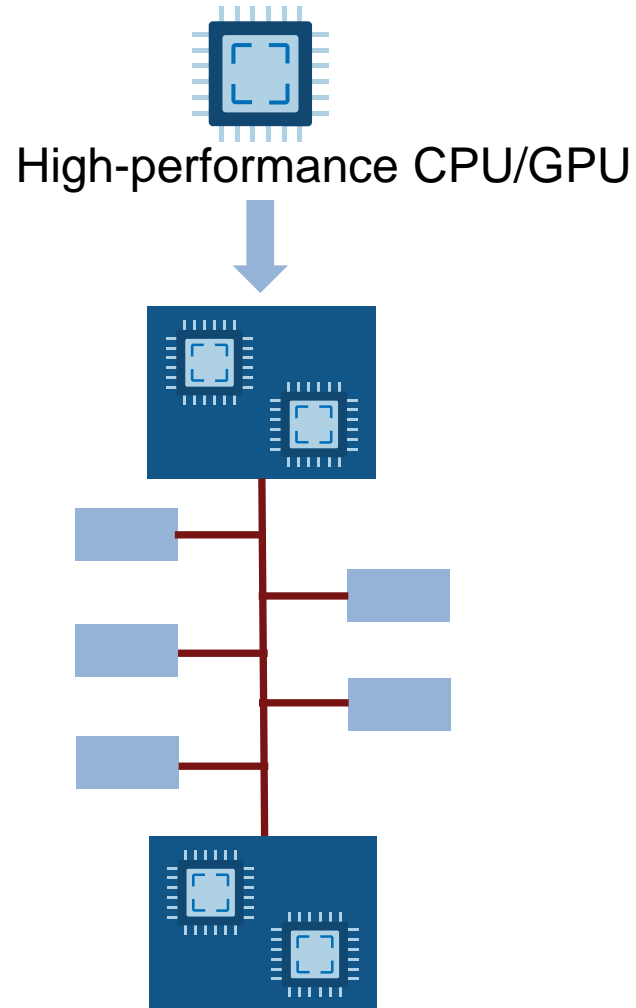


Higher HW abstraction:
Service-oriented architectures

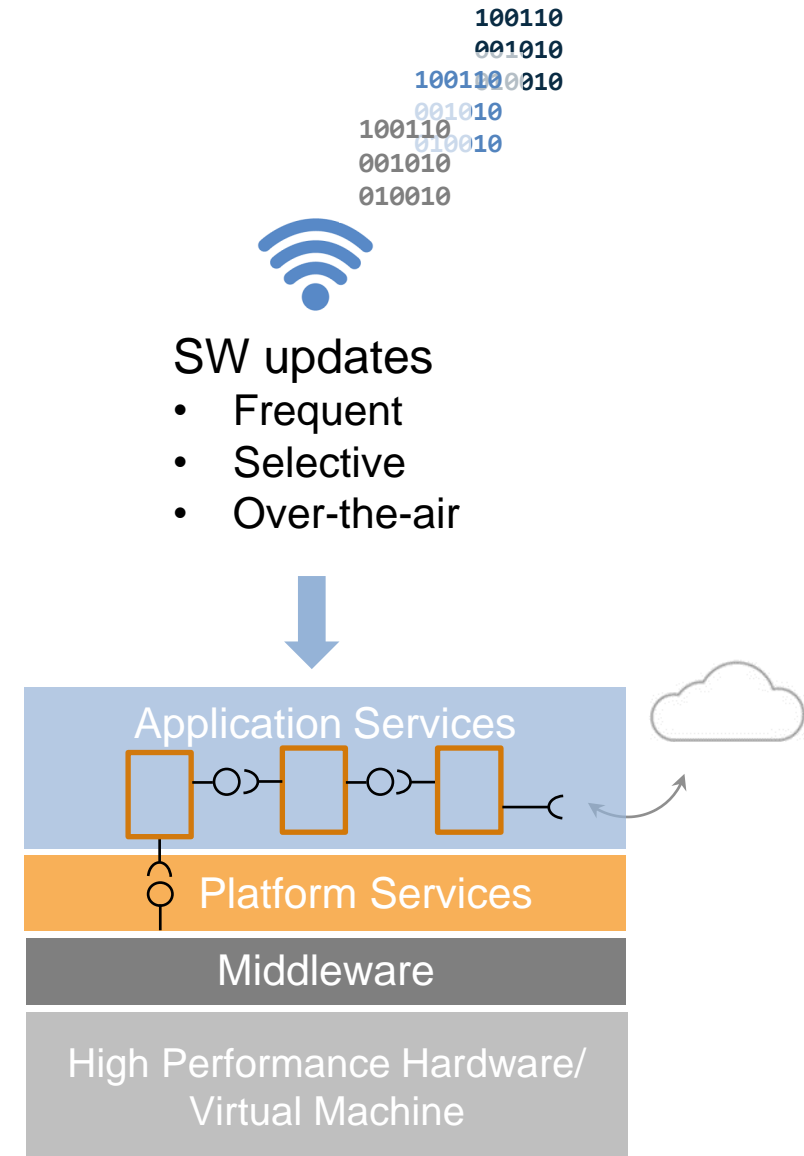
Centralization of computing and SOA



Consolidation and centralization of computing

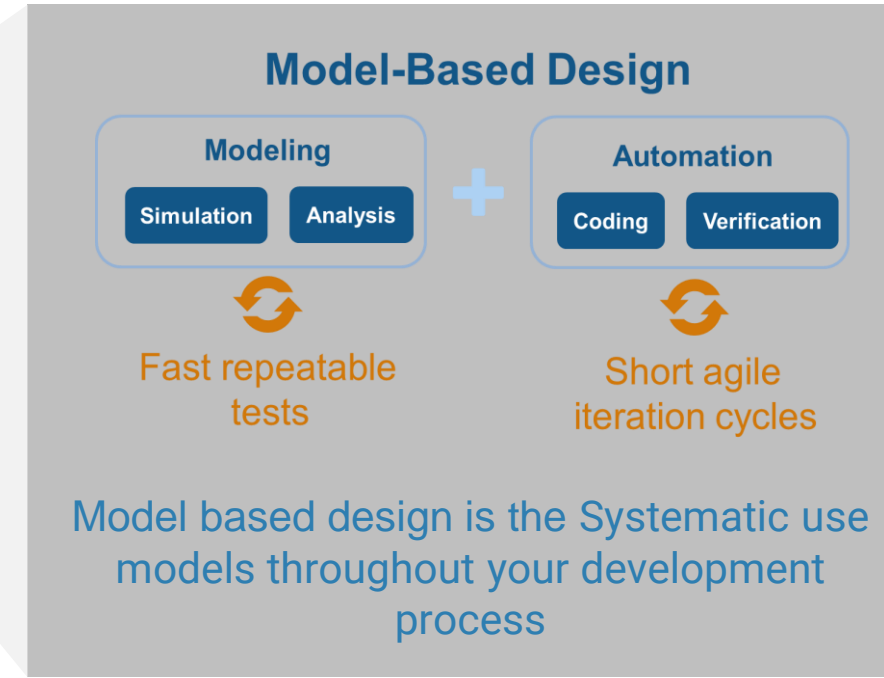
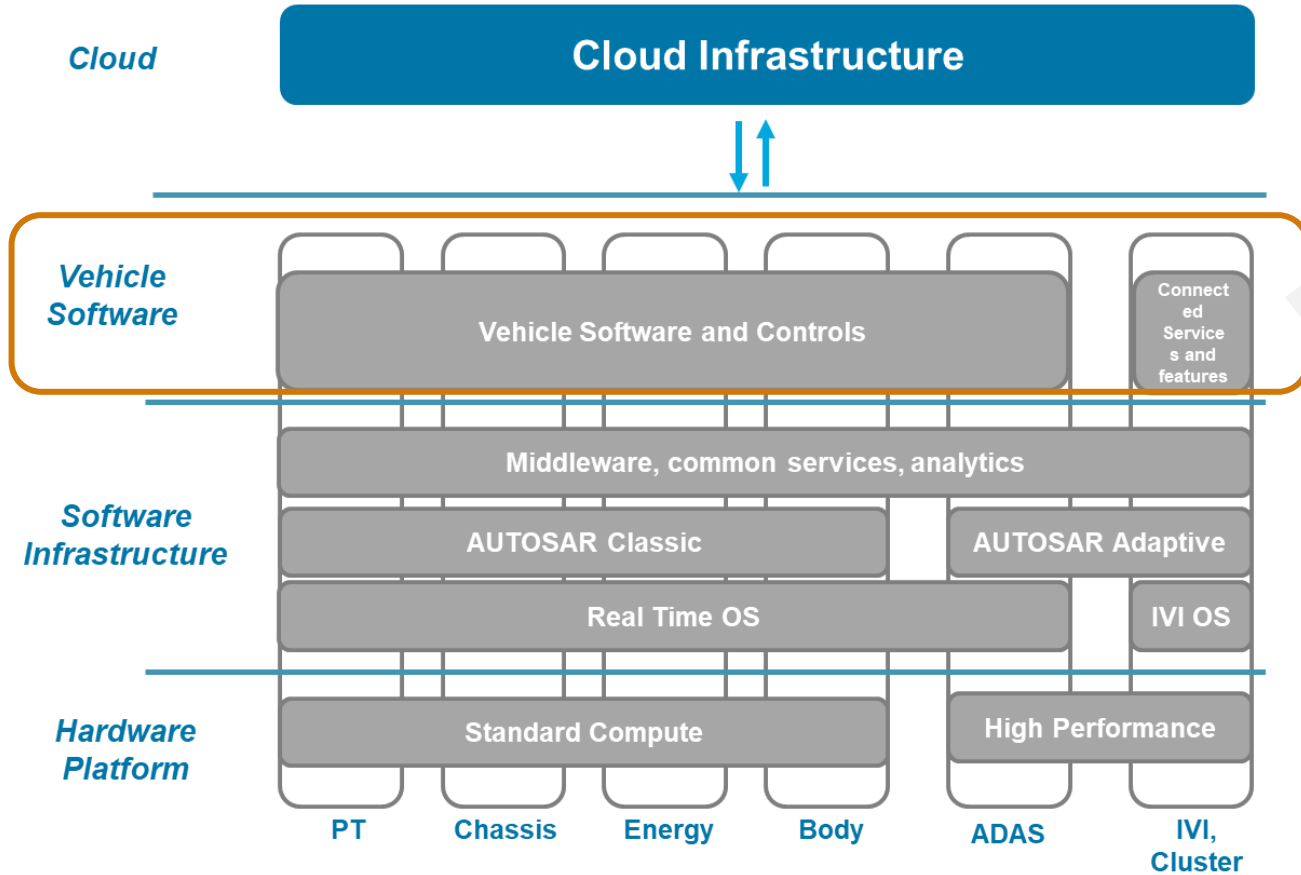


New E/E zonal architectures

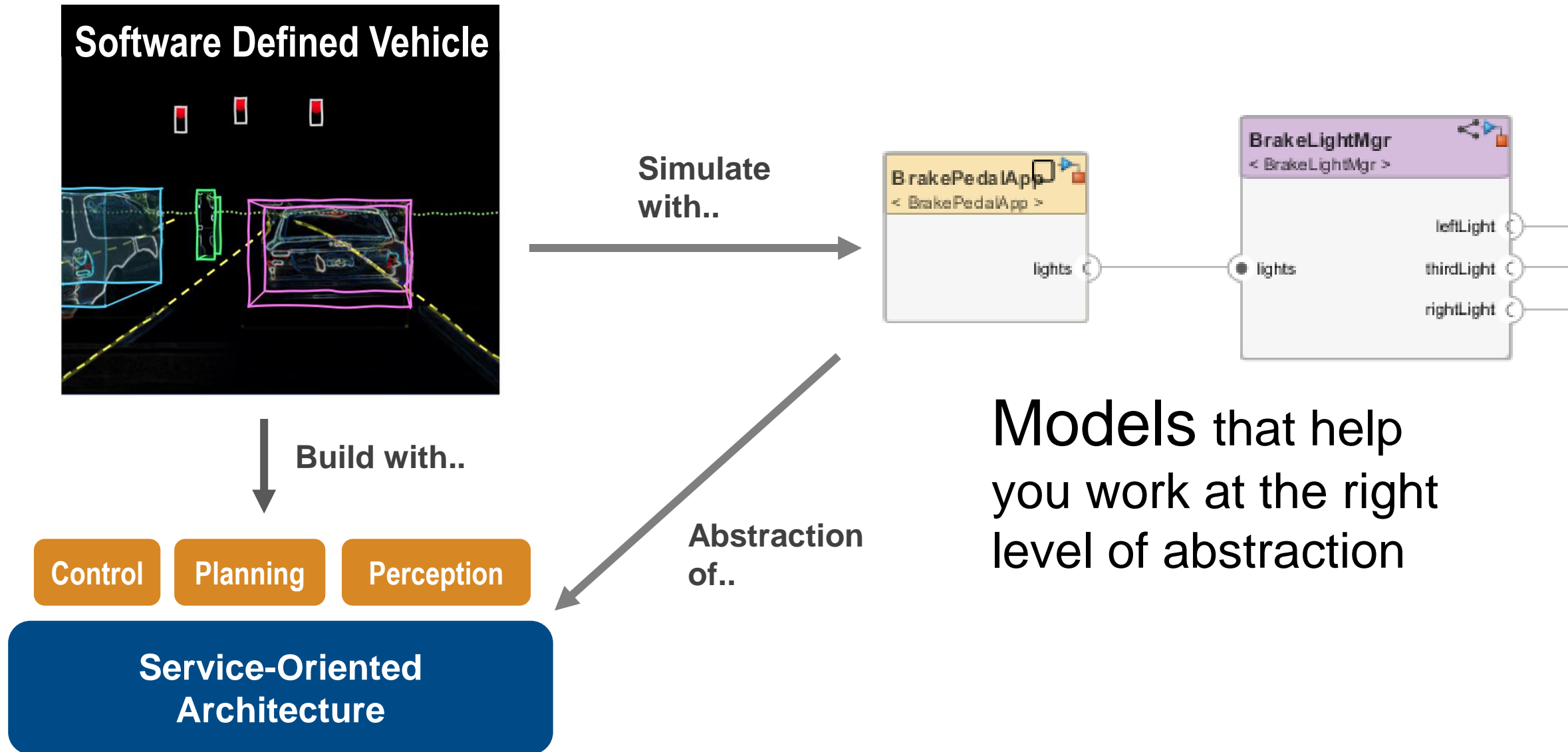


Higher HW abstraction:
Service-oriented architectures

Modeling and Automation for Software Defined Vehicle Applications



Smart CARS call for smart ways to write code

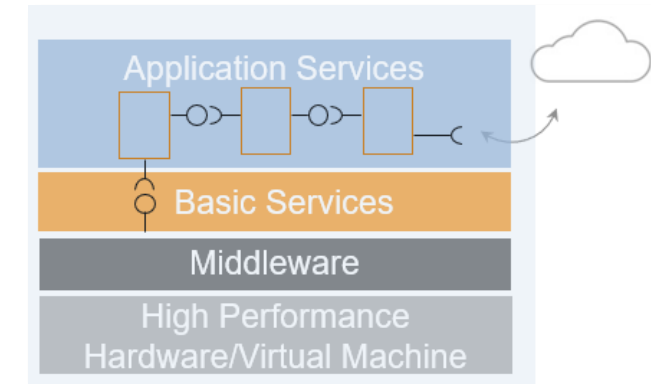


Agenda

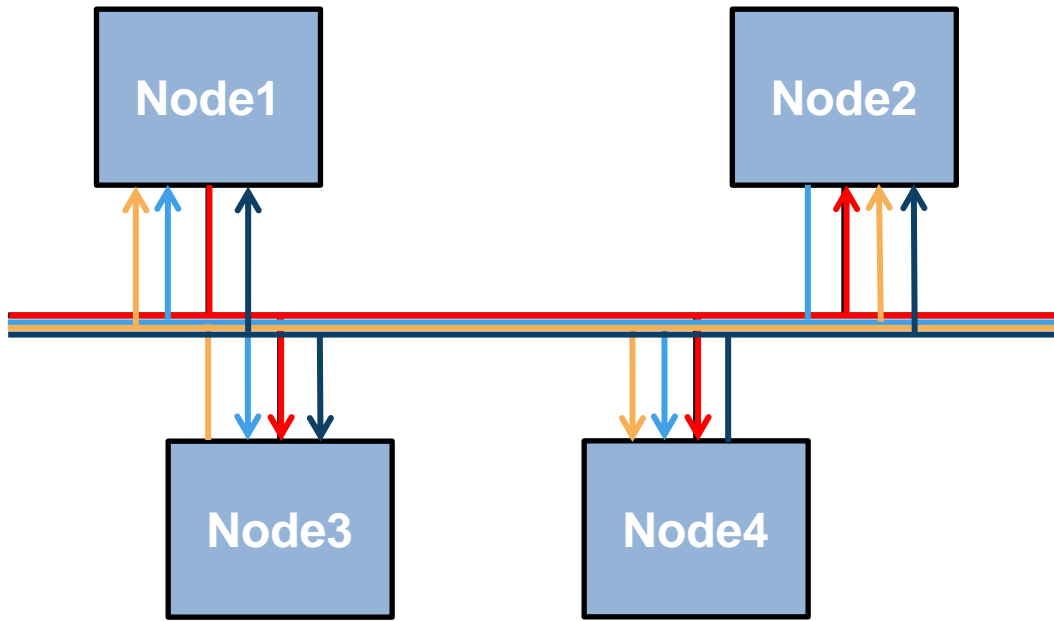
- From Architecture to Deployment
 - Developing SOA Applications
 - Deploying to Linux Targets & Virtual ECUs on Cloud
- Scaling For Production : CI/CD and Model DevOps
 - Case Study : Cruise Control System

SOA – What's it all about?

- SOA consists of services that communicate across different platforms over messages.
- SOA provides flexibility to add, remove, or update components without impacting the entire, typically large, software system
- SOA is used by multiple industrial standards including:
 - AUTOSAR Adaptive Platform
 - DDS (Data Distribution Services)
 - ROS (Robot Operating System)

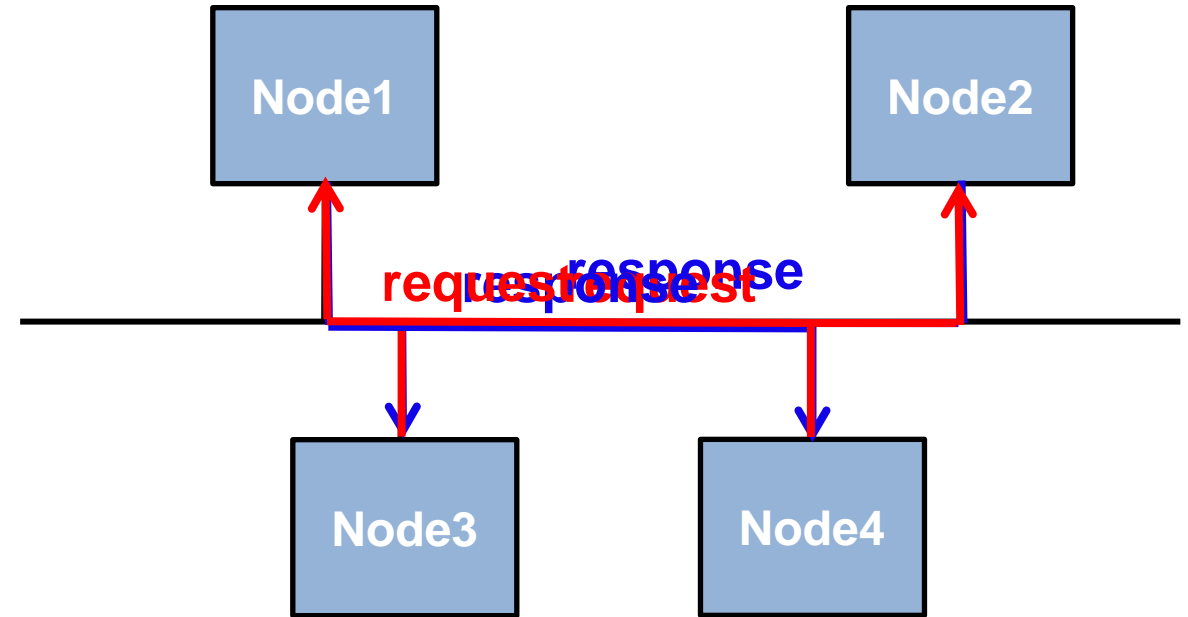


SOC (Service Oriented Communication)



signal-oriented communication

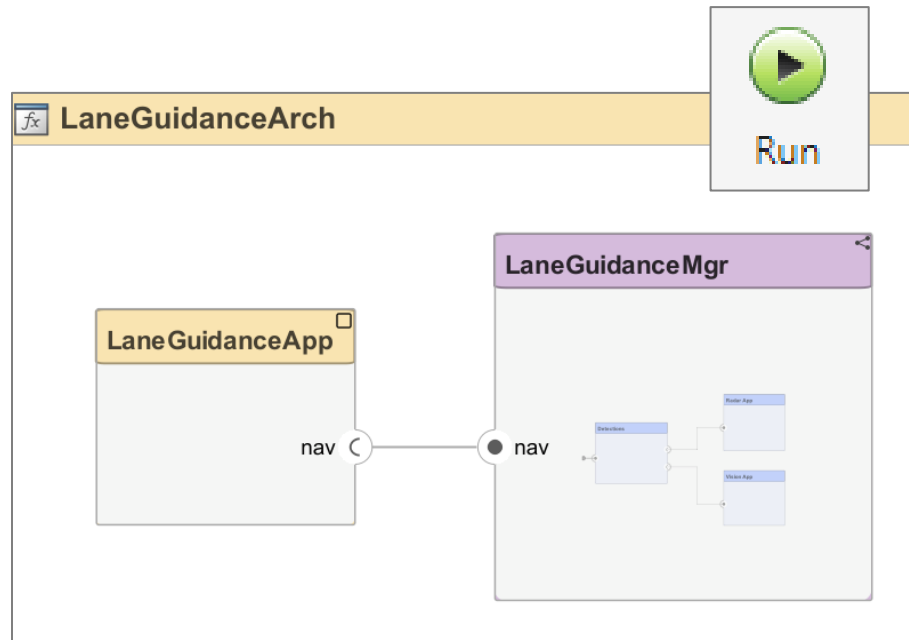
- send data independent of needs
- high bus load
- not efficient



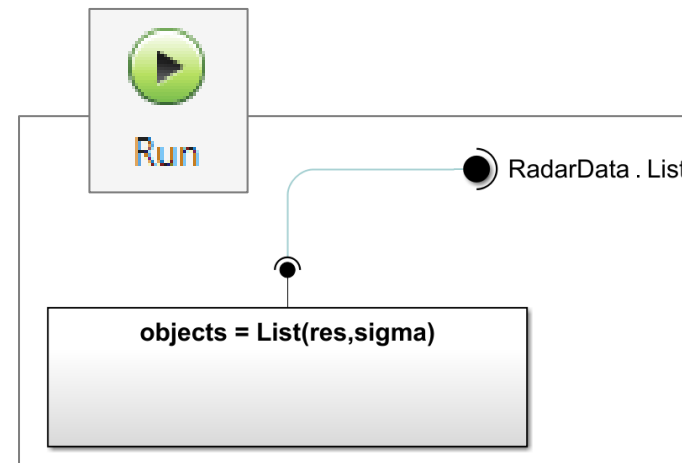
service-oriented communication

- send data dependent of needs
- low bus load
- more efficient

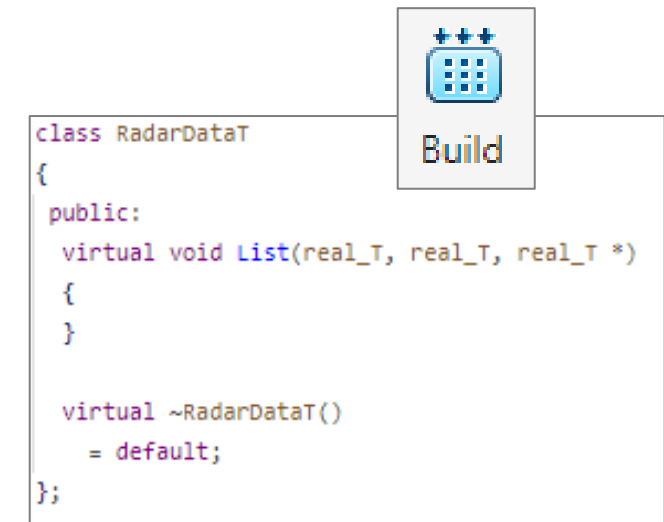
Service-Oriented Architecture (SOA) Design



Describe SOA with
System Composer



Implement detailed
components with
Simulink



Generate code with
Embedded Coder

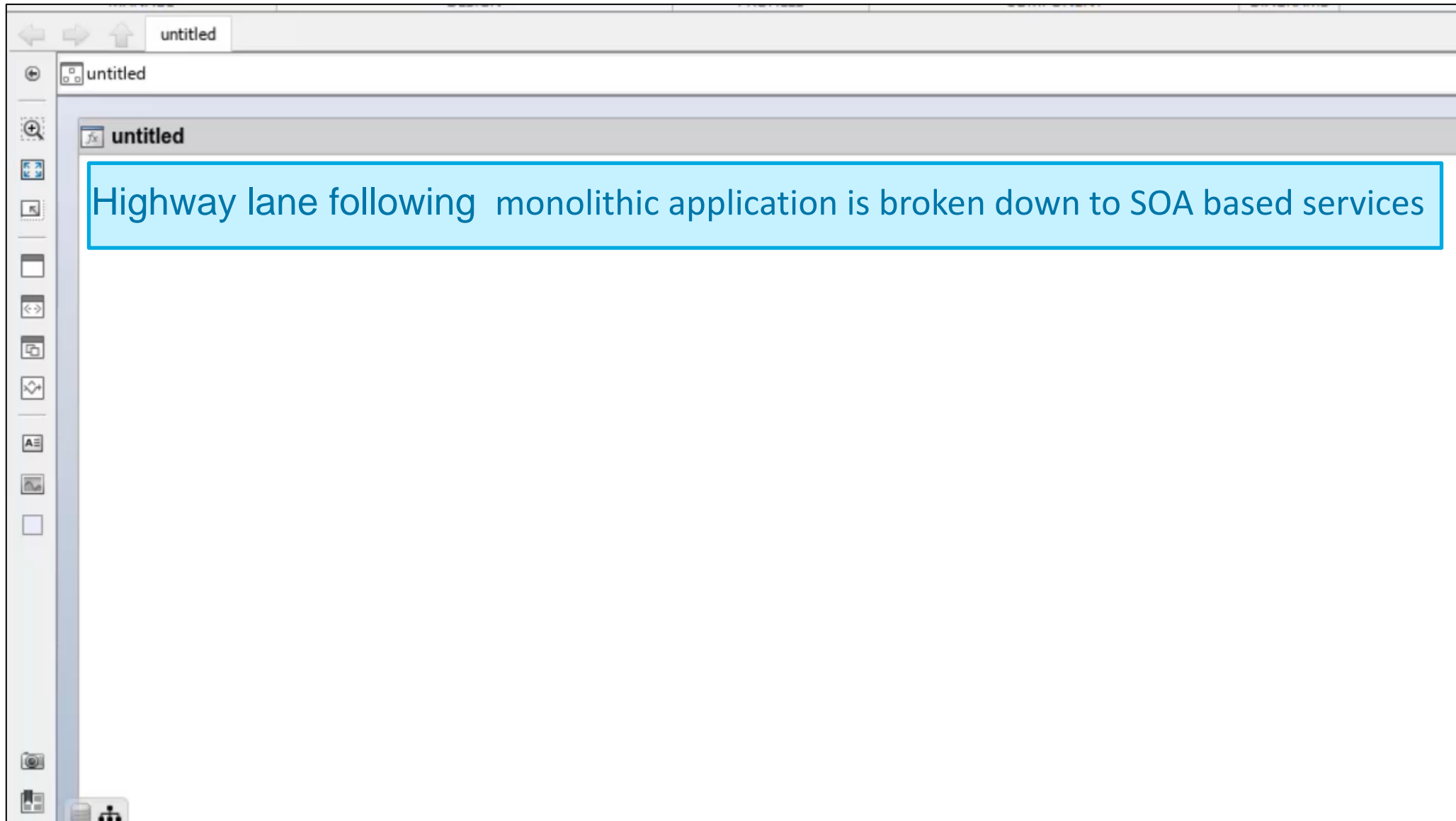
Define Services

Identify and
Analyze
Services

Define
Services and
its interfaces

Define Service
Behavior

Implement
and deploy
Services



Define Service Interfaces

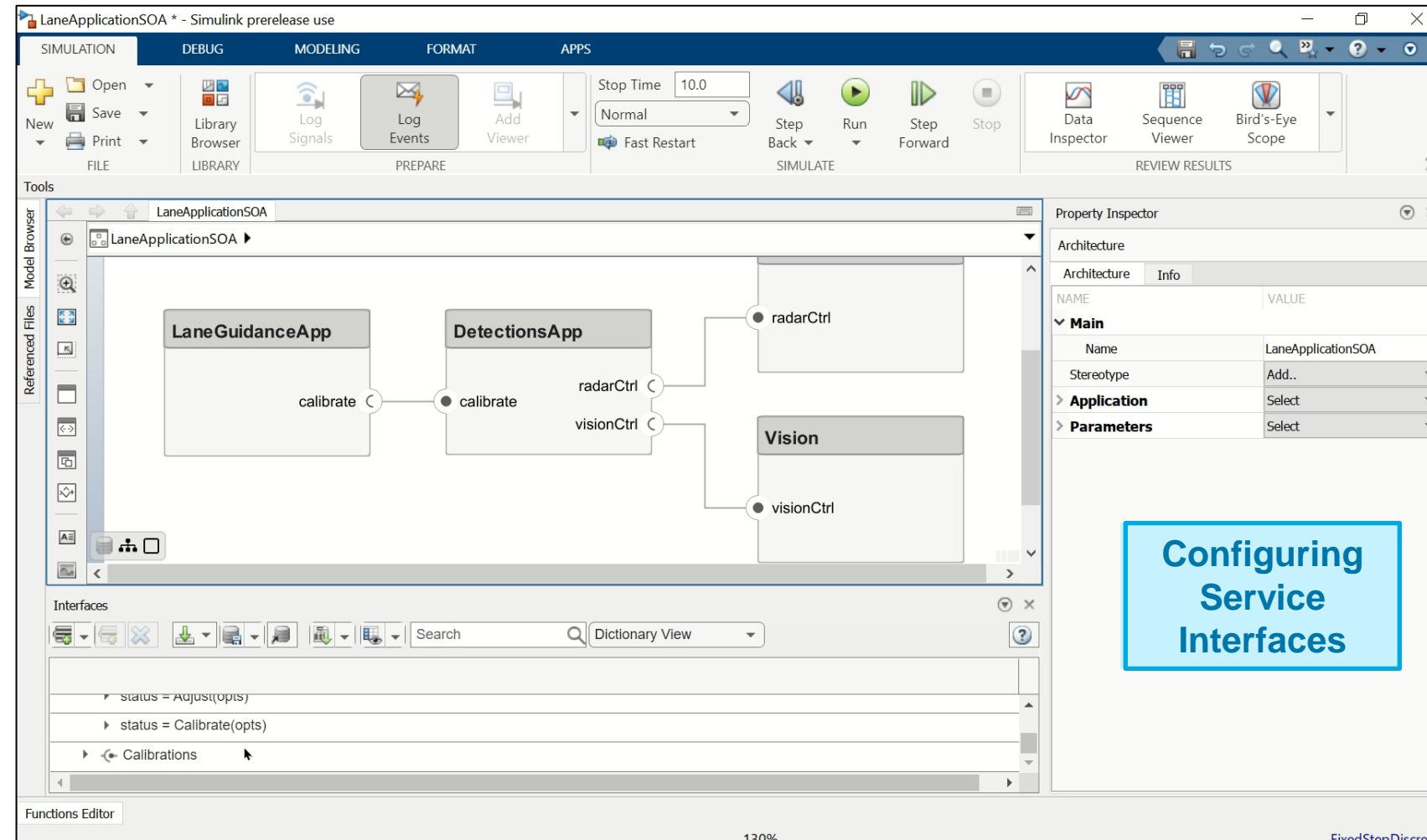
Identify and
Analyze
Services

Define
Services and
its interfaces

Define Service
Behavior

Implement and
deploy
Services

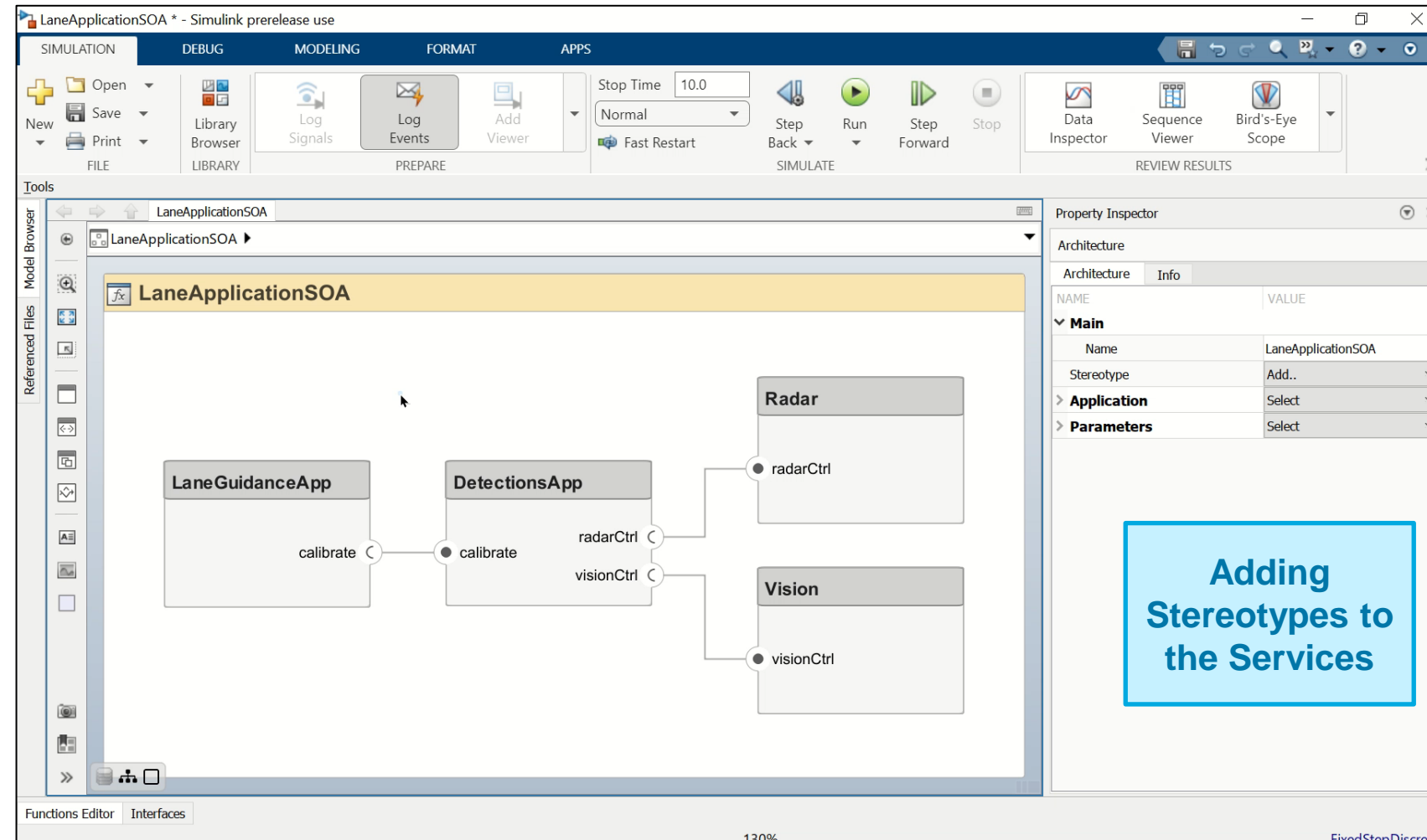
- A service interface is component of service boundary that separates the service from other services and the outside world



Define Service Stereotypes



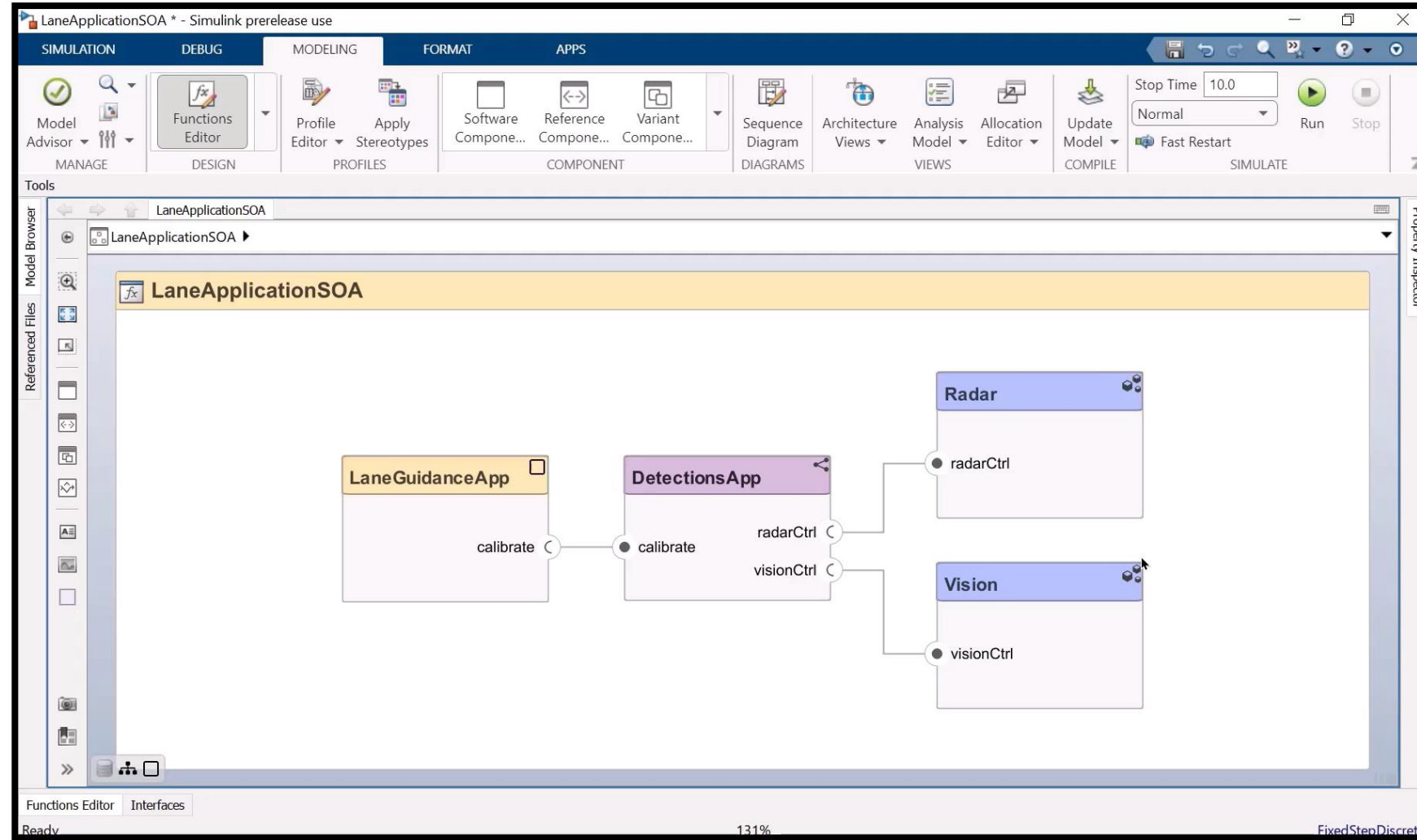
- Stereotypes provide service components with a common set of properties.
- It can help in identifying the boundaries of the services and ensures that each service has a clear and distinct responsibility.



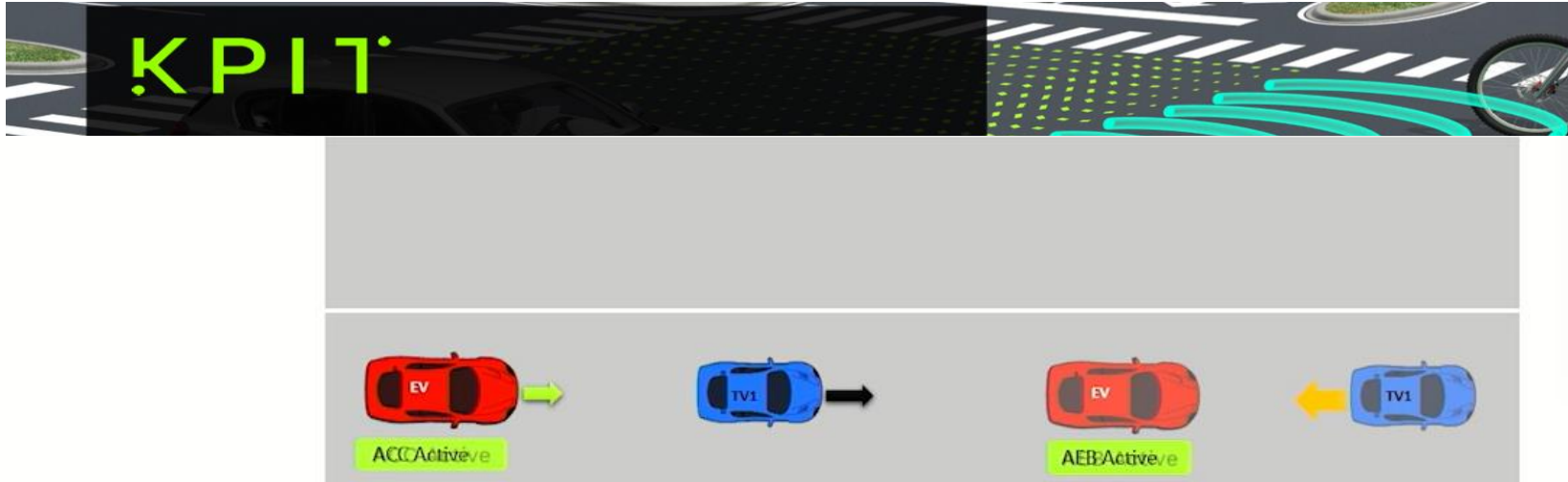
Define Service Behavior








- Each service should have a well-defined inputs, outputs, and behavior.



KPIT- Service-oriented arbitration of ADAS features with Model-Based Design



Advantages over conventional architecture

- 
Scalability : All components are designed to communicate using services resulting in ease for future enhancement. New software components can be designed and incorporated without affecting existing components.
- 
Re-usability : Services can be easily discovered and used when a new feature is deployed. A newly developed feature can depend on services provided by existing software components without updating or redeploying the entire software.
- 
Bandwidth and memory requirement for OTA is less as only specific software components need to update.
- 
Optimization of redundant software components between cross-domain. Services could be discovered and used across different automotive domains.
- 
Running components in **Shadow Mode** in order to test newly deployed version of a software component without affecting the original behaviour or a feature.

KPIT

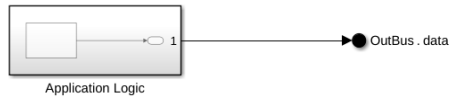

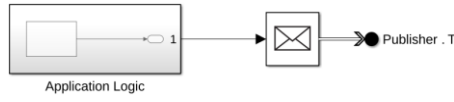
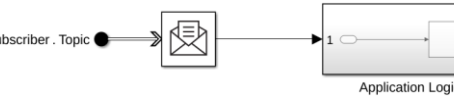


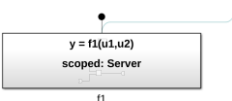
7/10/2023 10

[Link to the talk](#)

Description: -

- Traffic Vehicle (TV1) is cruising on the road with a little lower speed than ego vehicle(EV)(lower relative speed)
- Ego Vehicle enters follow mode and decelerates to match TV1 speed
- After a while, TV1 performs sudden deceleration. Current TTC is less than the threshold TTC for activation of emergency feature. Arbitration accepts the maneuver request of AEB.

Middleware Communication Interfaces in Simulink

		Categories	Simulink Construct	
Communication Interfaces	Shared Memory	Send/Write		
		Receive/Read		
	Messaging	Send/Publish		
		Receive/Subscribe	Polling	
			Event-Driven	
		Remote Procedure Call	Client	Synchronous
	Asynchronous			
	Server			

Implement and Deploy Services

Identify and
Analyze
Services

Define
Services and
its interfaces

Define Service
Behavior

Implement
and deploy
Services

- Services can be implemented based on AUTOSAR Adaptive platform using Model-Based-Design

Property Inspector

NAME	VALUE
Main	
Name	LaneApplicationSOA
Exported Composition Name	LaneApplicationSOA
Stereotype	Add..
Parameters	
	Select

Interfaces Functions Editor Diagnostic Viewer

Ready 77% FixedStepDiscrete

Implement and Deploy Services



- Each service need to be deployed as a standalone application, with its own artifacts including
 - Code
 - C++ Code
 - ARA Stub
 - AUTOSAR interface descriptions
 - Machine Manifest
 - Execution Manifest
 - Service Instance Manifest

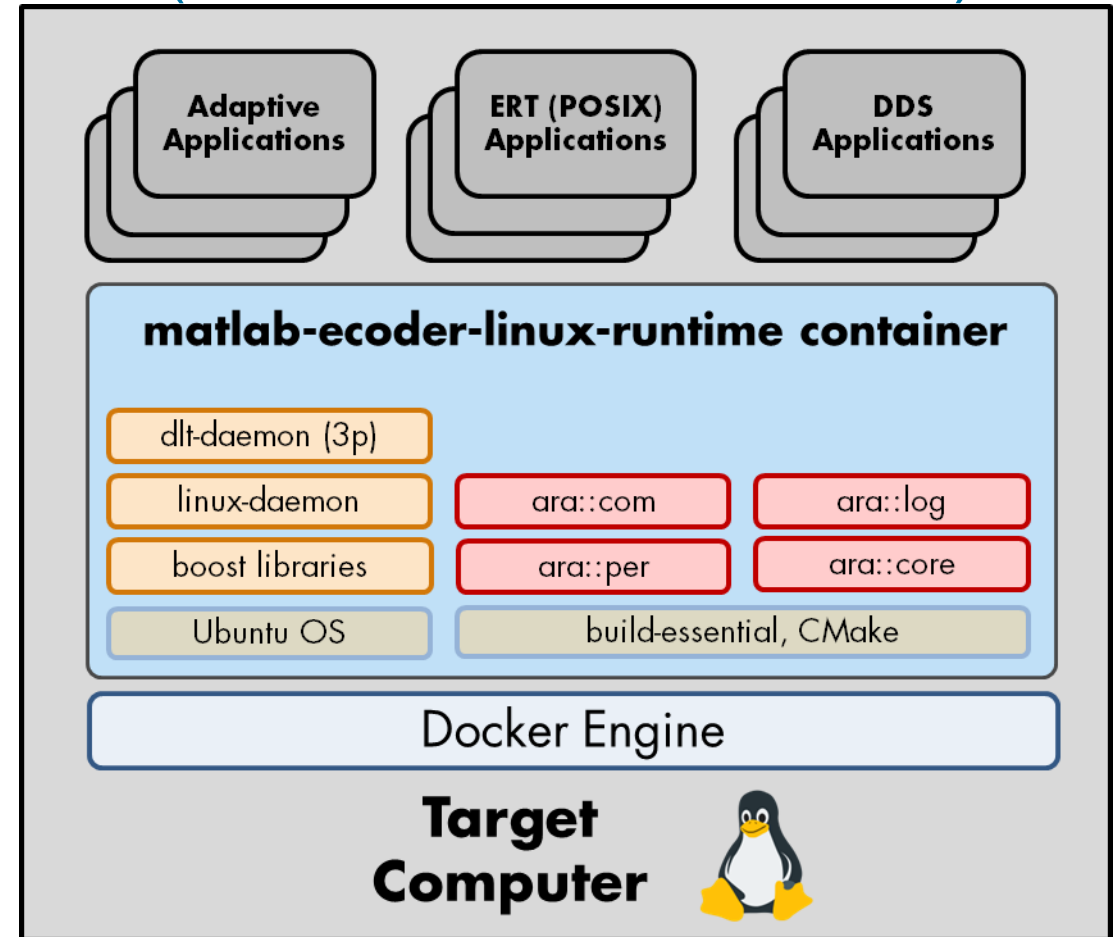
```

Code
Radar.cpp Search
Model files
Radar.cpp
Radar.h
Radar_private.h
Radar_types.h
Shared files
rt_defines.h
rtwtypes.h
Interface files
Radar_ExecutionManifest.arxml
Radar_ServiceInstanceManifest.arxml
Radar_component.arxml
Radar_datatype.arxml
Radar_interface.arxml
Other files
19 #include "Radar.h"
20 #include "rtwtypes.h"
21
22 // Model step function
23 void Radar::Adjust(real_T opts, real_T *status)
24 {
25     UNUSED_PARAMETER(opts);
26
27     // Outputs for Function Call SubSystem: '<Root>/Adjust
28     // SignalConversion generated from: '<S1>/status'
29     *status = 0.0;
30
31     // End of Outputs for SubSystem: '<Root>/Adjust'
  
```

...tiveArch\Radar_autosar_adaptive\Radar.cpp Ln 4 Col 28

Deploy, Control, and Instrument Software Applications on Linux Platform (Run-Time Environment)

- Target Linux system and Docker installation is provided by the customer
- Docker container will be installed by Support Package on target computer



Support Package for Deployment

Help Center

Search Help 🔍

CONTENTS

- « Documentation Home
- « Code Generation
- « Embedded Coder
- « Deployment, Integration, and Supported Hardware

Category

- Callable Function Integration
- Generated Code Interfacing
- Code Packaging
- Model Protection
- Accelerated Simulation
- Embedded Coder Support Package for Linux Applications**
- Embedded Coder Supported Hardware

Documentation
Examples
Functions
Blocks
Apps

Embedded Coder Support Package for Linux Applications

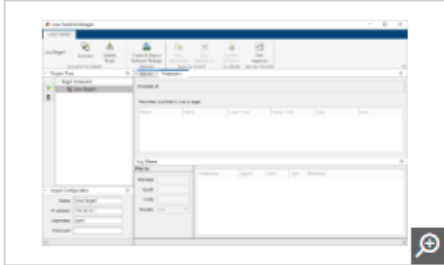
Support package to interface, manage, calibrate the Linux SOA applications

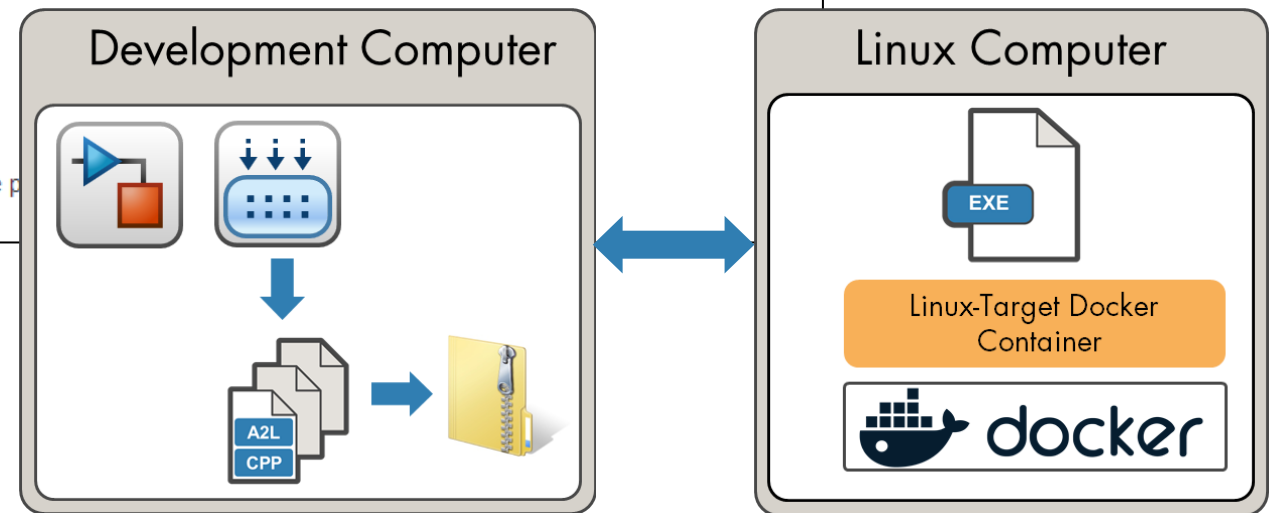
Embedded Coder® Support Package for Linux® Applications supports deploying the generated code, creating the executables, and running/stopping the executables on target, and instrument the running applications. This enables users to interact with multiple target computers at the same time.

The Embedded Coder Support Package for Linux Applications supports:

- Packaging and deployment
- Linux Runtime Manager application
- Log viewer
- Instrumentation

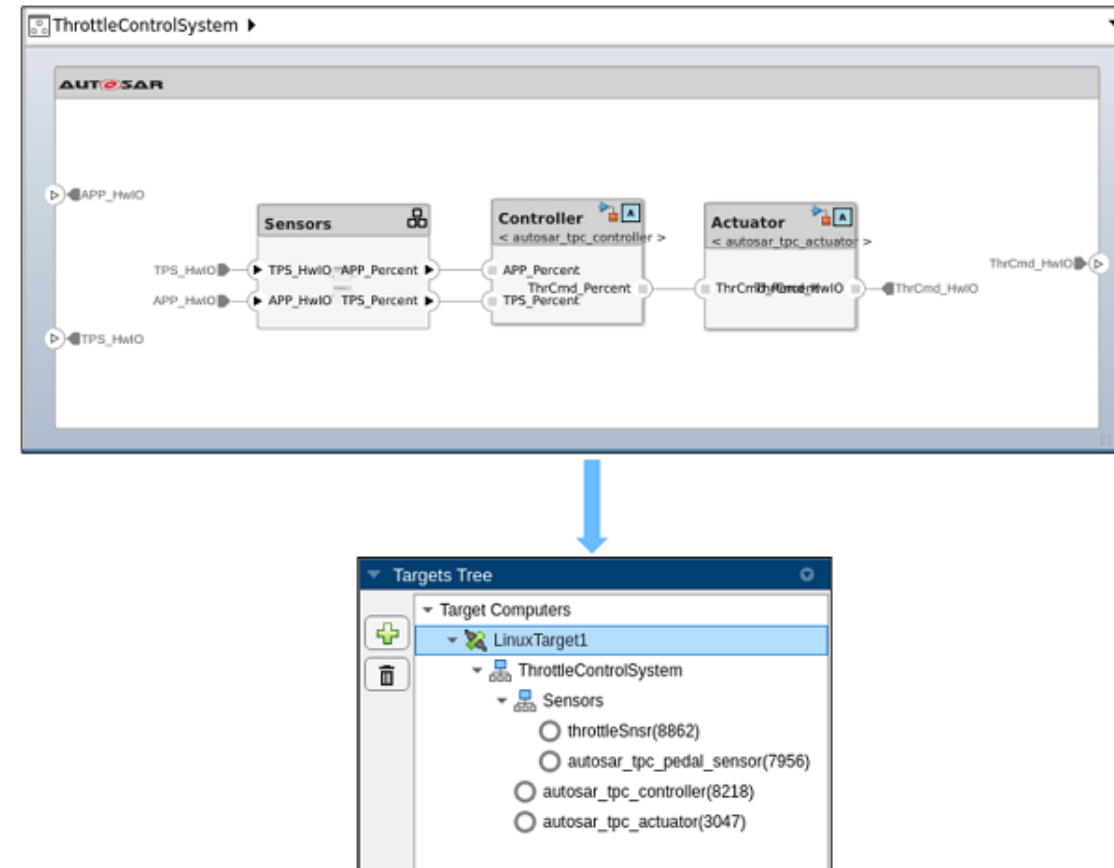
As part of the host services, the support package provides the **Runtime Manager** application.





Deployment support for AUTOSAR Adaptive Architecture models

- Deploy AUTOSAR Adaptive Architecture models using Linux Support Package
- Create application package from AUTOSAR Adaptive Architecture model.
- Deploy AUTOSAR Adaptive Architecture model on Linux target.
- Start/Stop executables of AUTOSAR Adaptive Architecture application on Linux target.



Virtual ECUs deployment and testing

Challenge:

- Ensure application software can integrate into a variety of environments, depending on hardware architecture & middleware
- Verify model based AUTOSAR applications on 'road ready' middleware in the cloud without target hardware

Solution:

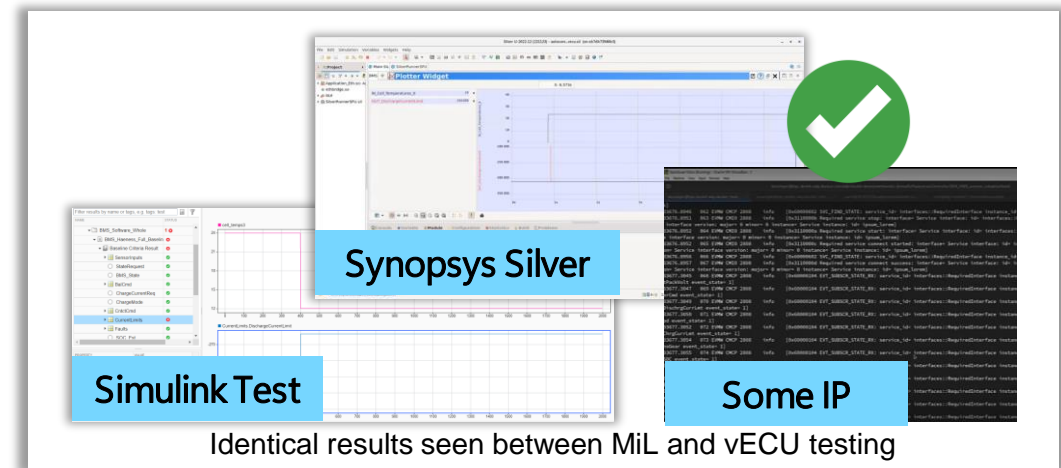
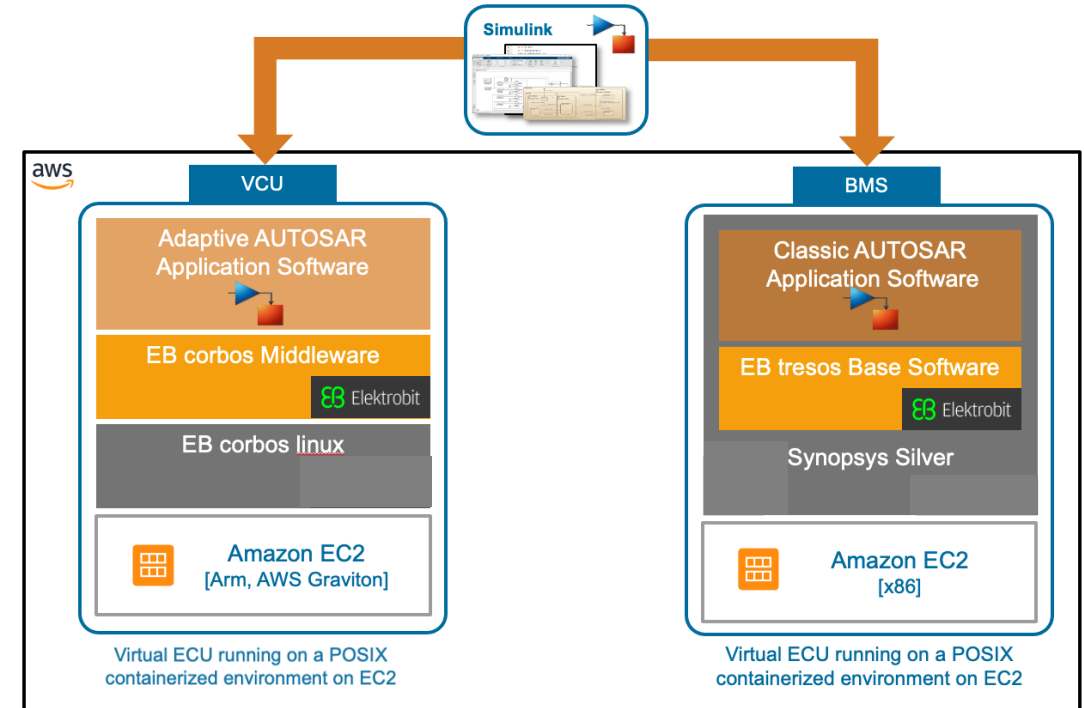
- Customize application code interface for specific middleware/hardware
 - Classic AUTOSAR
 - Adaptive AUTOSAR / DDS / ROS

 **Virtualize**

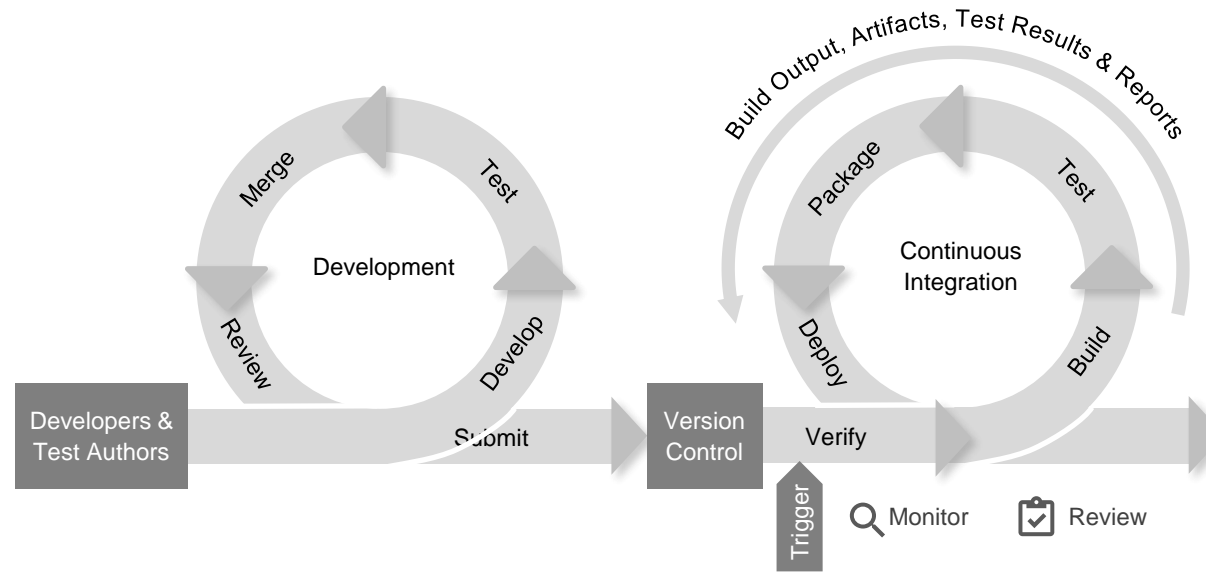
Integrate application code with underlying middleware to deploy within virtual ECUs

Scale

Run virtual ECUs in a cloud native environment and verify the vECU is equivalent to MiL/SiL tests



Continuous Integration



Continuous Integration (CI) originated as a software development process in which developers integrate their code into a shared repository on a regular basis.

Each commit into the shared repository is then verified by an automated build and test.

Continuous Integration Workflow with MATLAB and Simulink

Source Control Server

CI Server

Develop

Test

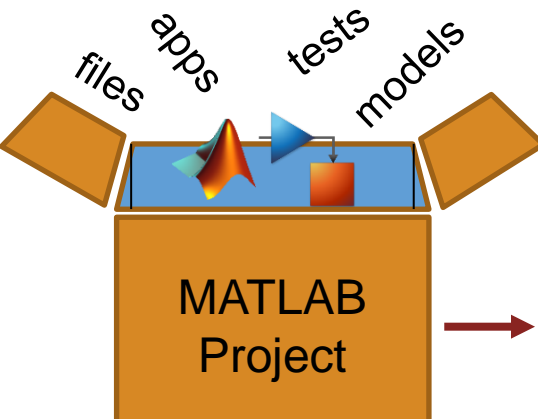
Build

Notify and Deploy

- Run tests:
 - ✓ MATLAB Unit Tests
 - ✓ Simulink Test

- Compile MEX
- Generate Code
- Package (Toolboxes, Apps)

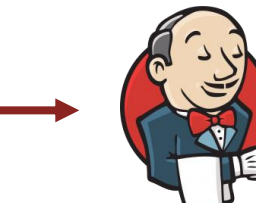
- Publish reports
- Email Notification
- Publish to Server
- Hardware



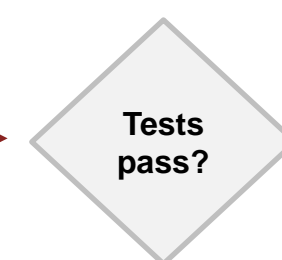
Commit and push changes to Git



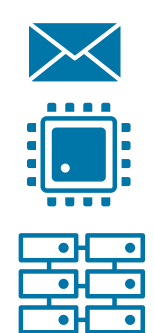
GitLab triggers Jenkins



Jenkins run tests



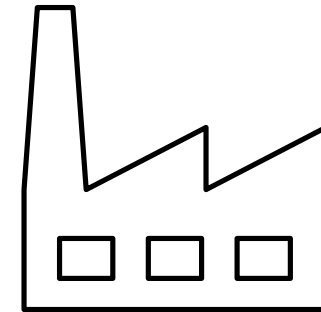
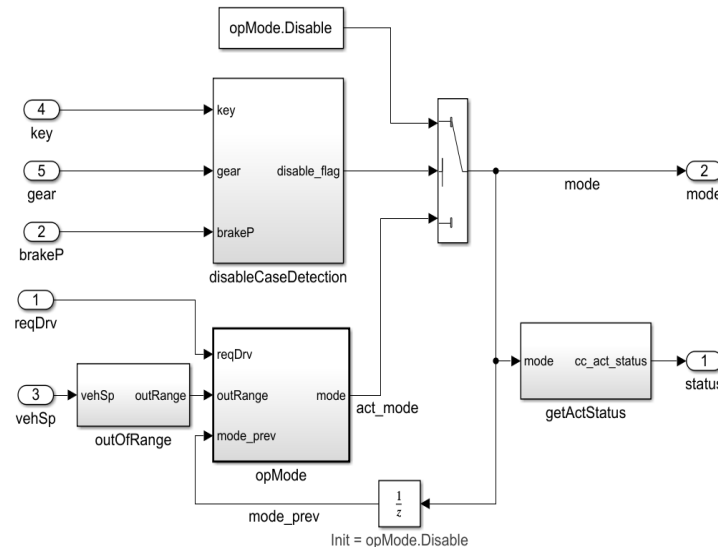
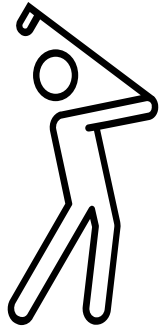
Build, generate code and package



Case Study: Cruise Control System

A Cruise Control Product must meet a new Driver Awareness requirement:

- If the “Driver Awareness” signal is false, then:***
- 1. Cruise control function shall be disabled.***
 - 2. And Enable shall be prevented.***



Is the driver aware?

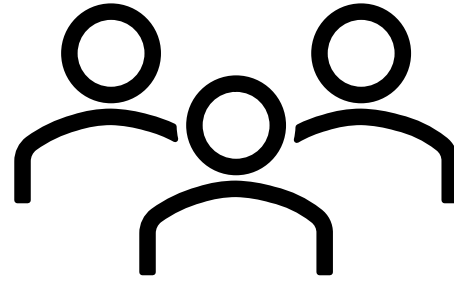
Modify logic and tests

Production

[Learn More:](#) The MBD Artifacts and YAML files for this case study can be found here

Case Study: Plan

- A team is formed

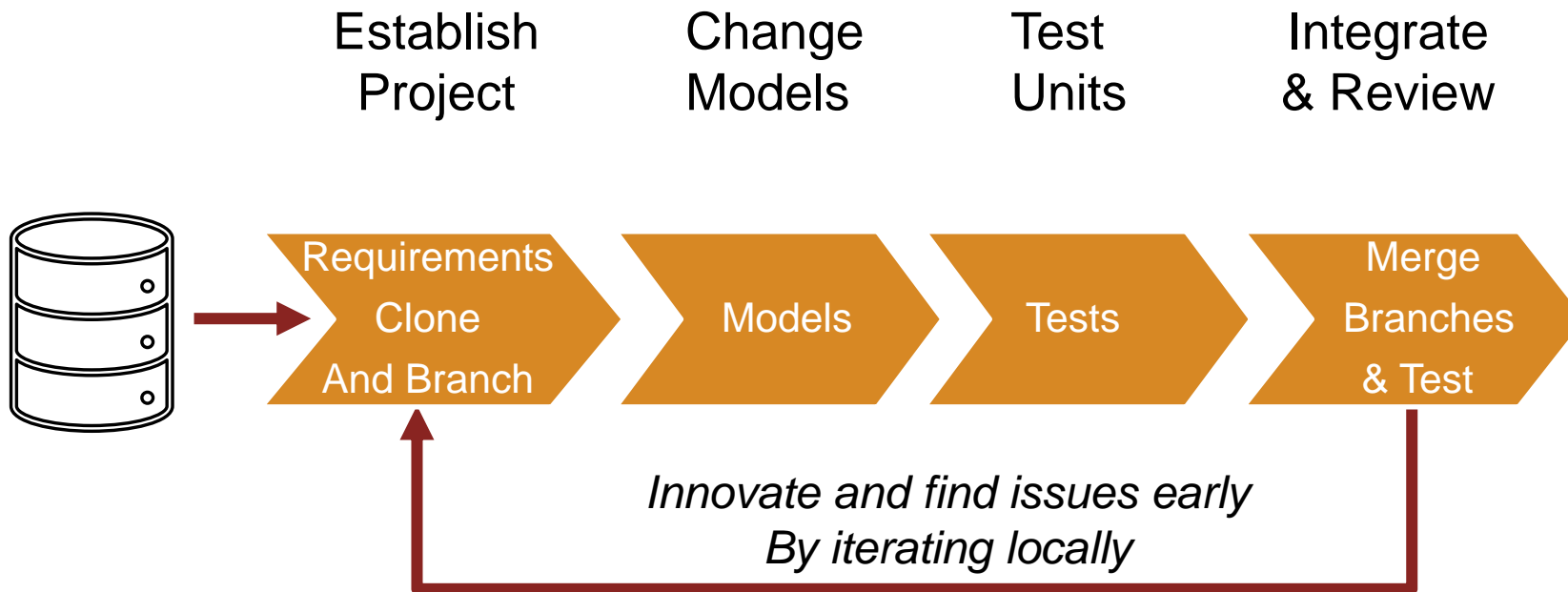


- **Team leader**, Source Control, & Integration
- **Controls Engineer 1** for Prevent Enable
- **Controls Engineer 2** for Disable
- **Test Engineer**

- The Driver Awareness feature must be released within weeks
- Fortunately, the process had been automated → speed and quality

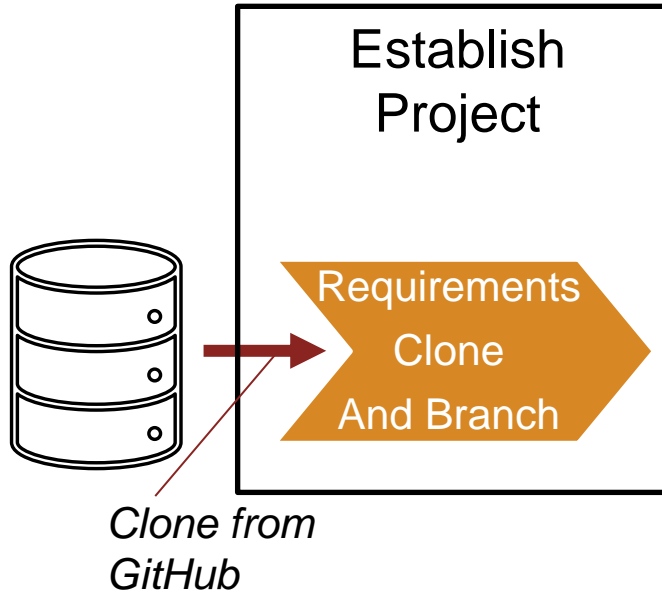
Takeaway #1: Iterate locally to innovate and prequalify

General Example



Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Clone project and create branches



Review Programmatic Operations

I will run Sections of this script
To perform most operations
programmatically.

2. Link with remote SCM

This informs MATLAB where the work will be pushed to for pipeline operations

>> Copy/ paste this GitLab project URL into the "Project Remote:

```
https://insidelabs-git.mathworks.com/bjohnson/demo\_2023\_a.git
```

>> Validate the URL

3. Sign in to the GitLab remote if needed

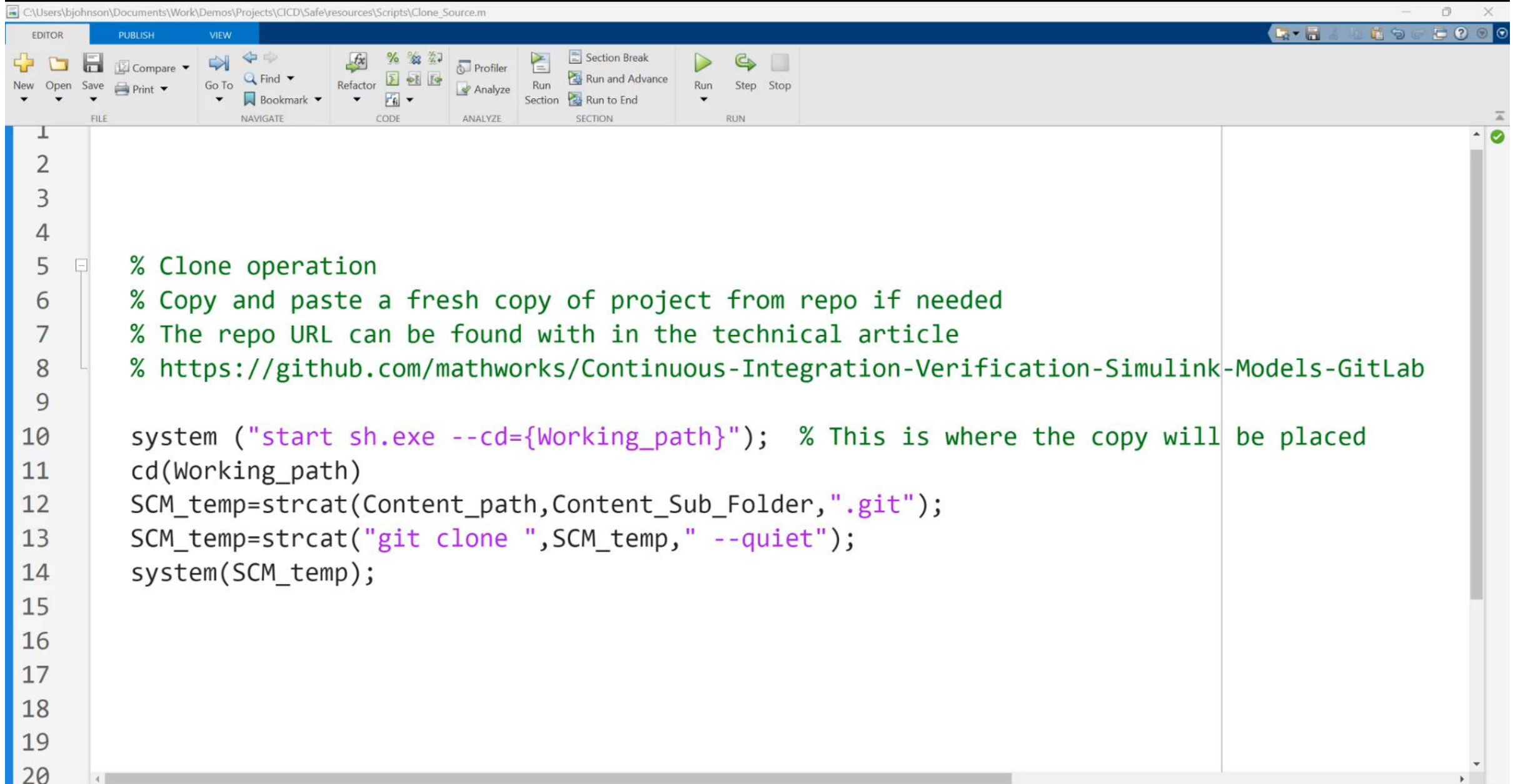
MUST - Run Section

```
9 GitLab_URL="https://insidelabs-git.mathworks.com/users/sign_in";  
10 web(GitLab_URL);
```

MUST - Run Section

3.1 Reset the GitLab project to the fresh clone

```
11 Runner_Location='C:\Gitlab-Runner'  
12 Assure_Runner_Running;  
13  
14 Demo=1;  
15 if Demo  
16 system('git add .');  
17 system('git commit -am "Reset Project without running pipeline" -q');  
18 system('git push -f origin main -o ci.skip'); % skip running the pipeline  
19 else  
20 system('git add .');  
21 system('git commit -am "Reset Project and run pipeline" -q'); %q: dont echo  
22 system('git push -f origin main');  
23 end
```

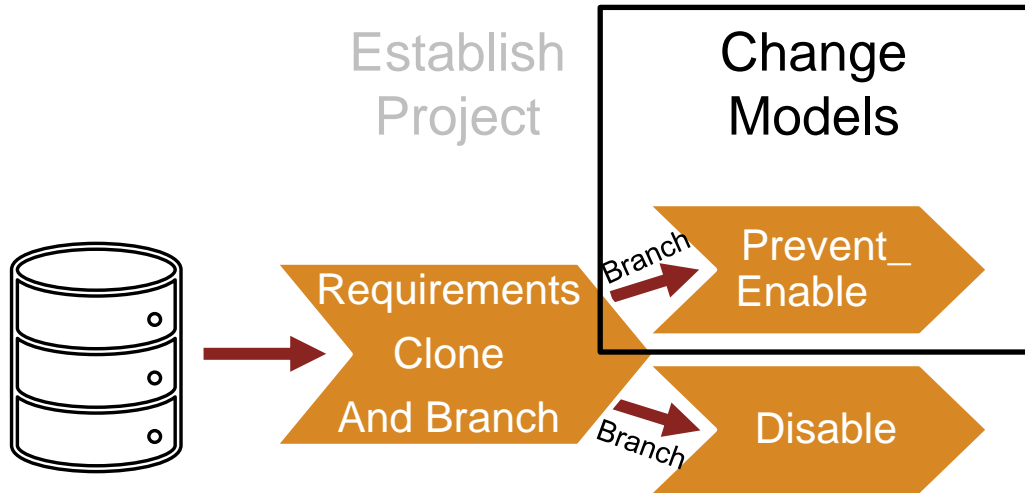


The screenshot shows the MATLAB IDE interface with a script editor. The script contains comments and MATLAB code for cloning a repository. The comments are in green, and the code is in purple. The script is as follows:

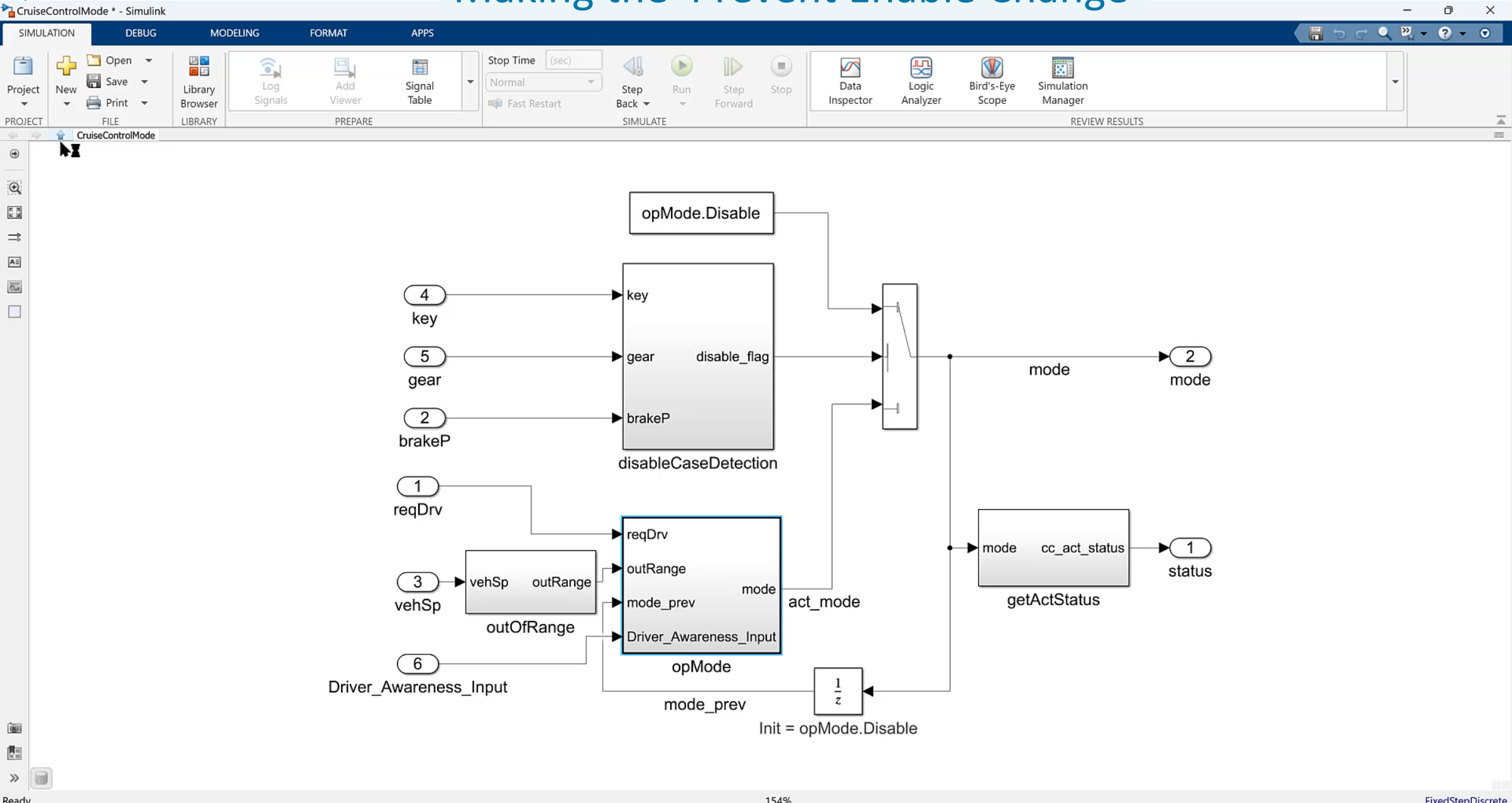
```
1  
2  
3  
4  
5 % Clone operation  
6 % Copy and paste a fresh copy of project from repo if needed  
7 % The repo URL can be found with in the technical article  
8 % https://github.com/mathworks/Continuous-Integration-Verification-Simulink-Models-GitLab  
9  
10 system ("start sh.exe --cd={Working_path}"); % This is where the copy will be placed  
11 cd(Working_path)  
12 SCM_temp=strcat(Content_path,Content_Sub_Folder,".git");  
13 SCM_temp=strcat("git clone ",SCM_temp," --quiet");  
14 system(SCM_temp);  
15  
16  
17  
18  
19  
20
```

Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Change Models

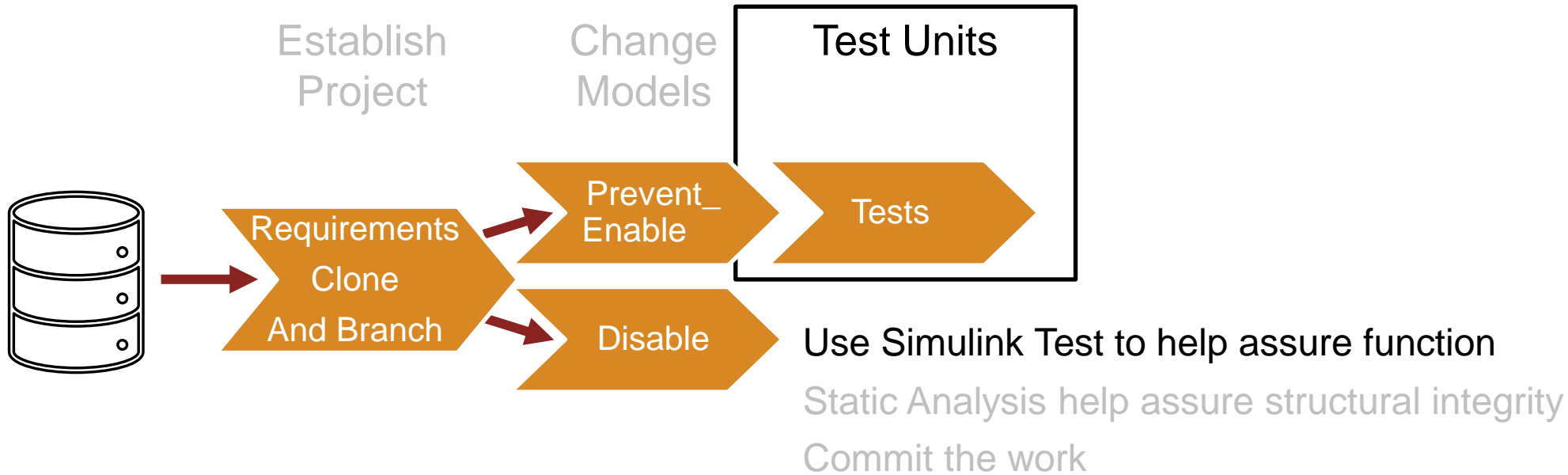


Making the Prevent Enable Change



Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Test Units

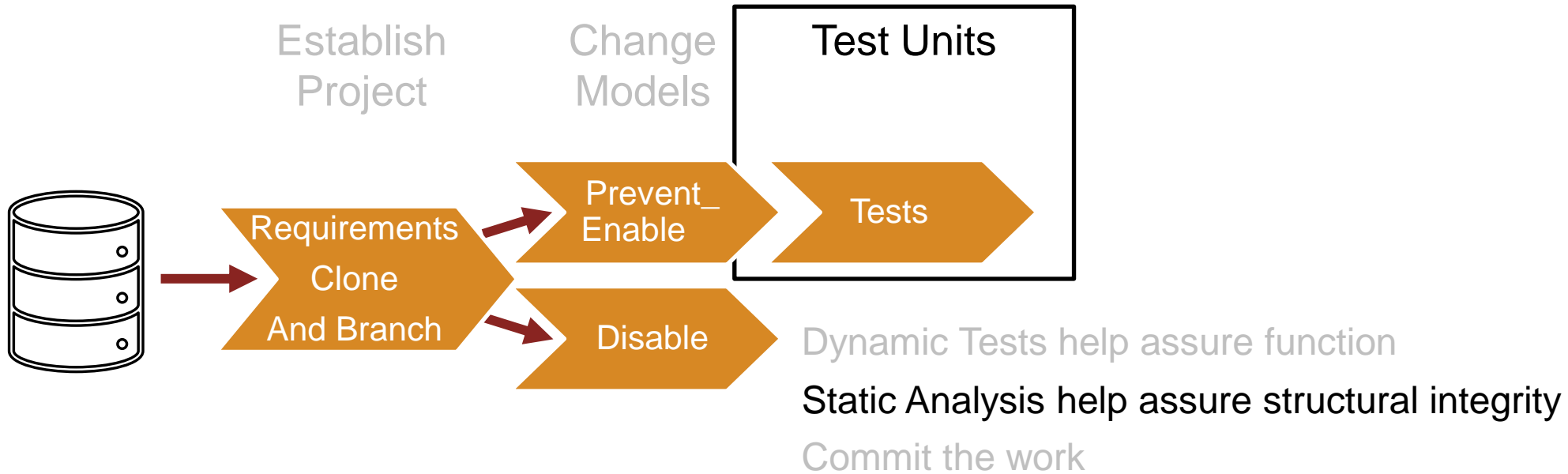


The screenshot displays the Test Manager application interface. At the top, there is a menu bar with 'TESTS' and a toolbar with various icons for file operations (New, Open, Save, Cut, Copy, Paste, Delete, Test Spec Report), execution (Run, Run with Stepper, Stop, Parallel), and analysis (Report, Visualize, Highlight in Model, Export, Import, Model Testing Dashboard). Below the toolbar are tabs for 'Test Browser' and 'Results and Artifacts'. The 'Test Browser' tab is active, showing a search filter and a tree view of test files under 'CruiseControlModeTestManager'. The 'Results and Artifacts' tab is also visible, showing a 'Start Page' with a 'Getting Started' section, 'New Test File' and 'Open Test File' buttons, and sections for 'RECENT FILES' and 'DOCUMENTATION'. The 'RECENT FILES' section lists 'CruiseControlModeTestManager' and 'crs_controllerTestManager'. The 'DOCUMENTATION' section lists 'Getting Started with Simulink Test'. At the bottom left, there is a 'PROPERTY' table for the selected test file.

PROPERTY	VALUE
Name	CruiseControlModeTestMa
Location	C:\Users\bjohnson\Docume...
Enabled	<input checked="" type="checkbox"/>
Tags	Type comma or space separa

Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Static Analysis



Projects - Dashboard - GitLab x Model Advisor Report for 'Cruise' x

File | C:/Users/bjohnson/Documents/Work/Demos/Projects/CICD/Working_CI_Dir/Continuous-Integration-Verification-Simulink-Models-GitLab/Design/CruiseControlMode/pipe...

Import favorites | Google | Inside | Bing | New tab | Other favorites

Model Advisor Report - CruiseControlMode.slx

Simulink version: 10.6
System: CruiseControlMode
Treat as Referenced Model: off

Model version: 8.8
Current run: 07-Apr-2023 13:46:29
Model Advisor configuration: iso26262Checks.json

Run Summary

Incomplete	Failed	Warning	Justified	Passed	Not Run	Total
0	0	0	0	39	0	39

Model Advisor

- 1 Modeling Standards for ISO 26262 0 0 0 0 39 0

Display configuration management data (Required)

Display model configuration and checksum information

Model configuration and checksum information

Attribute	Value
Model Version	8.8
Author	bjohnson
Date	Thu Mar 02 16:03:30 2023
Model Checksum	2071651209 3732916041 802016950 2965375980

Filter checks

- Passed
- Failed
- Warning
- Not Run
- Justified
- Incomplete

Navigation

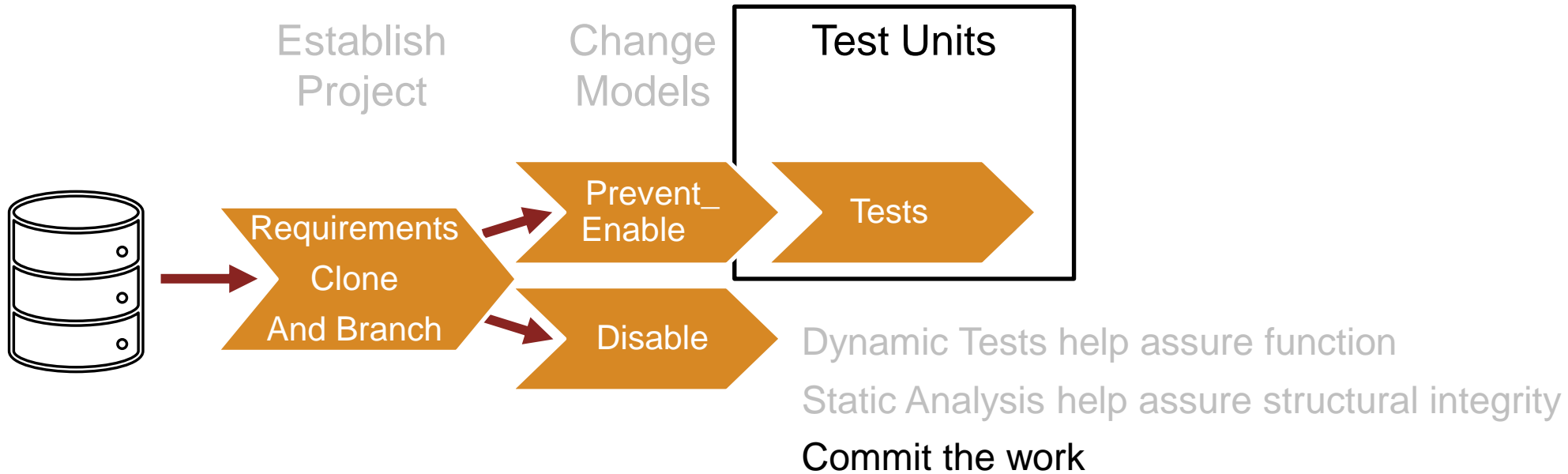
- Model Advisor
 - 1 Modeling Standards for ISO 26262
 - 1.1 High-Integrity Systems
 - 1.1.1 Simulink
 - 1.1.2 Stateflow
 - 1.1.3 MATLAB
 - 1.1.4 Configuration
 - 1.1.5 Naming
 - 1.1.6 Code

View

- Scroll to top
- Hide check details

Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Static Analysis



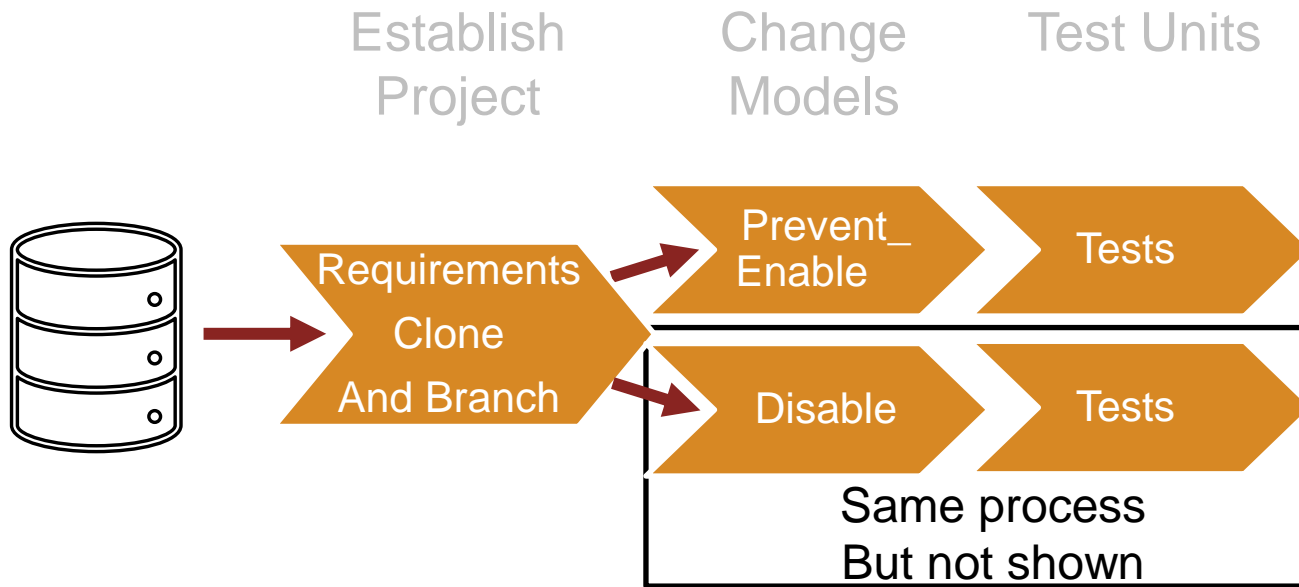
Committing Prevent Enable

The image shows the MATLAB R2022b interface with the 'PROJECT SHORTCUTS' tab active. The workspace contains a project named 'CruiseControlExample'. The file browser on the left shows a directory structure with folders like 'Code', 'Data', 'Design', 'Requirements', 'resources', and 'tools', along with various configuration files such as '.crs_controller-gitlab-ci.yml', '.cruiseControlMode-gitlab-ci.yml', and '.gitattributes'. The command window on the right displays the output of a simulation run, showing 10 completed simulation runs between 15:13:32 and 15:14:58 on 07-Apr-2023. The prompt 'fx >>' is visible in the command window.

```
[07-Apr-2023 15:13:32] Running simulations...
[07-Apr-2023 15:13:41] Completed 1 of 10 simulation runs
[07-Apr-2023 15:13:52] Completed 2 of 10 simulation runs
[07-Apr-2023 15:14:01] Completed 3 of 10 simulation runs
[07-Apr-2023 15:14:09] Completed 4 of 10 simulation runs
[07-Apr-2023 15:14:16] Completed 5 of 10 simulation runs
[07-Apr-2023 15:14:26] Completed 6 of 10 simulation runs
[07-Apr-2023 15:14:34] Completed 7 of 10 simulation runs
[07-Apr-2023 15:14:41] Completed 8 of 10 simulation runs
[07-Apr-2023 15:14:49] Completed 9 of 10 simulation runs
[07-Apr-2023 15:14:58] Completed 10 of 10 simulation runs
fx >>
```

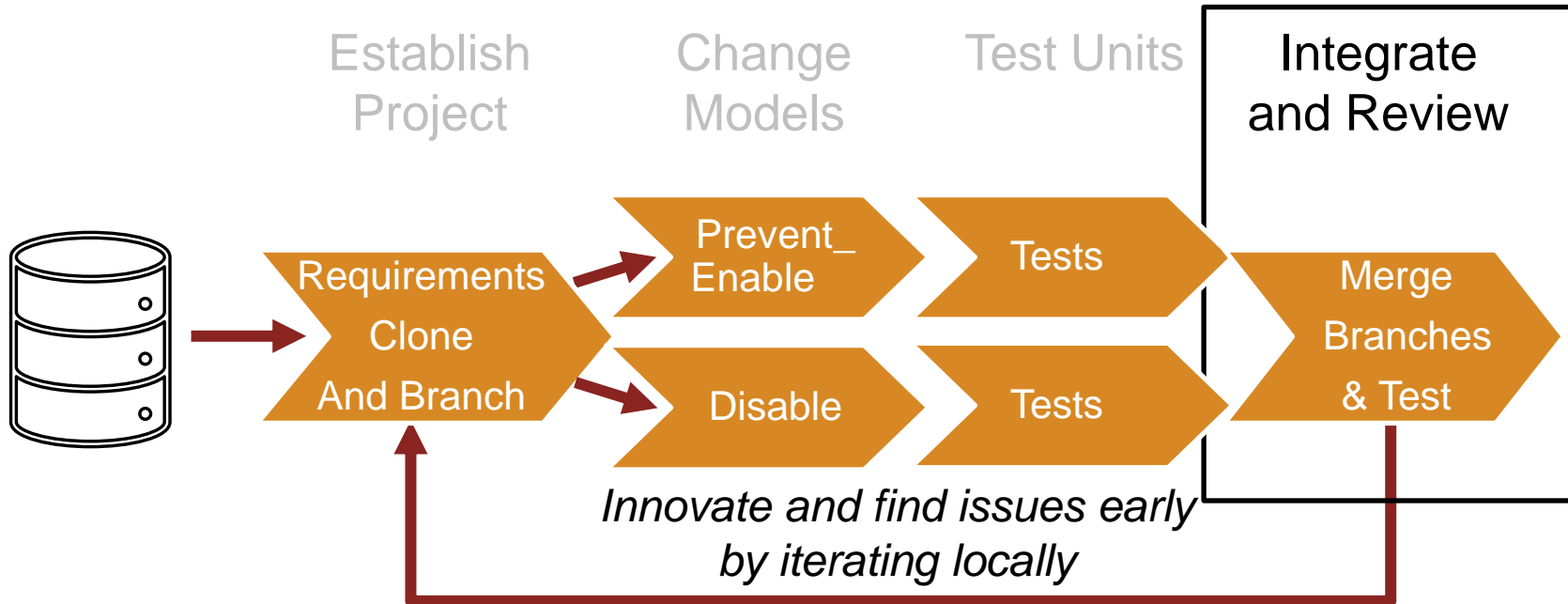
Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Complete the Disable branch



Takeaway #1: Iterate locally to innovate and prequalify

Cruise Control Case Study: Integrate and Review



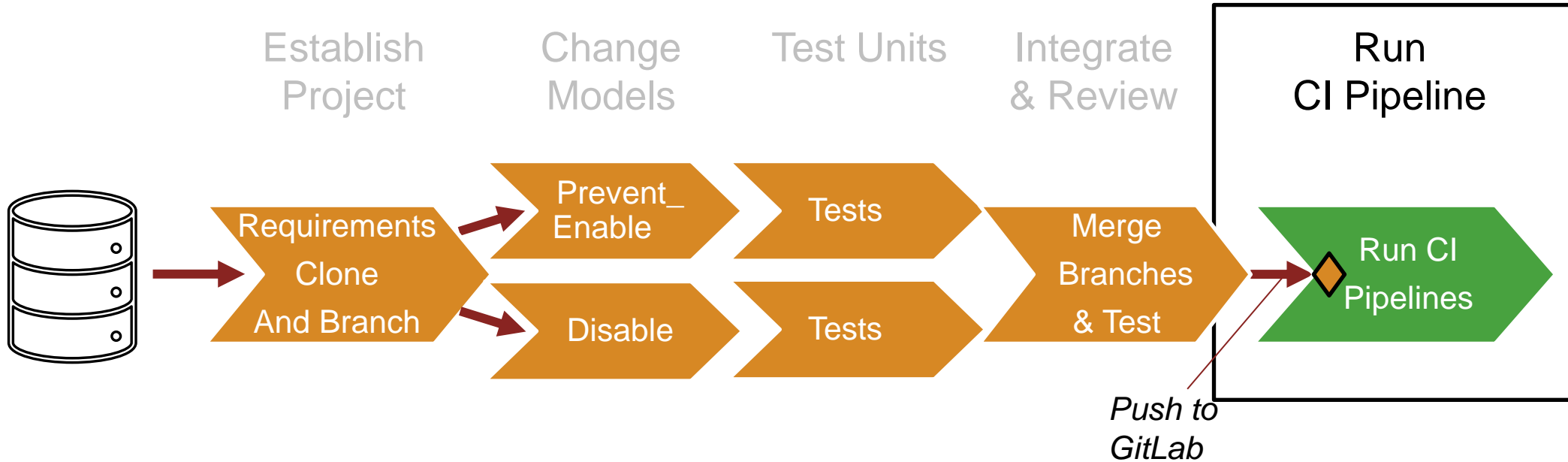
The image shows the MATLAB R2022b interface with the following components:

- Top Bar:** Includes tabs for HOME, PLOTS, APPS, PROJECT, and PROJECT SHORTCUTS. A search bar for documentation is on the right.
- Toolbars:** Contains icons for file operations (New, Open, Share), project tools (Dependency Analyzer, Class Diagram, Model Testing Dashboard), environment management (References, Project Path, Startup/Shutdown), and source control (Git, Refresh, Commit, Fetch, Push, Pull, Remote, Branches, Stashes).
- Project Path:** Shows the current location: C:\Users\bjohnson\Documents\Work\Demos\Projects\CICD\Working_CI_Dir\Continuous-Integration-Verification-Simulink-Models-GitLab.
- Project Explorer:** Displays the file structure for 'Project - CruiseControlExample'. It lists folders (Code, Data, Design, Requirements, resources, tools) and files (.crs_controller-gitlab-ci.yml, .cruiseControlMode-gitlab-ci.yml, .driverSwRequest-gitlab-ci.yml, .gitattributes, .gitignore, .gitlab-ci.yml, .targetSpeedThrottle-gitlab-ci.yml, CruiseControlExample.prj, license.txt, readme.md, SECURITY.md). A 'Git' column shows the status of each file.
- Command Window:** Shows the prompt `fx >>`.
- Workspace:** A vertical panel on the right side of the interface.

I

Takeaway #2: Extend Model-Based Design Workflows into CI Platforms

Cruise Control Case Study: Run CI Pipeline



C:\Users\bjohnson\Documents\Work\Demos\Projects\CICD\Safe\Jem\CICD_script_MAC_2.mlx

LIVE EDITOR INSERT VIEW

New Open Save Compare Print Export FILE

Go To Find Bookmark NAVIGATE

Text **B I U M** TEXT

Code Control Task CODE

Refactor Run Section Run and Advance Run to End SECTION

Pause Step Stop RUN

Push the integrated Project

```
129 Branch="Disable";  
130 Git_Cmd_Pre="git checkout ";  
131 Branch_to  
132 Assure_Runner_Running;  
133 Arg='git push -u -f origin';  
134 Arg=strcat(Arg,{' '},Branch);  
135 system(Arg)
```

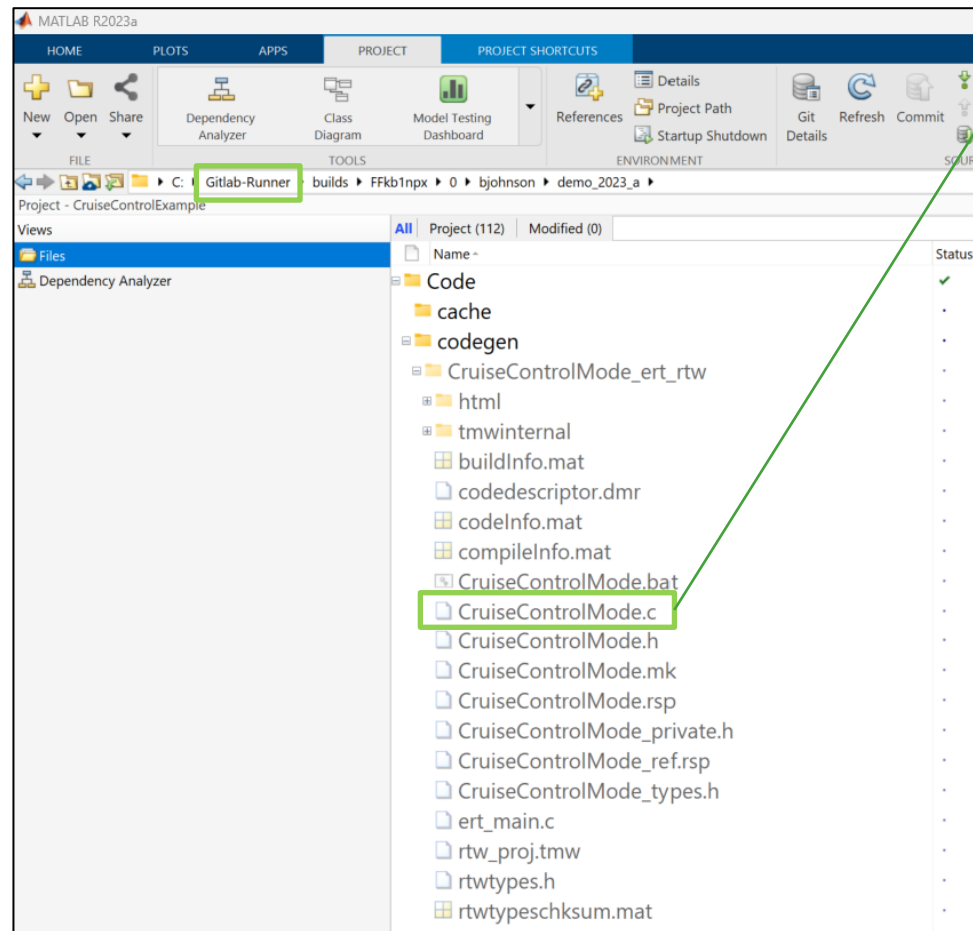
Zoom: 250% UTF-8 LF script

analyzer Startup Shutdown

FILE TOOLS ENVIRONMENT

Review The Code

- Pipeline Outputs are deposited into the Gitlab-Runner Directory
- Consider the CruiseControlMode.c file
- Open it with an Editor
- See that the Driver Awareness change is in place



```

CruiseControlMode.c
File Edit View
/*
 * Prerelease License - for engineering feedback and testing purposes
 * only. Not for sale.
 *
 * File: CruiseControlMode.c
 *
 * Code generated for Simulink model 'CruiseControlMode'.
 *
 * Model version           : 9.0
 * Simulink Coder version  : 23.2 (R2023b) 19-May-2023
 * C/C++ source code generated on : Thu Jul 13 10:13:58 2023
 *
 * Target selection: ert.tlc
 * Embedded hardware selection: Intel->x86-64 (Windows64)
 * Code generation objectives: Unspecified
 * Validation result: Not run
 */

#include "CruiseControlMode.h"
#include "CruiseControlMode_types.h"
#include "rtwtypes.h"
#include <string.h>

/* Block states (default storage) */
DW_CruiseControlMode_T CruiseControlMode_DW;

/* External inputs (root inport signals with default storage) */
ExtU_CruiseControlMode_T CruiseControlMode_U;

/* External outputs (root outports fed by signals with default storage) */
ExtY_CruiseControlMode_T CruiseControlMode_Y;

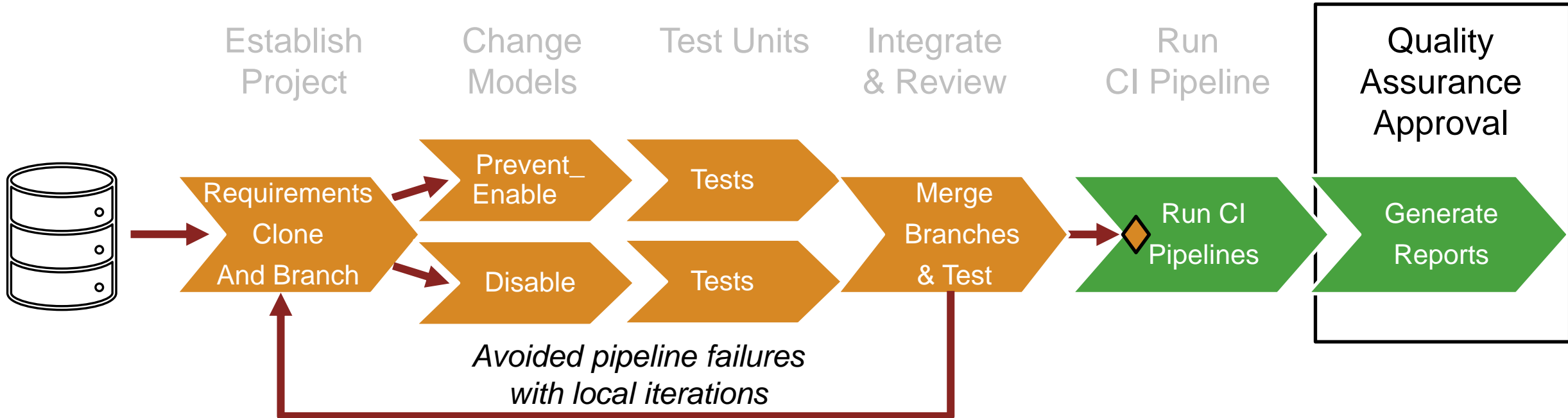
/* Real-time model */
static RT_MODEL_CruiseControlMode_T CruiseControlMode_M;
RT_MODEL_CruiseControlMode_T *const CruiseControlMode_M = &CruiseControlMode_M;

/* Model step function */
void CruiseControlMode_step(void)
{
    boolean_T rtb_Switch1_d;
    boolean_T tmp;
    opMode rtb_Switch;
    reqMode rtb_Switch1;

    /* Outputs for Atomic SubSystem: '<Root>/opMode'
     *
     * Block requirements for '<Root>/opMode':
     * 1. Operation mode determination
     */
    /* Switch: '<S4>/Switch1' incorporates:
     * Constant: '<S18>/Constant'
     * Inport: '<Root>/Driver_Awareness_Input'
     * Inport: '<Root>/reqDrv
  
```

Takeaway #2: Extend Model-Based Design Workflows into CI Platforms

Cruise Control Case Study: Approve Release



- Search GitLab
- Demo_2023_a
- Project information
- Repository
- Issues 0
- Merge requests 0
- CI/CD
- Pipelines**
- Editor
- Jobs
- Schedules
- Security and Compliance
- Deployments
- Packages and registries
- Infrastructure
- Monitor
- Analytics
- Wiki
- Snippets
- Settings

Bernard Johnson > Demo_2023_a > Pipelines

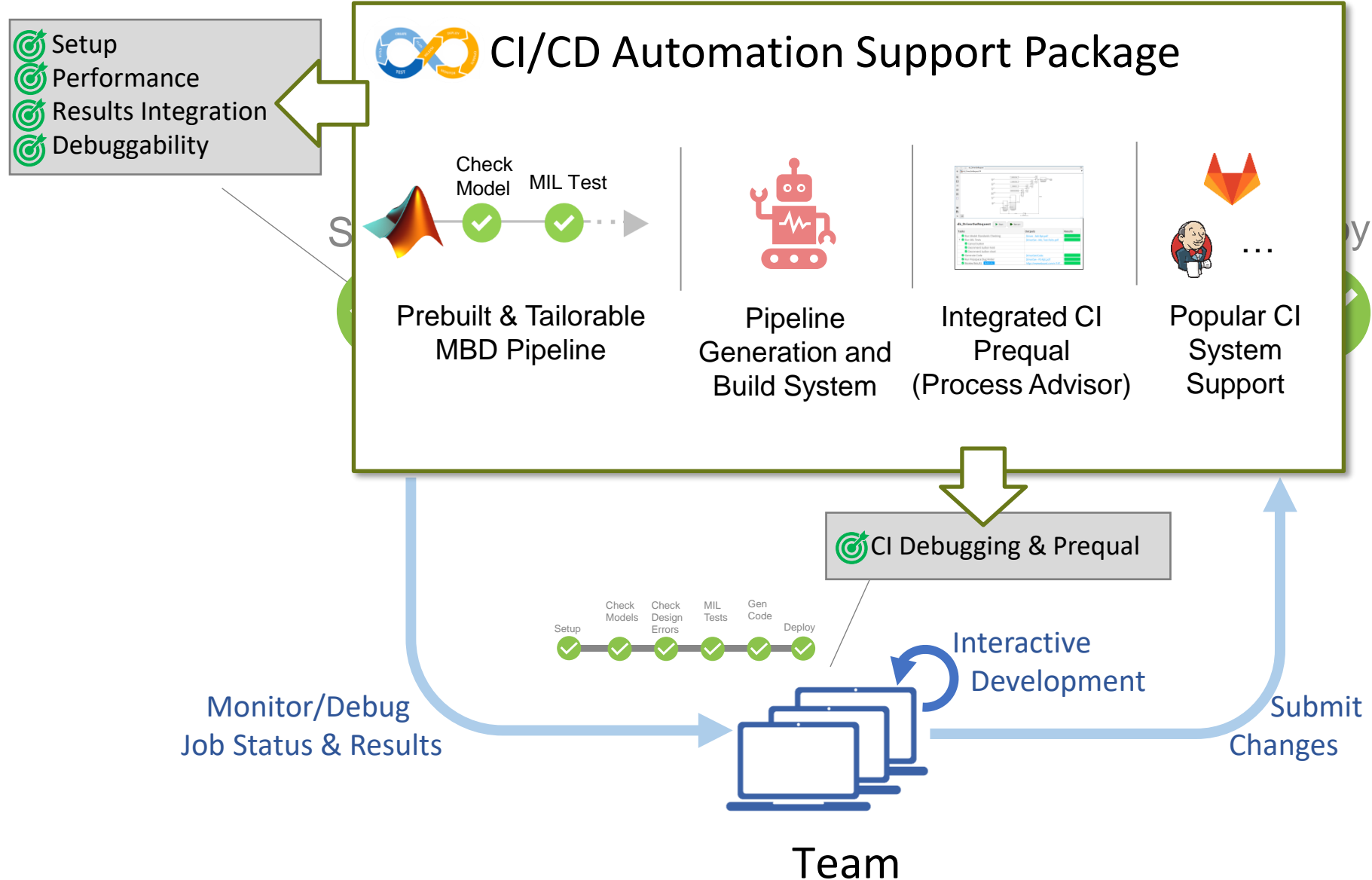
All 108 Finished Branches Tags

Clear runner caches CI lint Run pipeline

Filter pipelines [] Show Pipeline ID []

Status	Pipeline	Triggerer	Stages	
passed 00:08:43 just now	"Driver_Awareness Final Release" #87380 Disable -> 09e30afa latest	[Avatar]	✓ → ✓	Download
skipped » Skipped	Reset Project without running pipeline #87379 main -> 18f658e2 latest	[Avatar]		Download
canceled 00:01:03 16 minutes ago	"Driver_Awareness Final Release" #87374 Disable -> 5a17ca0b	[Avatar]	⊘ → ⊘ ⊘ ⊘ ⊘ +1	Download
passed 00:14:29 22 hours ago	"Driver_Awareness Final Release" #87200 Disable -> 7ad9c90a	[Avatar]	✓ → ✓	Download
canceled 00:00:42 23 hours ago	"Driver_Awareness Final Release" #87191 Disable -> e8706b03	[Avatar]	⊘ → ⊘ ⊘ ⊘ ⊘ +1	Download
skipped » Skipped	Reset Project without running pipeline #87187 main -> 1e08b360	[Avatar]		Download
skipped » Skipped	Reset Project without running pipeline #87169 main -> d8089cc5	[Avatar]		Download
skipped » Skipped	Reset Project without running pipeline #87150 main -> b5d9ba55	[Avatar]		Download
skipped » Skipped	Reset Project without running pipeline #86702 main -> 5f190883	[Avatar]		Download
skipped » Skipped	Reset Project without running pipeline #86695 main -> 01e88f48	[Avatar]		Download

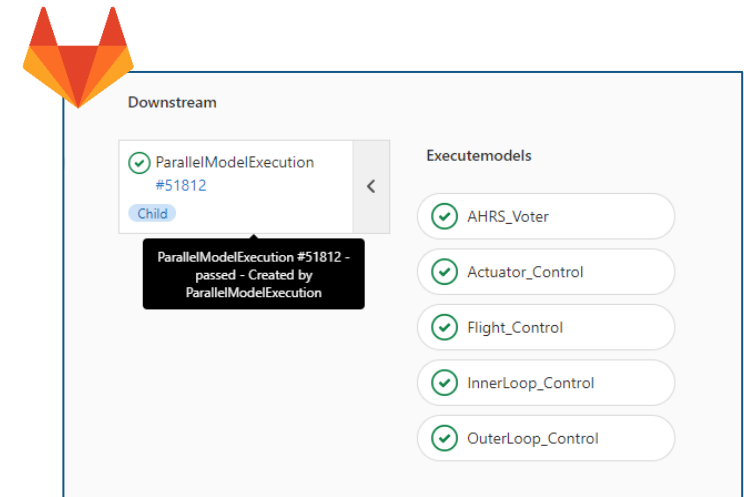
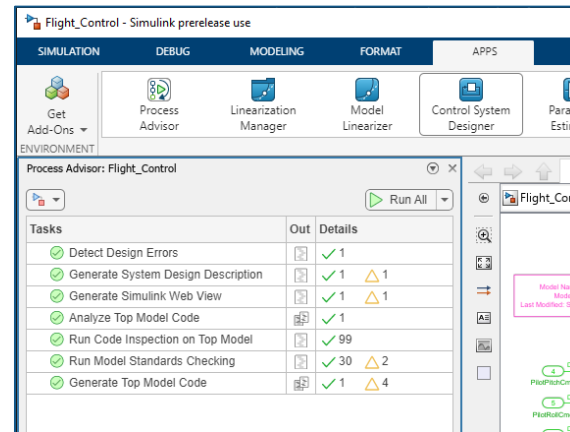
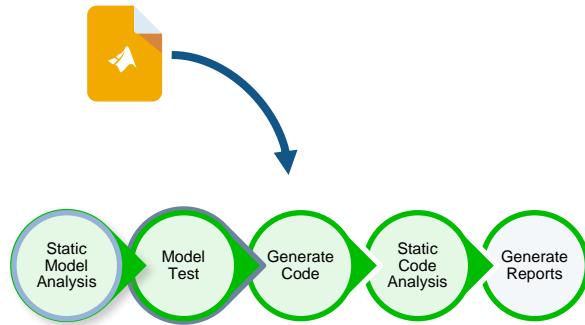
Simplifying Adoption and Optimizing CI/CD for Model-Based Design



CI/CD Automation for Simulink Check Support Package

R2022a

August



1) Simple Setup

- ✓ Prebuilt Model-Based Design pipeline
- ✓ Built-in Model-Based Design tool support
- ✓ Tailorable

2) Desktop Integration with Process Advisor app

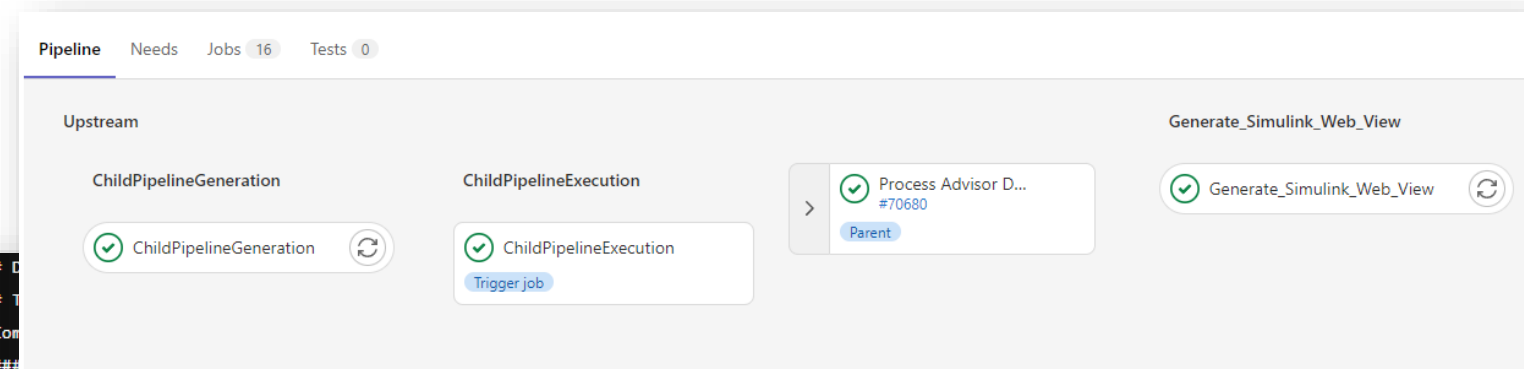
- ✓ Local prequalification
- ✓ Local Debugging

3) 3rd Party CI Integration

- ✓ Jenkins/Gitlab YAML
- ✓ Optimized Model-Based Design Builds
- ✓ CI Results Integration

Integration with common CI Systems

- Automated Pipeline Generation
- Publish Results
- Debug on Desktop



```

181 #### D
182 #### T
183 ## Co
184 #####
185 #####
186 ## Ending Process Advisor build at 29-Nov-2022 08:00:16
187 #### Duration:          00:05:45
188 #### Build Status:      Pass
189 #### Number of tasks:    5
190 #### Number of tasks executed: 5
191 #### Number of tasks skipped: 0
192 #### Number of tasks in queue: 0
193 #### Number of tasks failed: 0
194 #####
195 Saving cache for successful job
  
```

Process Advisor - db_Controller

Run All

Tasks	Out	Results
<ul style="list-style-type: none"> Run Code Generator <ul style="list-style-type: none"> db_ControlMode db_Controller db_DriverSwRequest db_TargetSpeedThrottle Run Model Standards Checking <ul style="list-style-type: none"> db_ControlMode db_Controller db_DriverSwRequest db_TargetSpeedThrottle Run Design Error Detection 	<ul style="list-style-type: none"> db_ControlMode db_Controller db_DriverSwRequest db_TargetSpeedThrottle db_ControlMode db_Controller db_DriverSwRequest db_TargetSpeedThrottle 	<ul style="list-style-type: none"> 20 ✓ 2 ✗ 3 ⚠ 10 ✓ 1 ✗ 1 ⚠ 2 ✓ 3 ✓ 0 ✗ 0 ⚠ 5 ✓ 1 ✗ 0 ⚠ 60 ✓ 1 ✗ 5 ⚠ 15 ✓ 0 ✗ 4 ⚠ 15 ✓ 15 ✓ 1 ✗ 0 ⚠ 15 ✓ 0 ✗ 1 ⚠ 93 ✓ 7 ✗ 15 ⚠

A few Continuous Integration success stories in Automotive

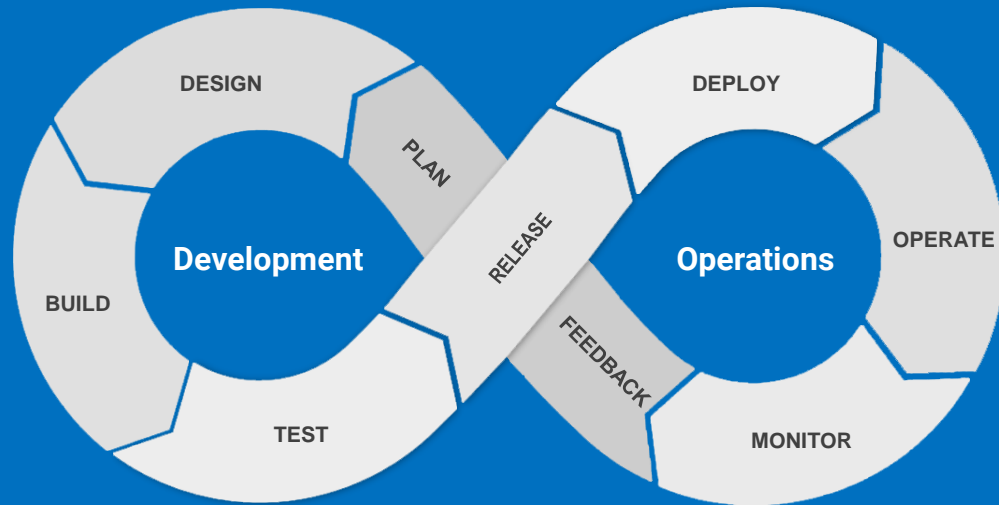


[A123 link: Similar case study, but uses Jenkins](#)



[HL Klemove link: Focus on Polyspace](#)

Summary



Model DevOps



Engineering, Data Science, IT, and Operations teams must collaborate to ensure success



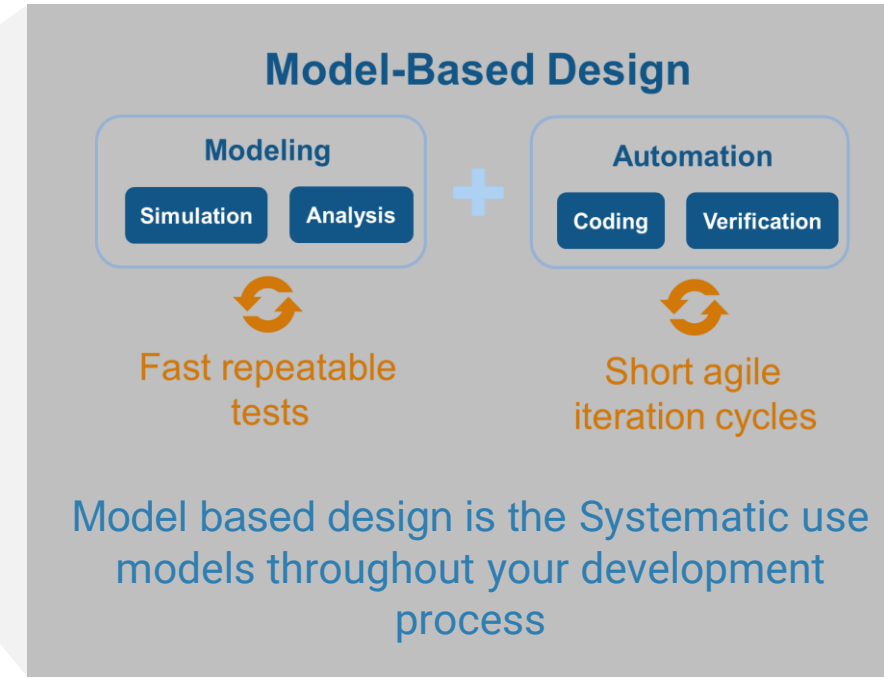
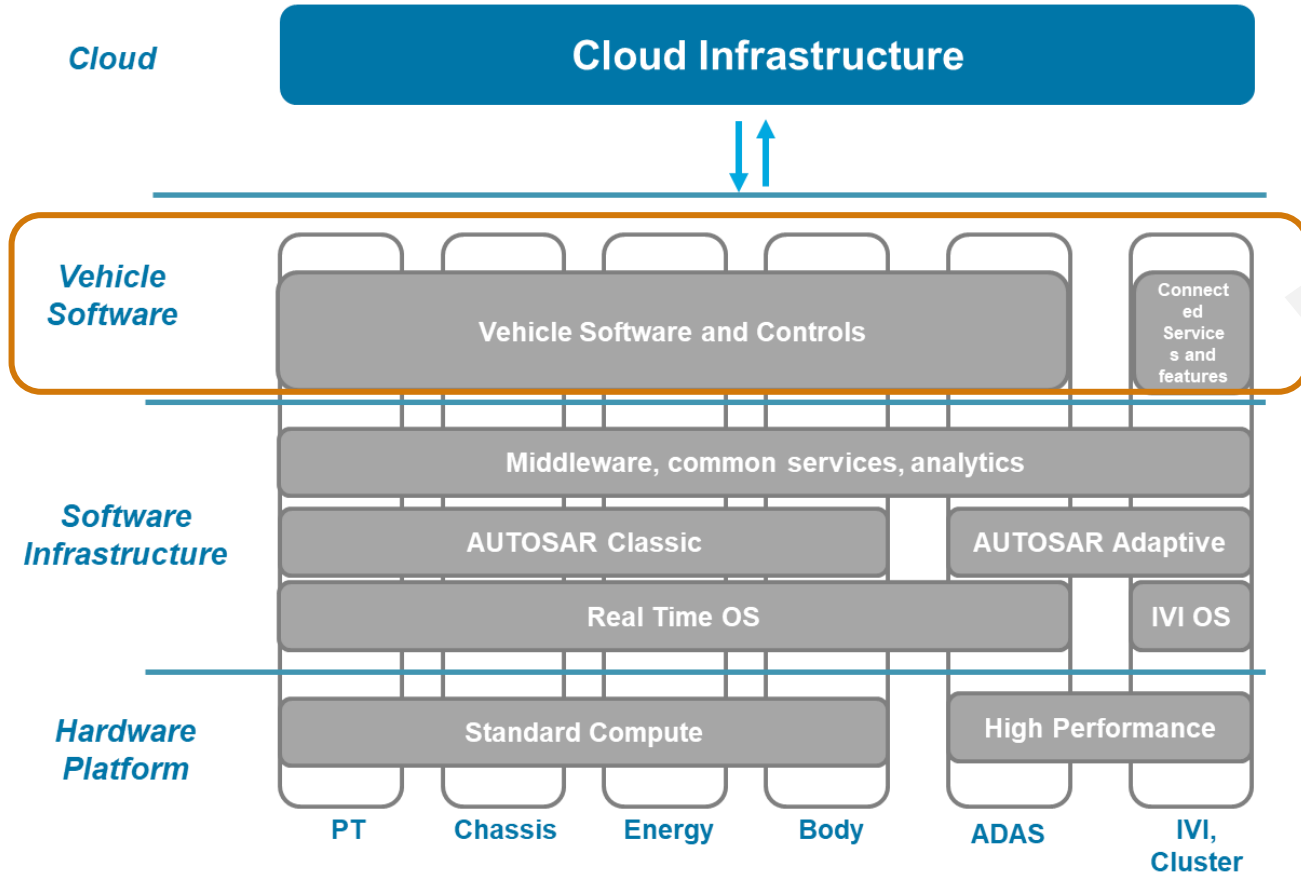
MATLAB & Simulink can be integrated into your development environment and leverage data from a variety of data sources



MATLAB & Simulink models can be deployed into a variety of platforms: embedded, edge, IT/OT, and cloud

Learn more: [Continuous Integration for Model-Based Design](#)

Modeling and Automation for Software Defined Vehicle Applications



MathWorks
**AUTOMOTIVE
CONFERENCE 2023**
India

Thank you

