2022 MathWorks
中国汽车年会

基于模型的持续集成加速车载软件开发和交付

胡乐华, MathWorks中国

MathWorks®

内容提要

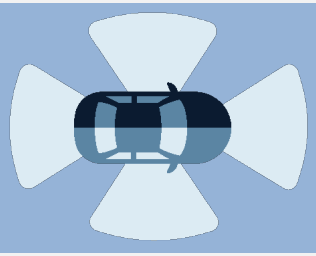
车载软件系统开发新挑战

基于模型设计用于CI的流程

云端部署CI系统用于大规模系统

车载软件系统开发新挑战

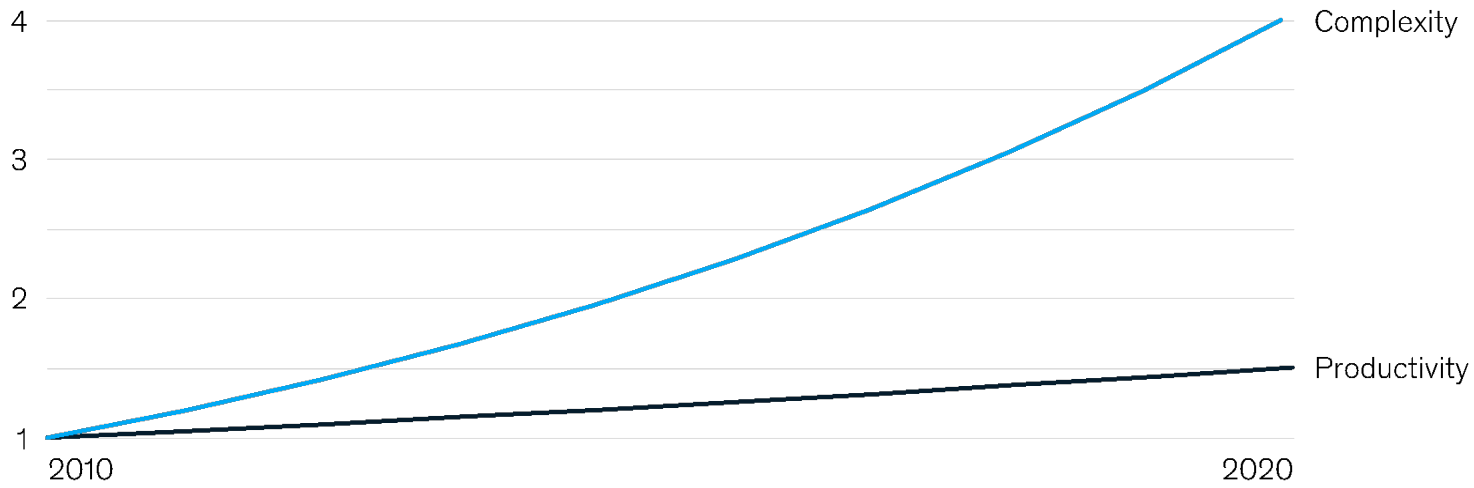
汽车代码量增长迅猛

1980	2000	2020	Future
			
10s of KLOC	100s of K	100s of M	>500M
~5 years / release	3 – 5 years	0.4 – 1 year	On-demand

数字化转型

软件复杂度的增长远超开发效率

Relative growth of software complexity and productivity over time, indexed for automotive features



Source: McKinsey's SoftCoster embedded software project database

Source: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/when-code-is-king-mastering-automotive-software-excellence

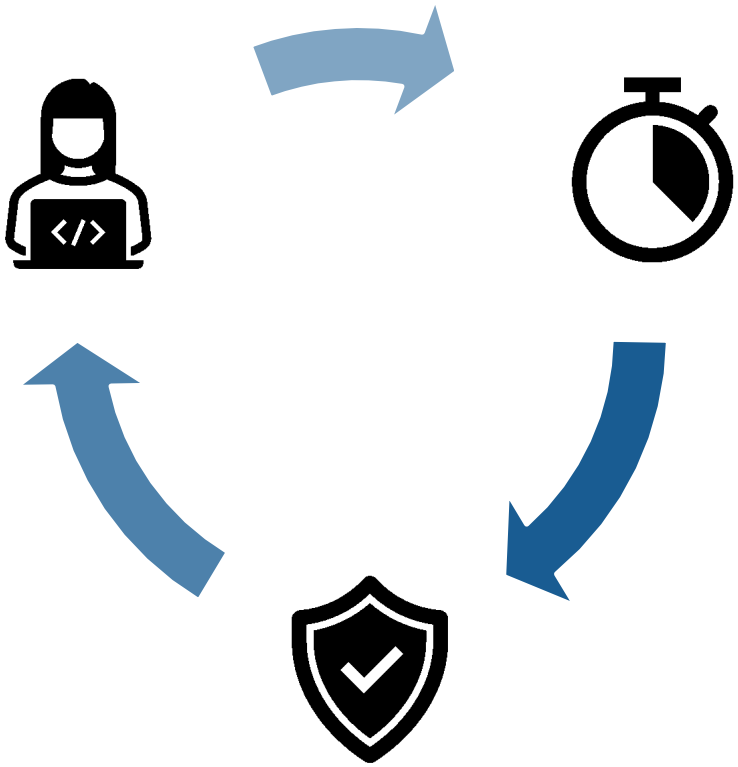
车载嵌入式系统开发面临更多挑战

Complexity and Size

Low Productivity

C++, AUTOSAR

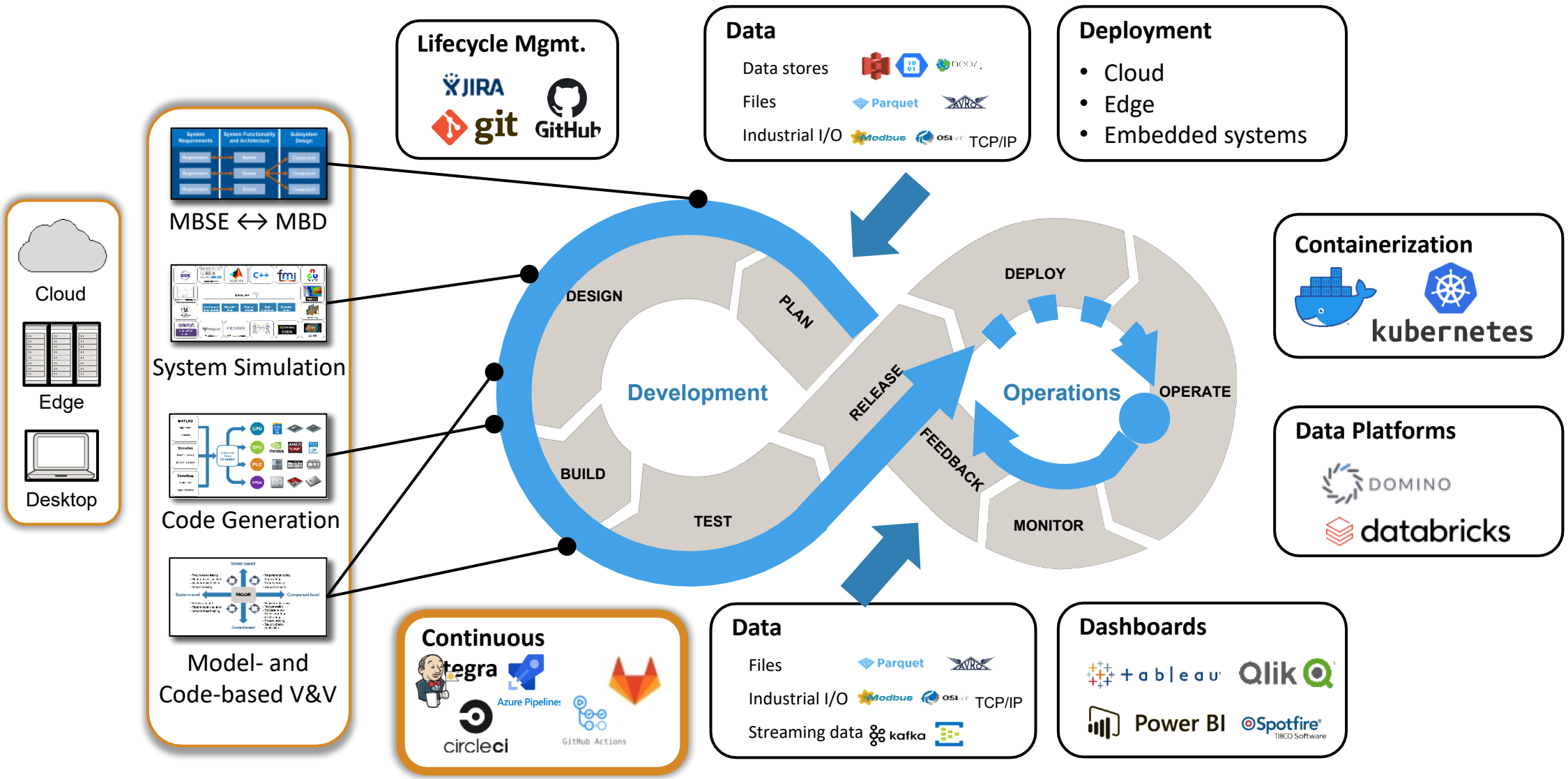
Staffing SW developers



Faster delivery

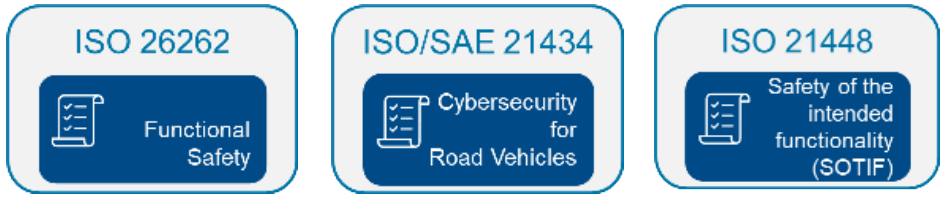
Safety and Cybersecurity
ISO 26262, ISO/SAE 21434

持续集成是 DevOps 的重要环节



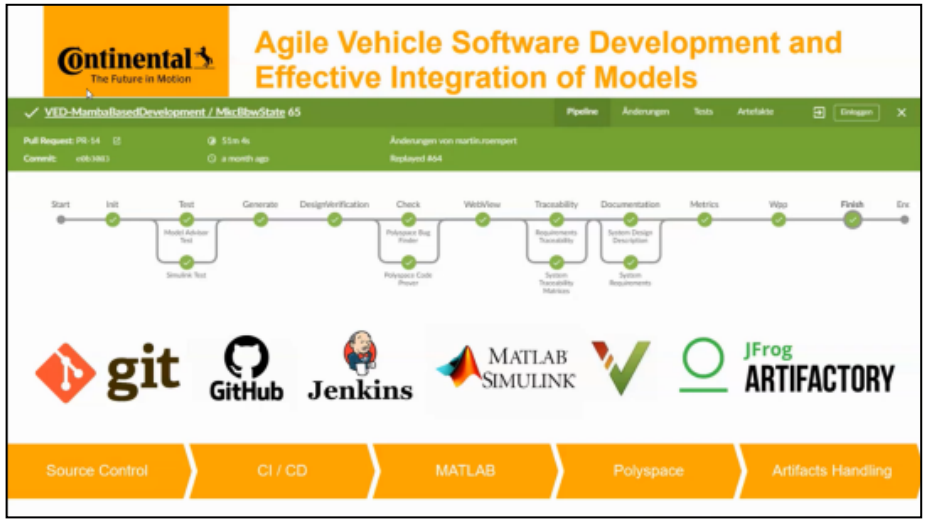
基于模型设计促进高效的 DevOps

- DevOps 目标: 缩短提交代码到产品更新的时间，并保证高质量、高安全和安全标准的合规性



	High Performers	Low Performers
Lead Time	**<1hour**	**>6months**
Change Failure Rate	**0-5%**	**15-30%**

Source: [state-of-devops-2021.pdf (google.com)]



Online Panel Discussion: [Agile Vehicle Software Development and Effective Integration of Models]

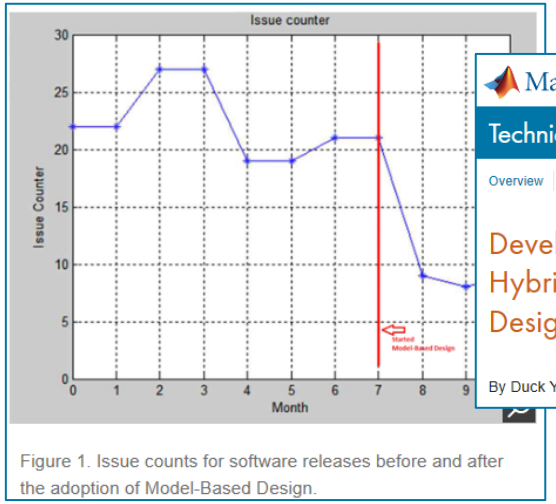
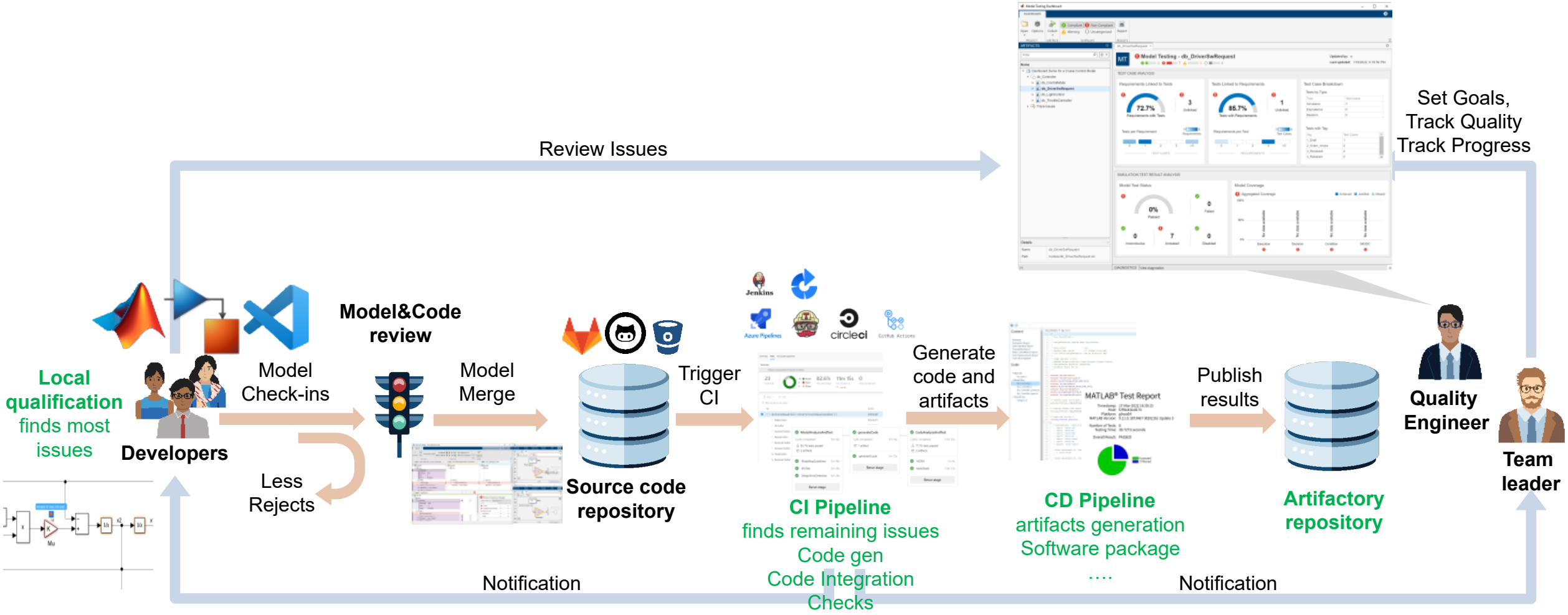


Figure 1. Issue counts for software releases before and after the adoption of Model-Based Design.

[Link to technical article]

基于模型设计的持续集成工作流

将基于模型设计融入 CI 系统

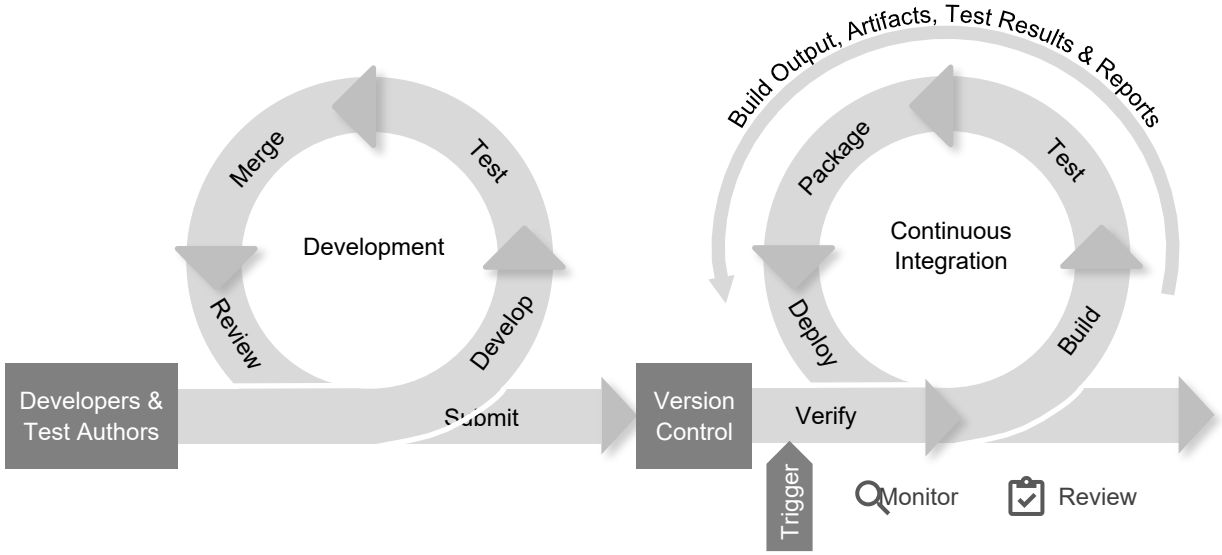


Direct Feedback

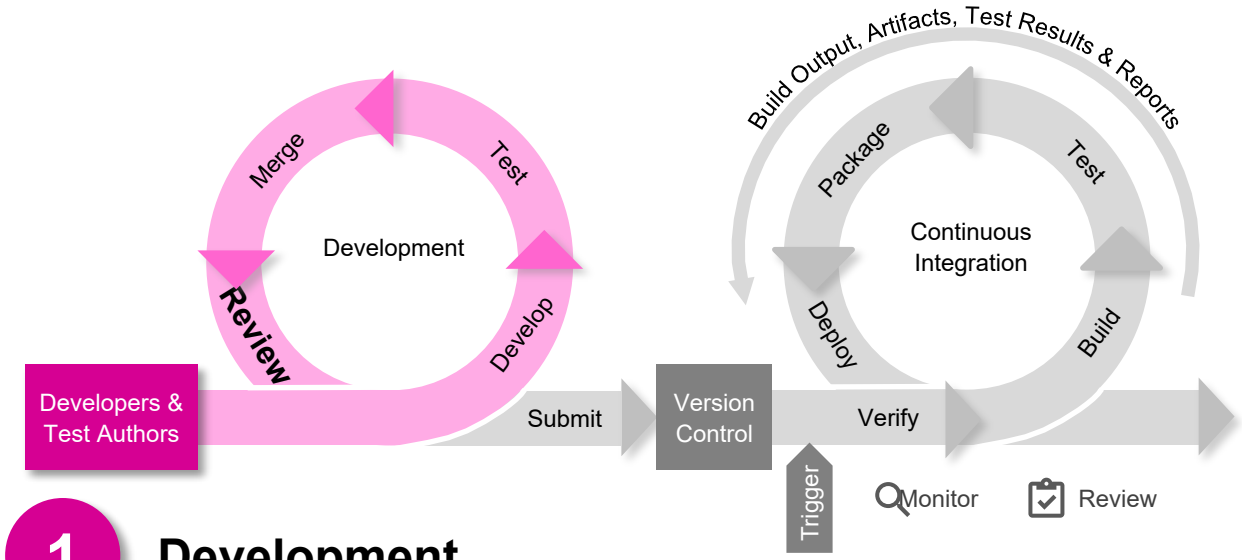
Flow / Automation

Learning

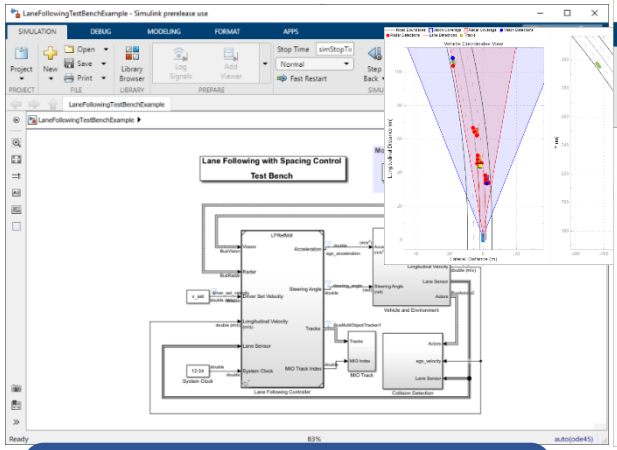
使用 MATLAB 和 Simulink 述说持续集成的故事

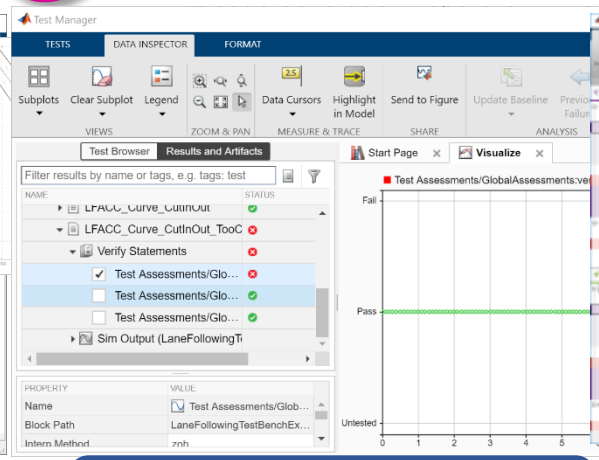


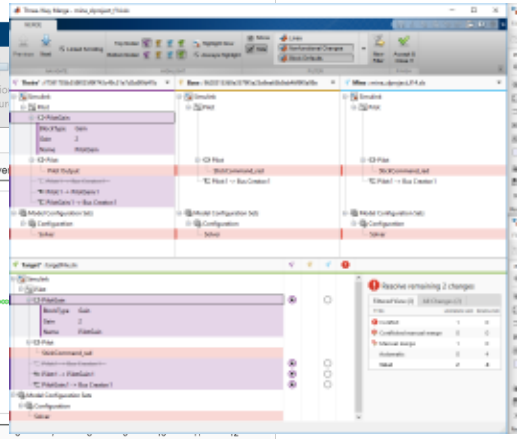
基于模型设计的持续集成工作流
开发阶段

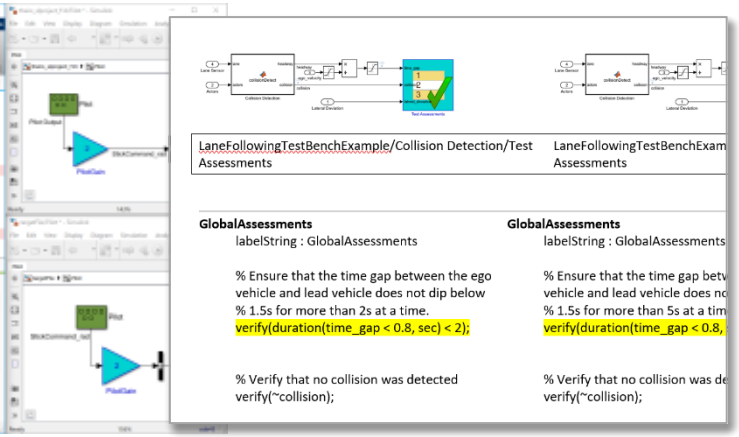


1 Development



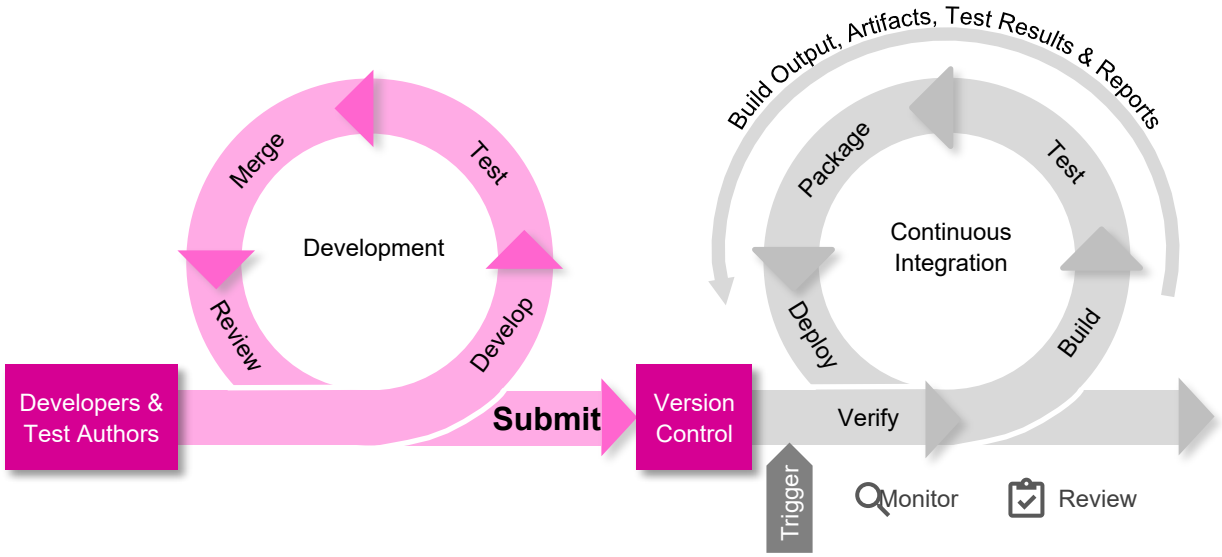


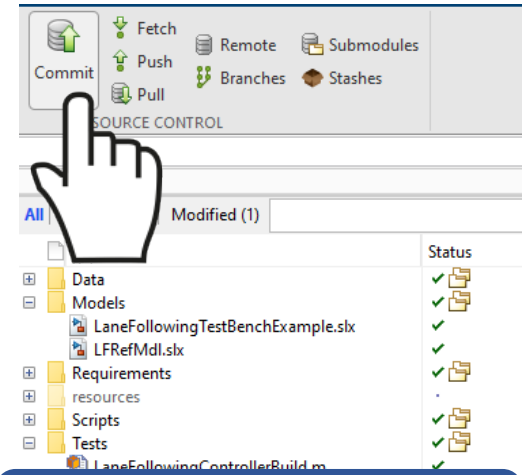




Modelling | *Verification &Test* | *Compare & Merge* | *Gen Reports*

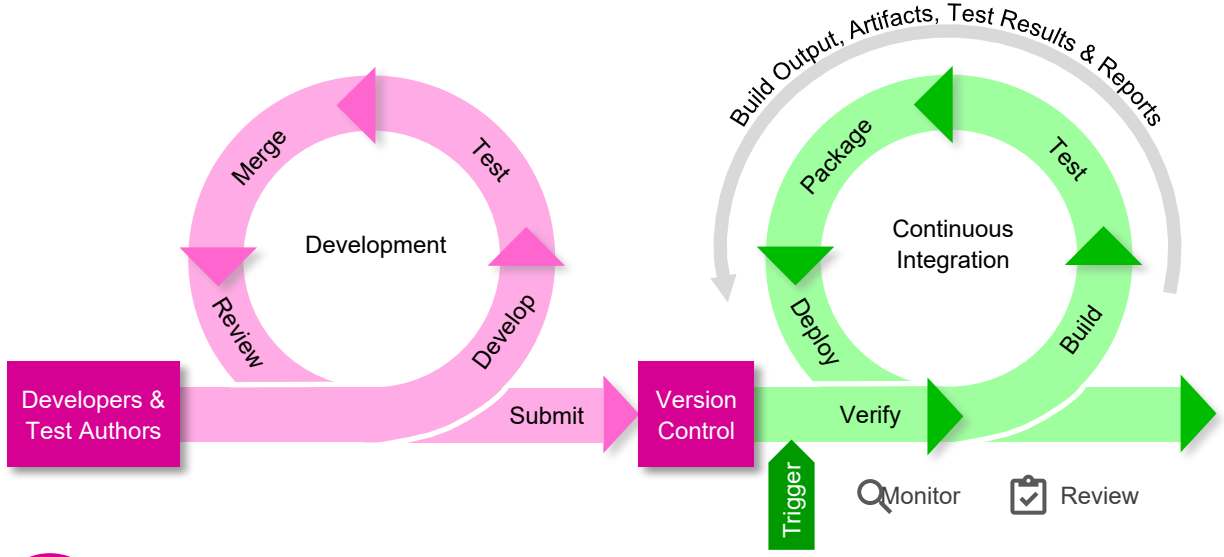
基于模型设计的持续集成工作流推送





Projects with SCM

基于模型设计的持续集成工作流



1 **Development**

2 **Continuous Integration**

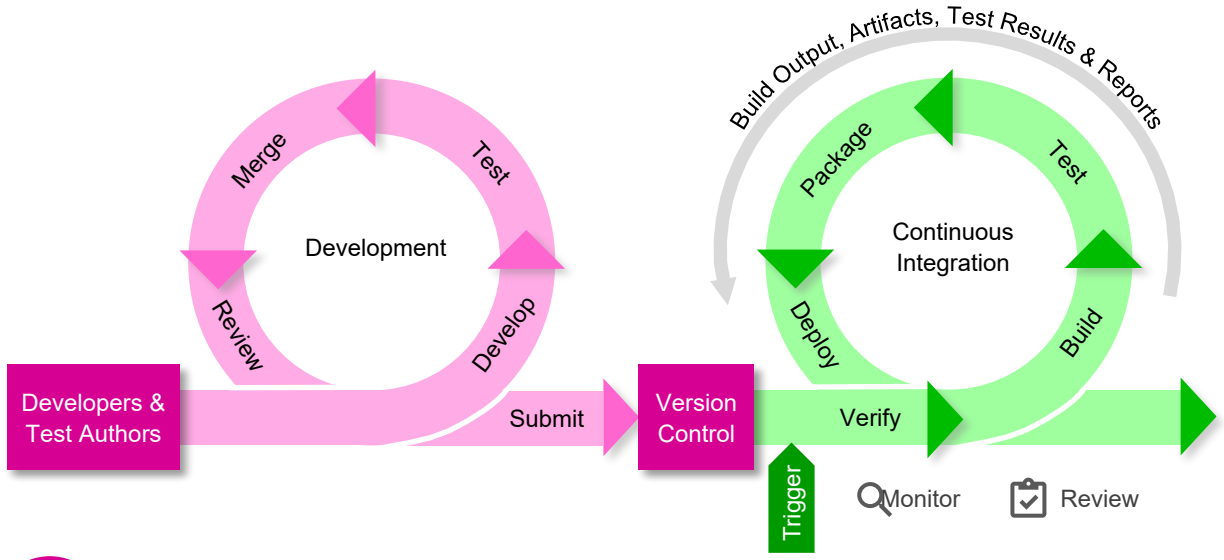
④ ④ **Pipelines** ④ ④

Build **Test** **Package** **Deploy**

Test may include:

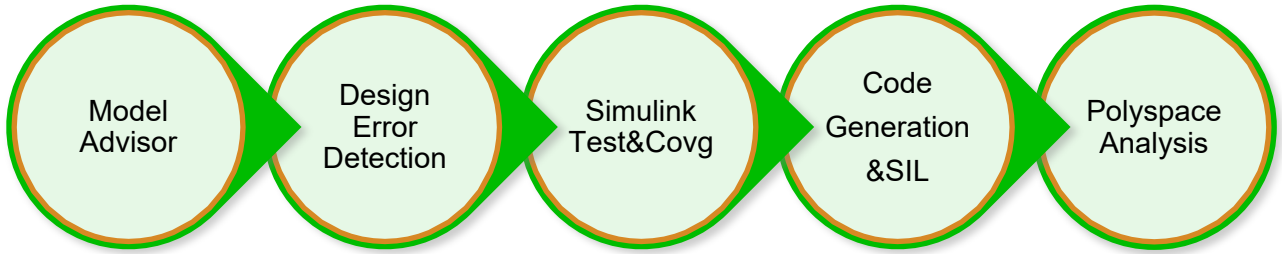
- Dynamic Test
- Static Checks
- Coverage…

基于模型设计的持续集成工作流



1 **Development**

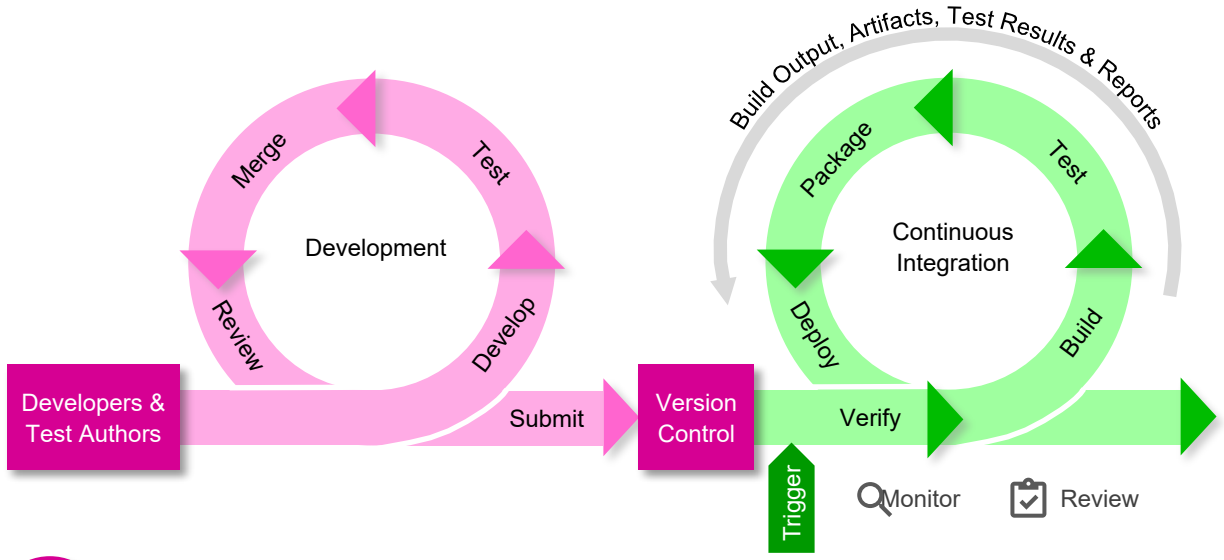
2 **Continuous Integration**



Reports, Build Logs, Test Results,
Code Coverage, Code Compliance, Runtime Errors

Verification, Build and Test

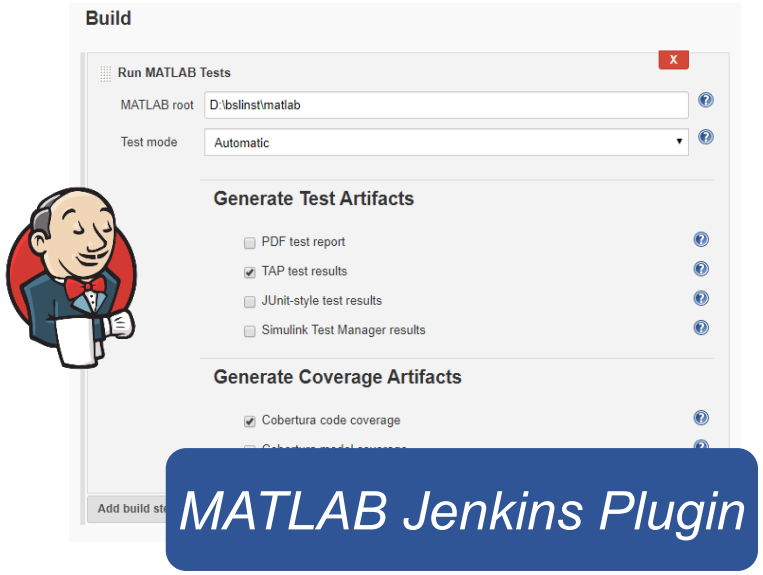
基于模型设计的持续集成工作流



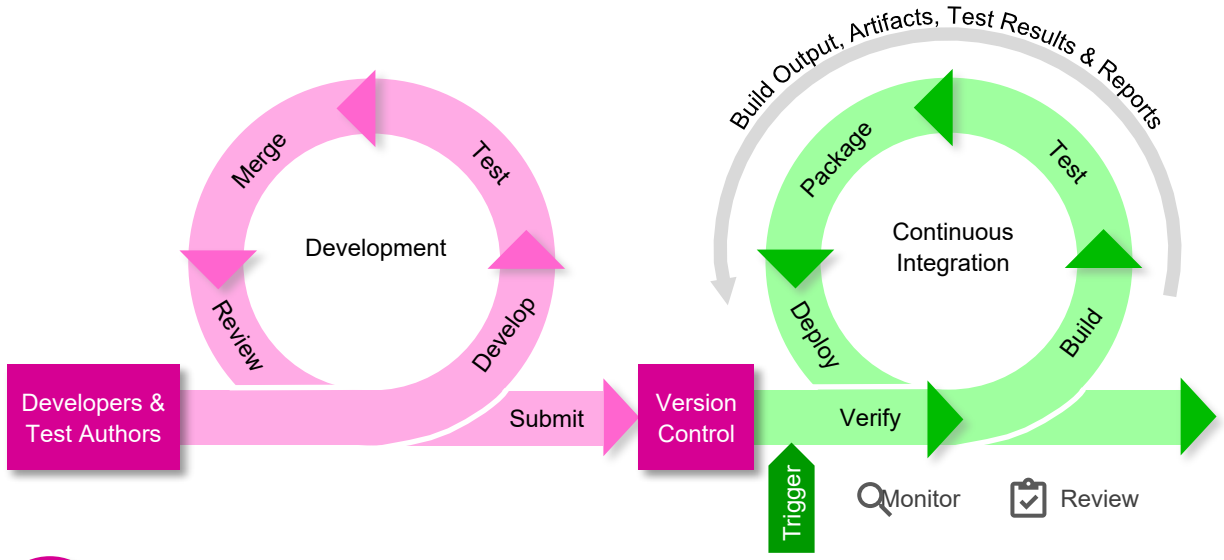
- 安装路径设置
- 过滤测试并运行
- 生成报告和其他产出物
- 提供pipeline接口

1 **Development**

2 **Continuous Integration**



基于模型设计的持续集成工作流



Run any MATLAB scripts/functions using -batch

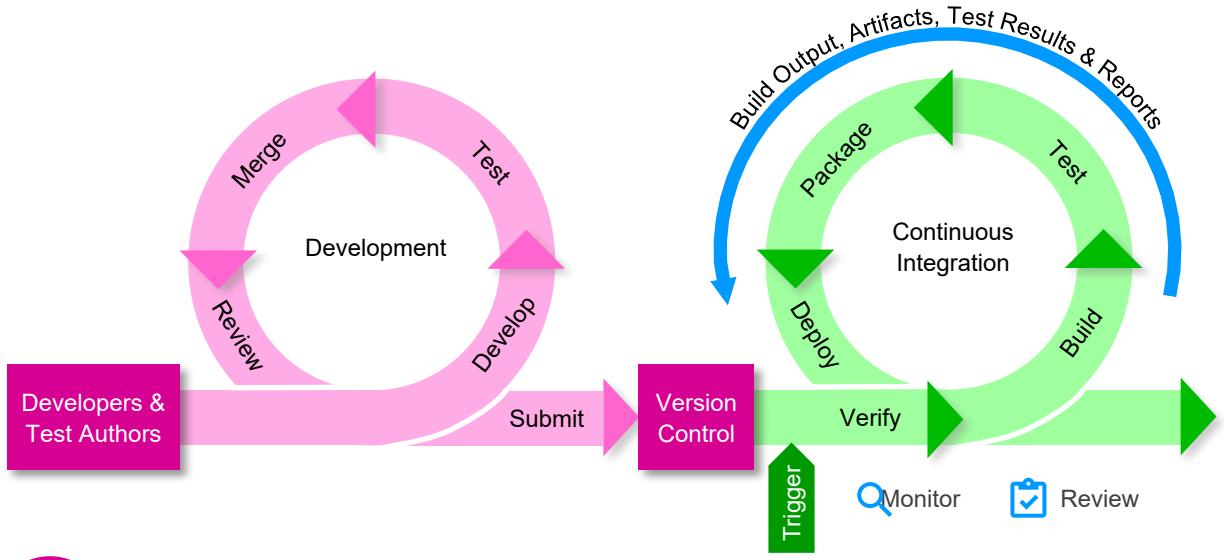
1 **Development**

2 **Continuous Integration**

matlab -batch

MATLAB -batch

基于模型设计的持续集成工作流

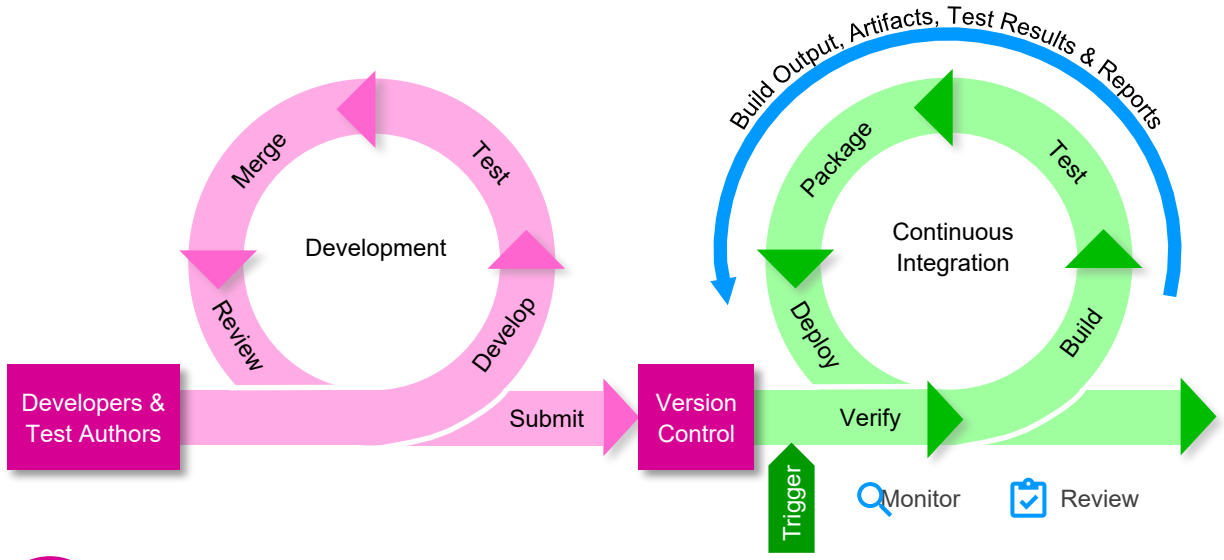


1. **Development**
2. **Continuous Integration**
3. **Results Monitor and Review**

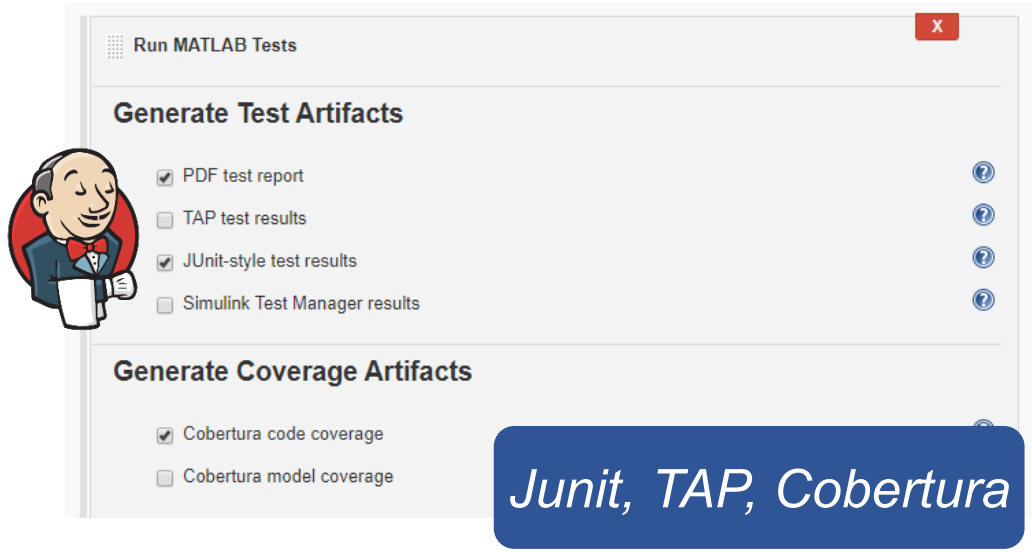
```
Failure Summary:

     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

基于模型设计的持续集成工作流

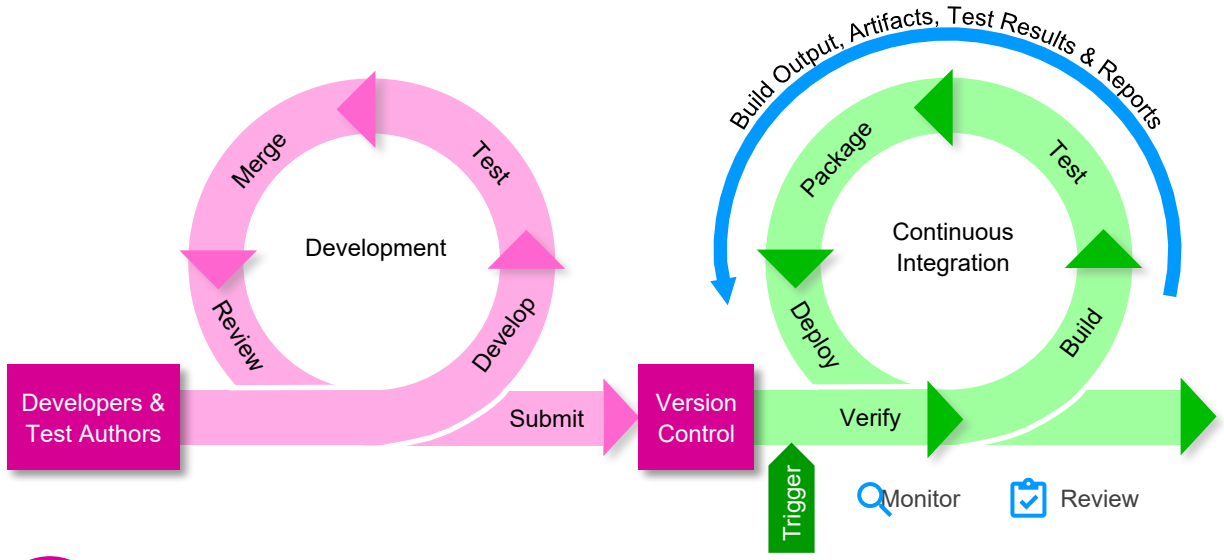


1. **Development**
2. **Continuous Integration**
3. **Results Monitor and Review**

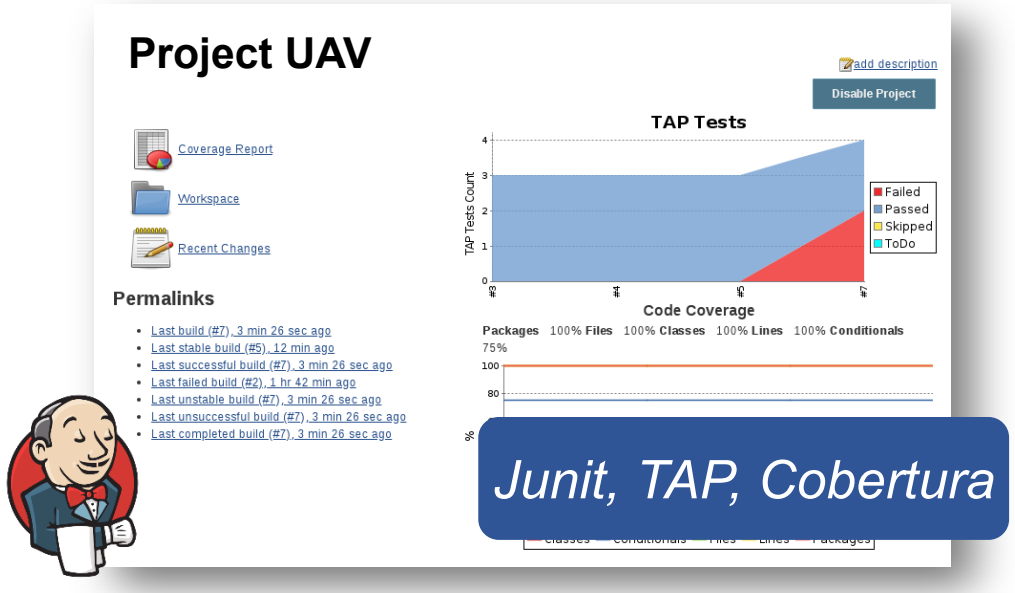


Junit, TAP, Cobertura

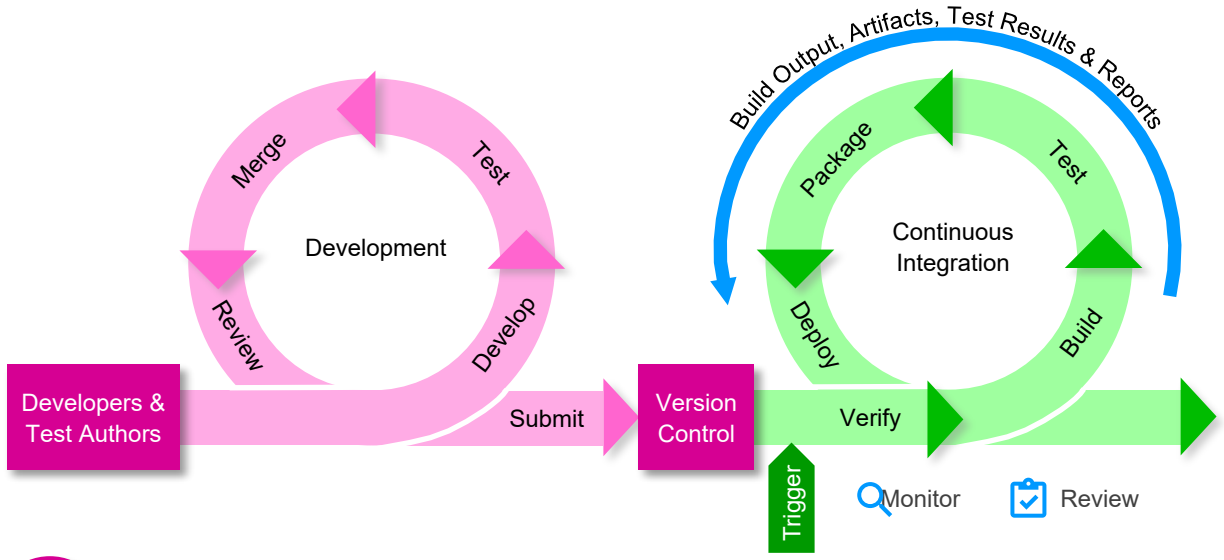
基于模型设计的持续集成工作流



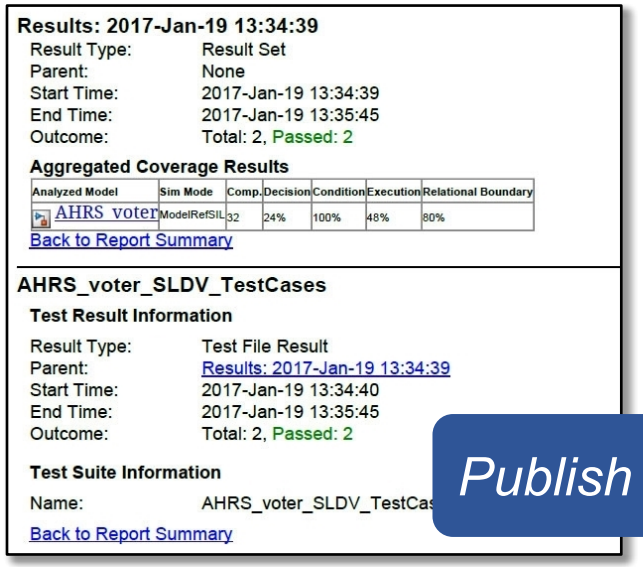
1. **Development**
2. **Continuous Integration**
3. **Results Monitor and Review**



基于模型设计的持续集成工作流

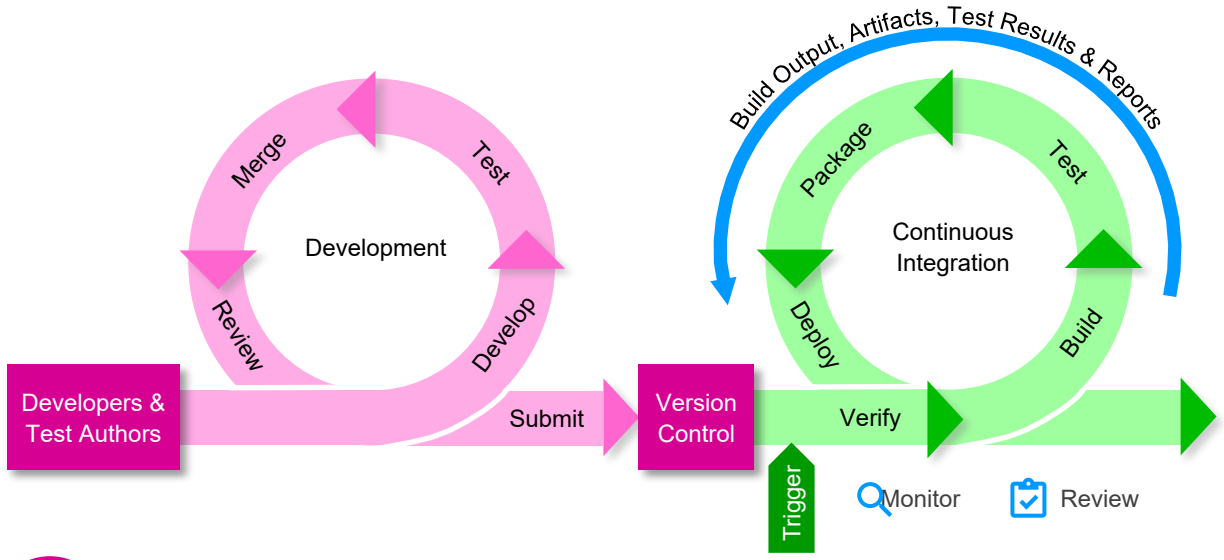


1. **Development**
2. **Continuous Integration**
3. **Results Monitor and Review**

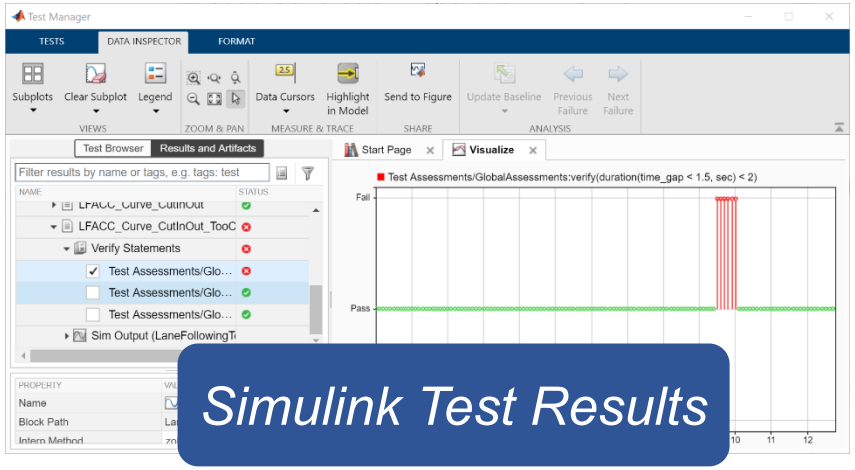


Publish Reports

基于模型设计的持续集成工作流



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

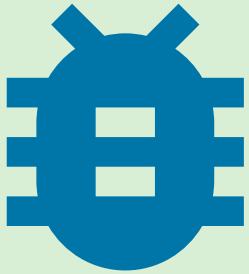


Simulink Test Results

持续集成强化基于模型设计的流程

自动化和快速化

协同合作



提高质量

节约成本

审查&认证就绪

集成 Polyspace 到 CI 系统

Polyspace 静态分析能力

Bug Finder

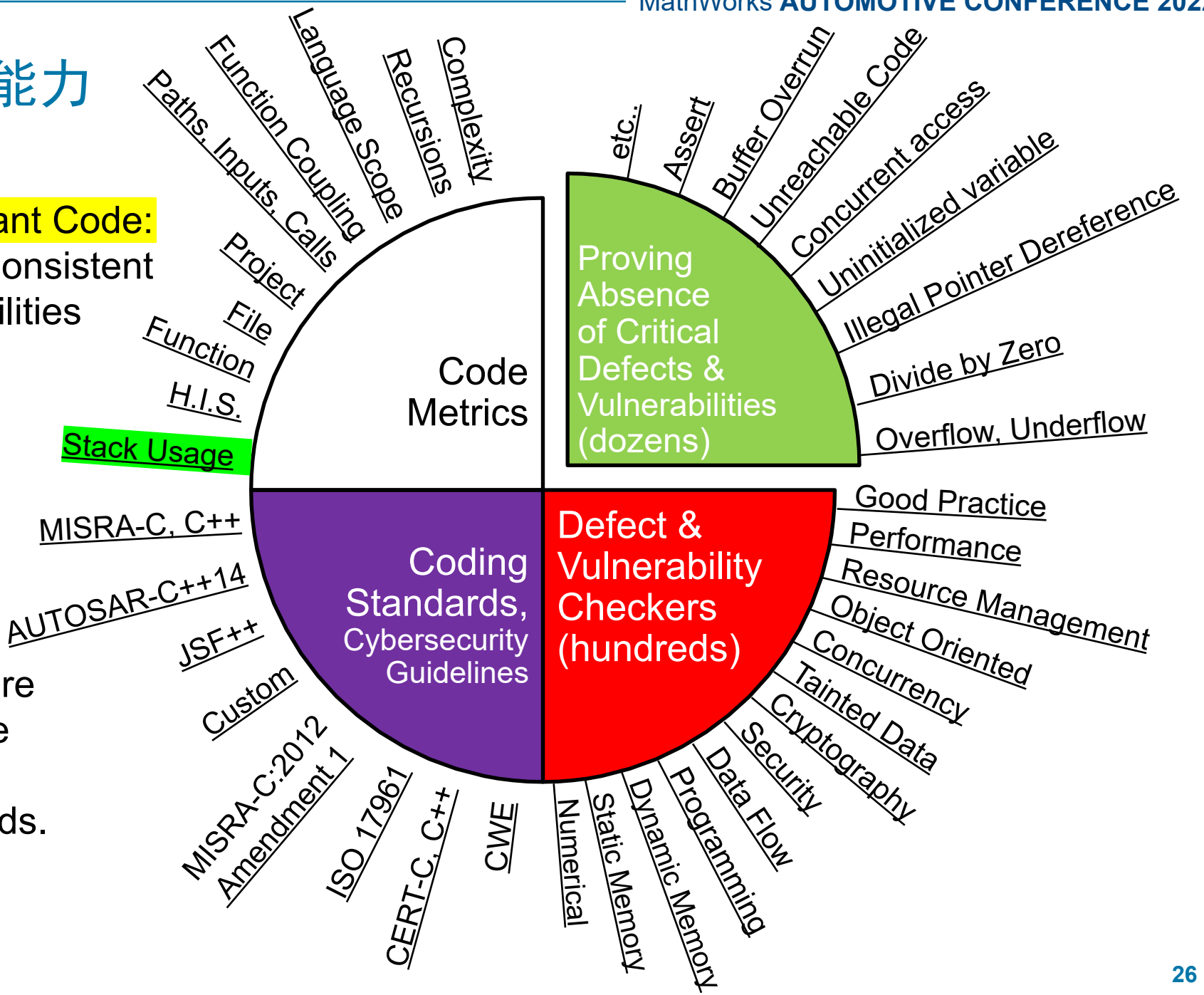
→High Quality, Secure, Compliant Code:

- Measurable, Maintainable, Consistent
- Very few defects or vulnerabilities
- Credits for functional safety, cybersecurity standards.

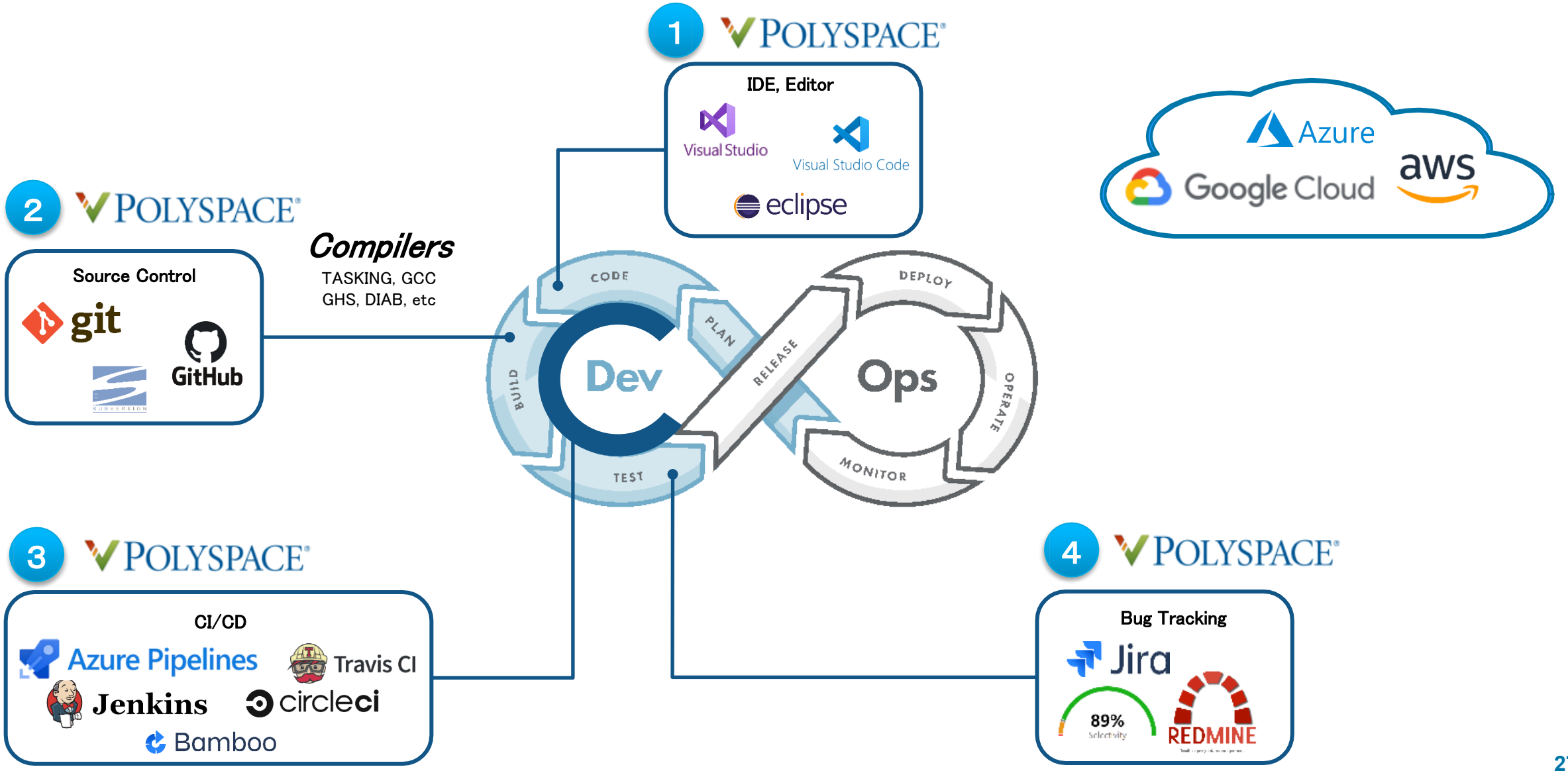
Code Prover

→Fully Trusted Components:

- Reliable, Robust, Safe, Secure
- Proven free of critical runtime defects and vulnerabilities
- Additional credits for standards.



集成到 DevOps 中


1
POLYSPACE
IDE, Editor
Visual Studio
Visual Studio Code
eclipse
2
POLYSPACE
Source Control
git
GitHub
SUBVERSION
Compilers
TASKING, GCC
GHS, DIAB, etc
Azure
Google Cloud
aws
CODE
PLAN
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MONITOR
Dev
Ops
3
POLYSPACE
CI/CD
Azure Pipelines
Travis CI
Jenkins
circleci
Bamboo
4
POLYSPACE
Bug Tracking
Jira
89%
REDMINE

集成到 DevOps 中

1 POLYSPACE

IDE, Editor

Visual Studio

Visual Studio Code

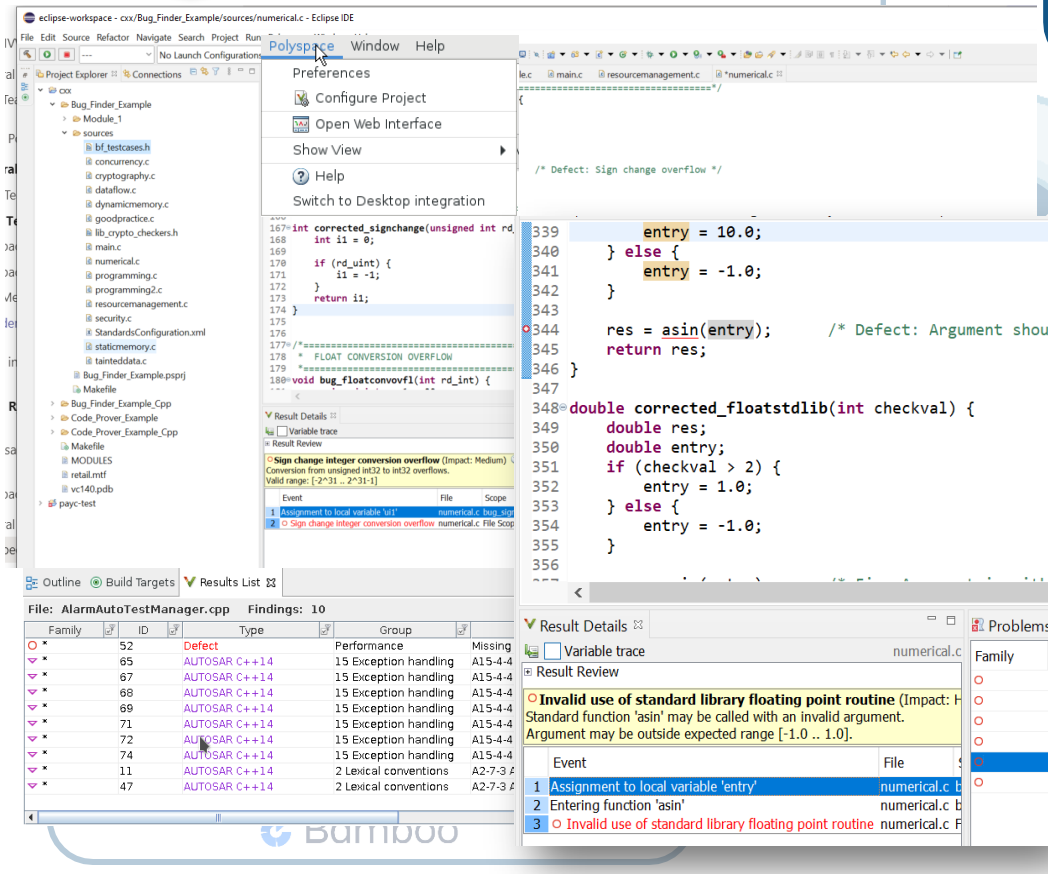
eclipse

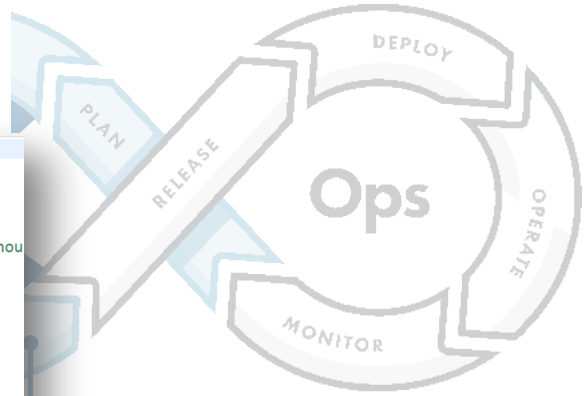
Azure

Google Cloud

aws

2 POLYSPACE





4 POLYSPACE

Bug Tracking

Jira

89%

REDMINE

集成到 DevOps 中

```
polyspace-configure -allow-build-error -allow-overwrite -module -output-options-path options_path make -B

clude/ src/idl_options.proto

-DBTHREAD_USE_FAST_PTHREAD_MUTEX -D__const__= -D_GNU_SOURCE -DUSE_SYMBOLIZE -DNO_TCMALLOC -D__STDC_FORMAT_MACROS -D_
35\|2022-04-27T10:25:52+08:00\" -O2 -pipe -Wall -W -fPIC -fstrict-aliasing -Wno-invalid-offsetof -Wno-unused-paramete
r.cpp -o src/mcpack2pb/generator.o
```

```
y/esp_authenticator.o src/brpc/policy/file_naming_service.o src/brpc/policy/hasher.o src/brpc/policy/consul_naming_service.o src/
c/brpc/policy/redis_protocol.o src/brpc/policy/ubrpc2pb_protocol.o src/brpc/policy/auto_concurrency_limiter.o -Xlinker "-)" -lgfla
t
> Copying to output/include
> Copying to output/lib
> Copying to output/bin
polyspace-configure: 416s: WARNING: Compilation units detected during your build in module libbrpc_a use different languages.
polyspace-configure: 416s: WARNING: Compilation units detected during your build in module libbrpc_so use different languages.
```

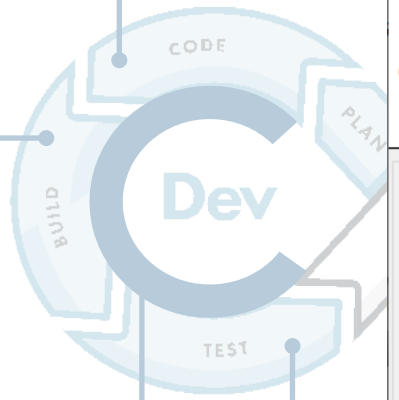
2 POLYSPACE

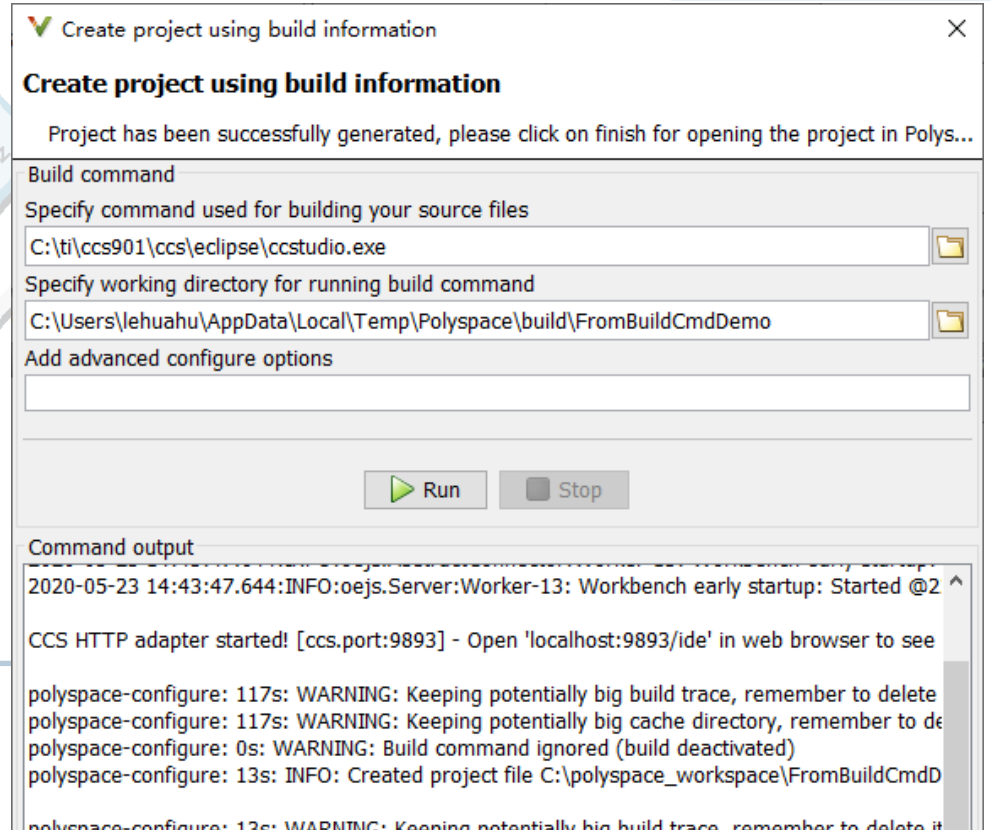
Source Control

git GitHub SUBVERSION

Compilers

TASKING, GCC
GHS, DIAB, etc





3 POLYSPACE

CI/CD

Azure Pipelines Travis CI
Jenkins circleci
Bamboo

集成到 DevOps 中

2 POLYSPACE

Source Control

git GitHub SUBVERSION

Compilers

TASKING, GCC
GHS, DIAB, etc

3 POLYSPACE

CI/CD

Azure Pipelines Travis CI Jenkins circleci Bamboo

流水线

定义

Pipeline script

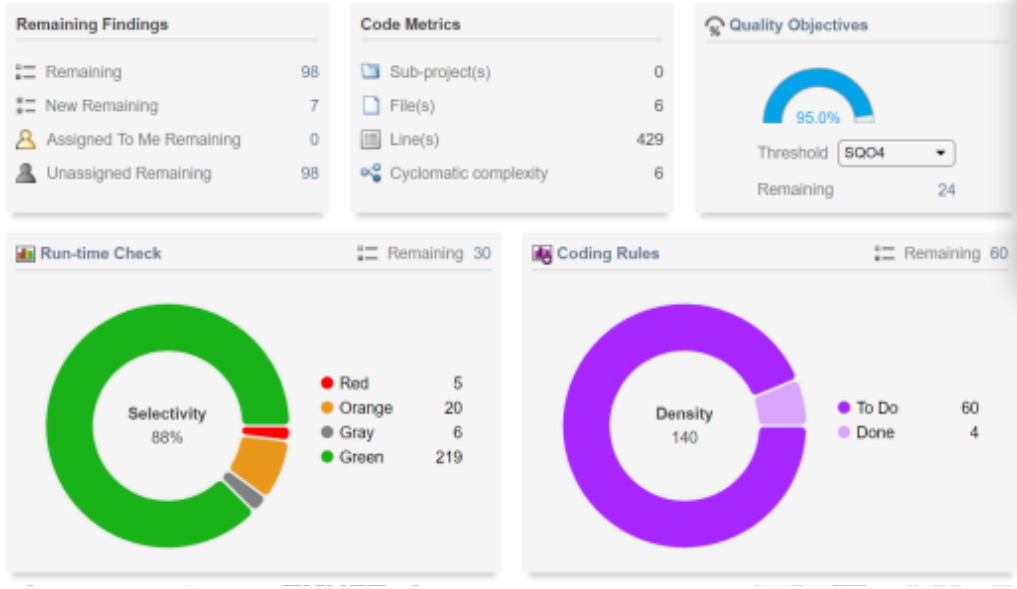
脚本 ?

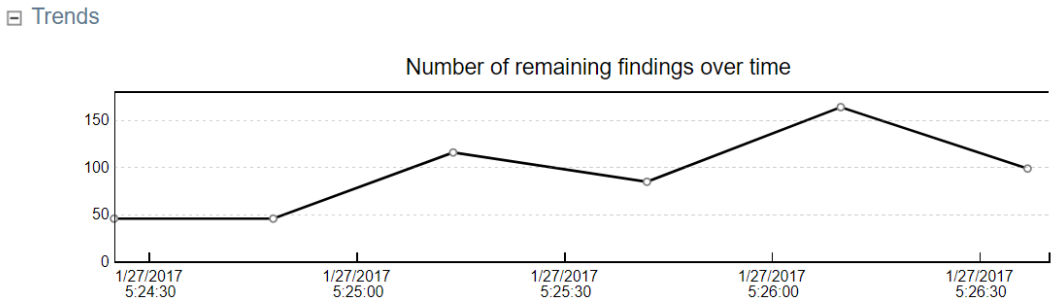
```
59        {
60            sh label: "Cleanup", script: "make clean"
61            sh label: "Polyspace configure", script: "$configure -allow-build-error -allow-overwrite -lang
62        }
63        stage ("Analyze")
64        {
65            sh label: "Polyspace analysis",
66            script: "$analyze -options-file ${PROG}.opts -misra3 all -code-metrics -results-dir $RESULT"
67        }
68
69    // remove this paragraph if you do not use Polyspace Access
```

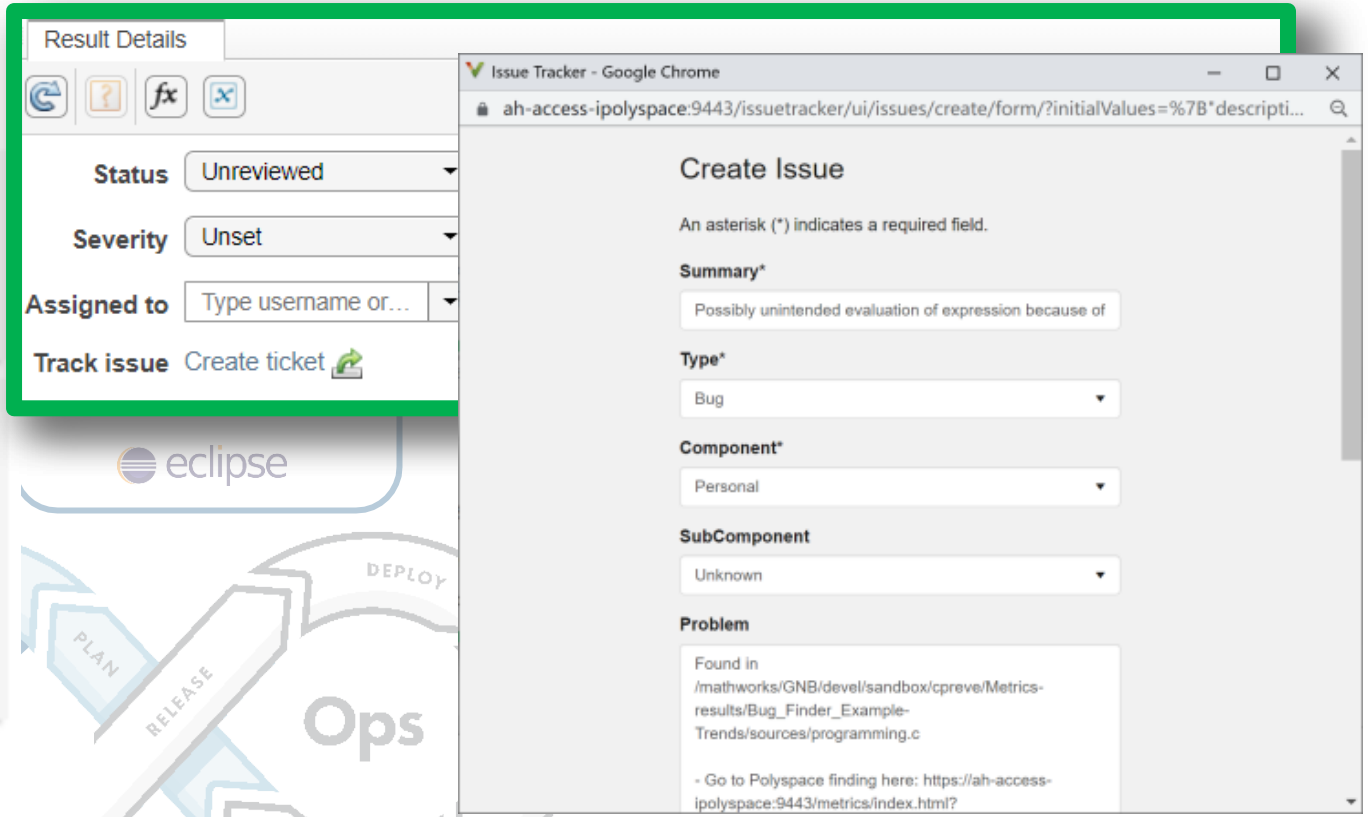
阶段视图

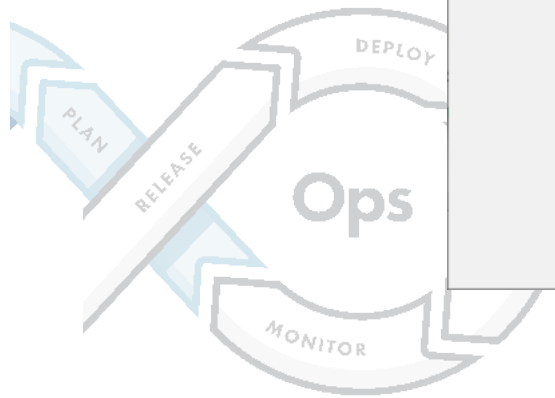
	Prepare	Checkout	Build	Polyspace Analysis	Upload to Access	Quality Gate	Email Notification
Average stage times: (Average full run time: ~40s)	374ms	2s	9s	12s	4s	8s	875ms
#41 Sep 13 15:35 No Changes	323ms	2s	10s	13s	4s	8s	1s
#40 Sep 10 23:08 No Changes	303ms	1s	9s	11s	5s	8s	1s
#39 Sep 10 22:52 No Changes	471ms	3s	10s	15s	4s	8s	1s

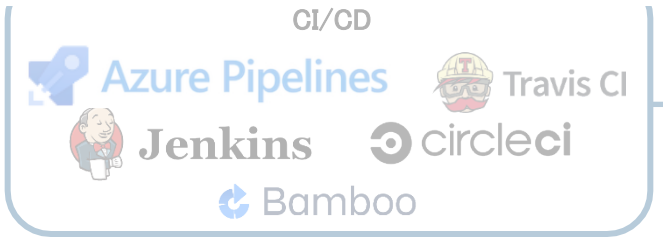
集成到 DevOps 中



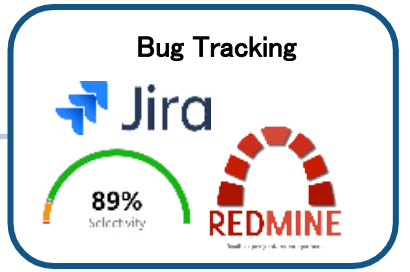


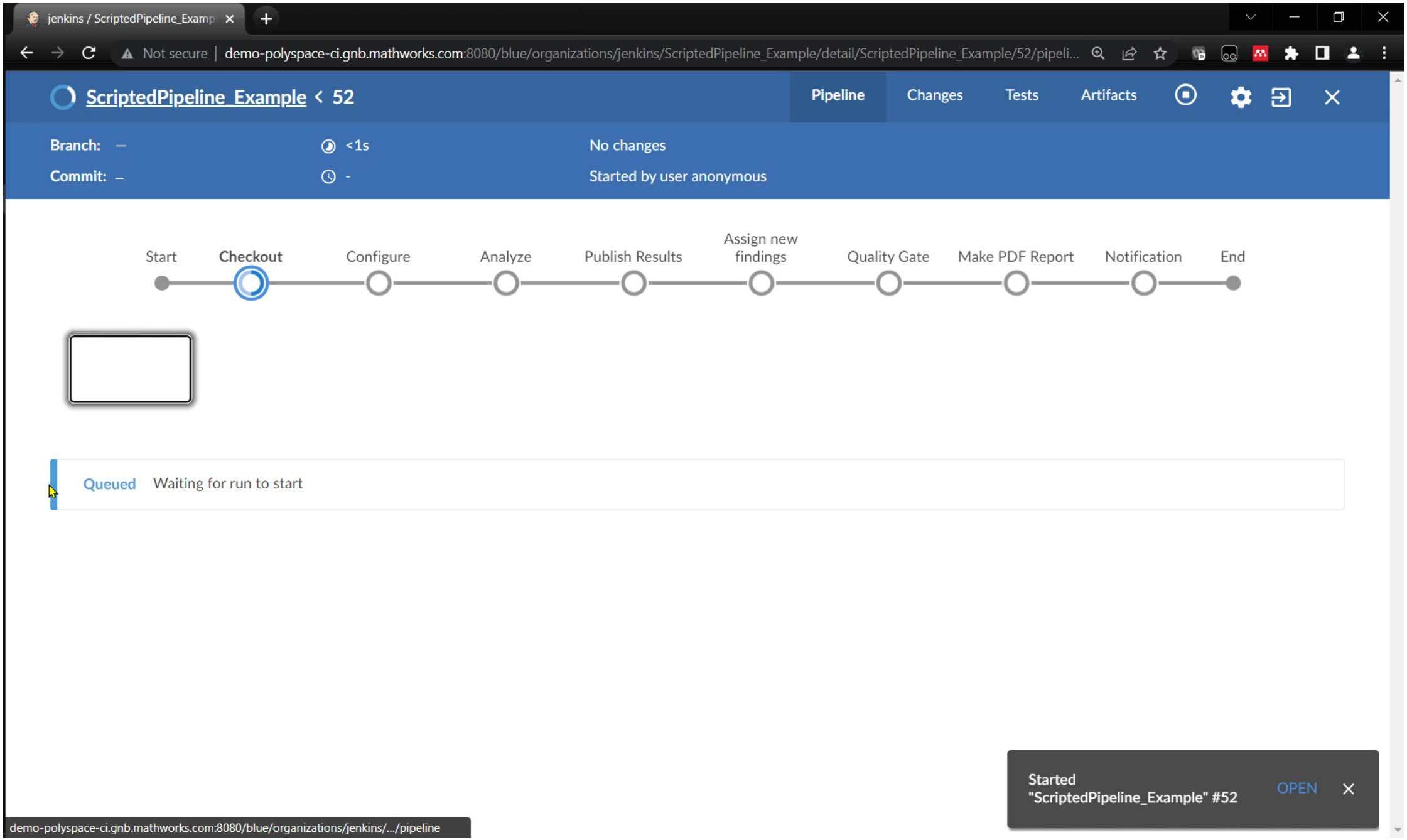






4 POLYSPACE




jenkins / ScriptedPipeline_Examp
Not secure | demo-polyspace-ci.gnb.mathworks.com:8080/blue/organizations/jenkins/ScriptedPipeline_Example/detail/ScriptedPipeline_Example/52/pipeli...
ScriptedPipeline_Example < 52
Pipeline
Changes
Tests
Artifacts
Branch: –
Commit: –
<1s
-
No changes
Started by user anonymous
Start
Checkout
Configure
Analyze
Publish Results
Assign new findings
Quality Gate
Make PDF Report
Notification
End
Queued
Waiting for run to start
Started "ScriptedPipeline_Example" #52
OPEN
demo-polyspace-ci.gnb.mathworks.com:8080/blue/organizations/jenkins/.../pipeline

云端部署持续集成，用于大规模系统

Azure Pipeline 的持续集成工作流
使用参考架构进行快速设置

Dockerfile

Edit

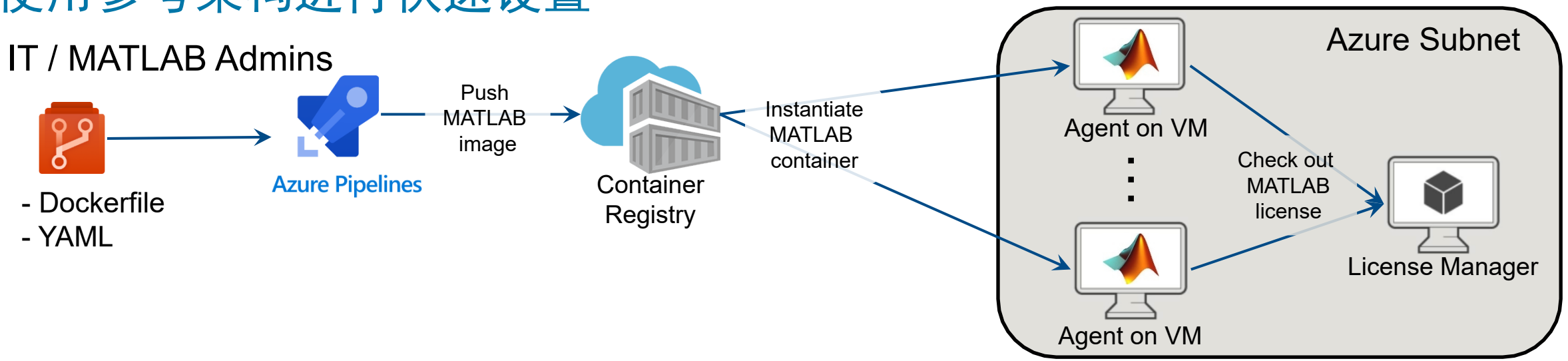
Contents History Compare Blame

Press F11 to exit full screen

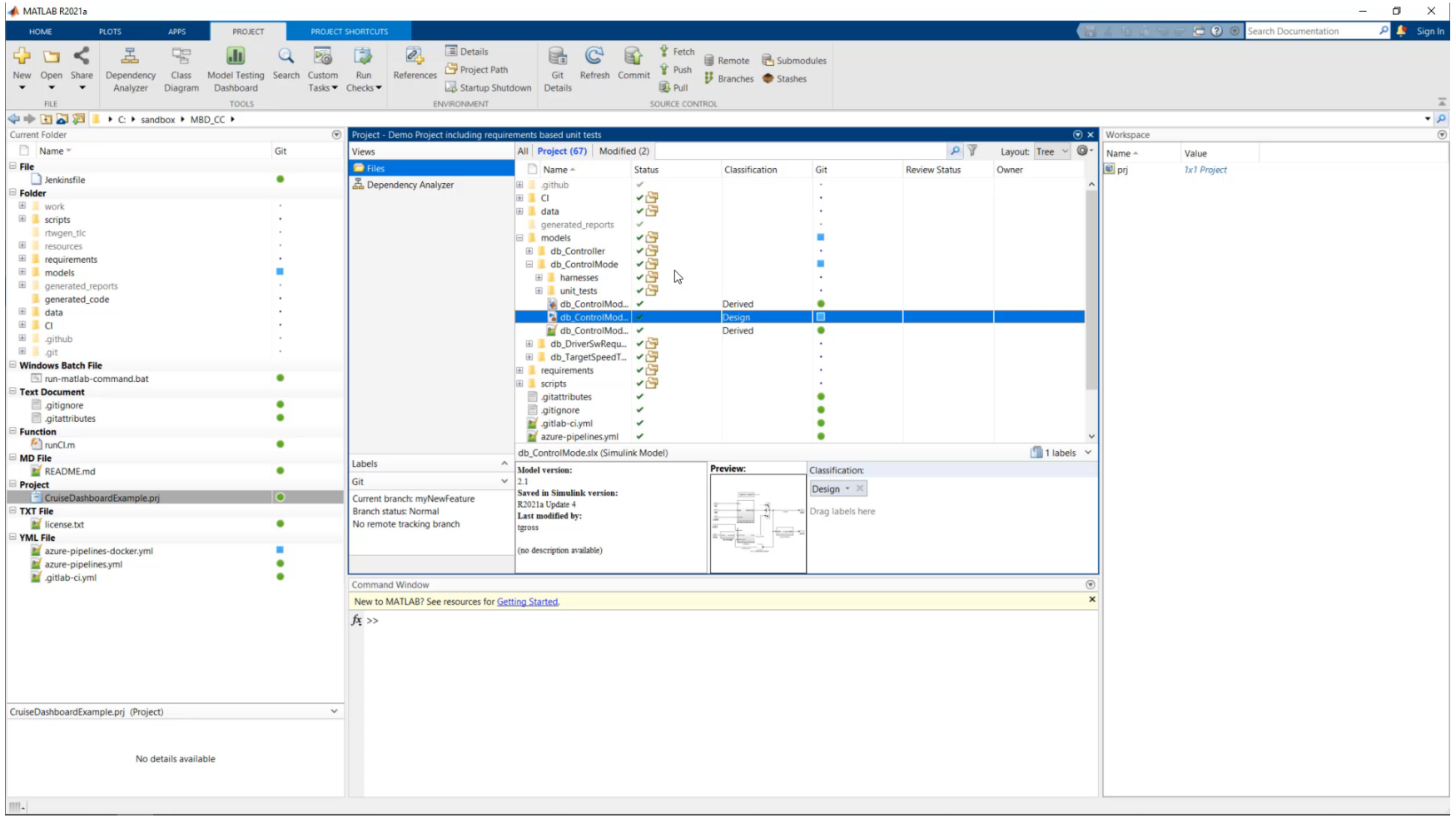
```
# Copyright 2019 - 2021 The MathWorks, Inc.

# To specify which MATLAB release to install in the container, edit the value of the MATLAB_RELEASE argument.
# Use lower case to specify the release, for example: ARG MATLAB_RELEASE=r2020a
#ARG MATLAB_RELEASE=r2021b
ARG MATLAB_RELEASE

# When you start the build stage, this Dockerfile by default uses the Ubuntu-based matlab-deps image.
# To check the available matlab-deps images, see: https://hub.docker.com/r/mathworks/matlab-deps
FROM mathworks/matlab-deps:${MATLAB_RELEASE}

# Declare the global argument to use at the current build stage
ARG MATLAB_RELEASE

# Install mpm dependencies
RUN export DEBIAN_FRONTEND=noninteractive && apt-get update && \
    apt-get install --no-install-recommends --yes \
        wget \
        unzip \
        ca-certificates && \
    apt-get clean && apt-get autoremove

# Run mpm to install MATLAB in the target location and delete the mpm installation afterwards
RUN wget -q https://www.mathworks.com/mpm/glnxa64/mpm && \
    chmod +x mpm && \
    ./mpm install \
        --release=${MATLAB_RELEASE} \
        --destination=/opt/matlab \
        --products MATLAB Simulink Stateflow Simulink_Requirements MATLAB_Coder Simulink_Coder Embedded_Coder Simulink_Check Simulink_Test Simulink_Coverage Simulink_Design_Verifier && \
    rm -f mpm /tmp/mathworks_root.log && \
    ln -s /opt/matlab/bin/matlab /usr/local/bin/matlab

# Add "matlab" user and grant sudo permission.
RUN adduser --shell /bin/bash --disabled-password --gecos "" matlab && \
    echo "matlab ALL=(ALL) NOPASSWD: ALL" > /etc/sudoers.d/matlab && \
    chmod 0440 /etc/sudoers.d/matlab

# One of the following 2 ways of configuring the license server to use must be
# uncommented.

ARG LICENSE_SERVER
# Specify the host and port of the machine that serves the network licenses
# if you want to bind in the license info as an environment variable. This
# is the preferred option for licensing. It is either possible to build with
# something like --build-arg LICENSE_SERVER=27000@MyServerName, alternatively
# you could specify the license server directly using
ENV MLM_LICENSE_FILE=27000@netlm-server.internal.cloudapp.net
#ENV MLM_LICENSE_FILE=$LICENSE_SERVER

# Alternatively you can put a license file into the container.
# You should fill this file out with the details of the license
# server you want to use and uncomment the following line.
# COPY network.lic /opt/matlab/licenses/

# Set user and work directory
USER matlab
WORKDIR /home/matlab
ENTRYPOINT ["matlab"]
CMD [""]
```

Dependencies

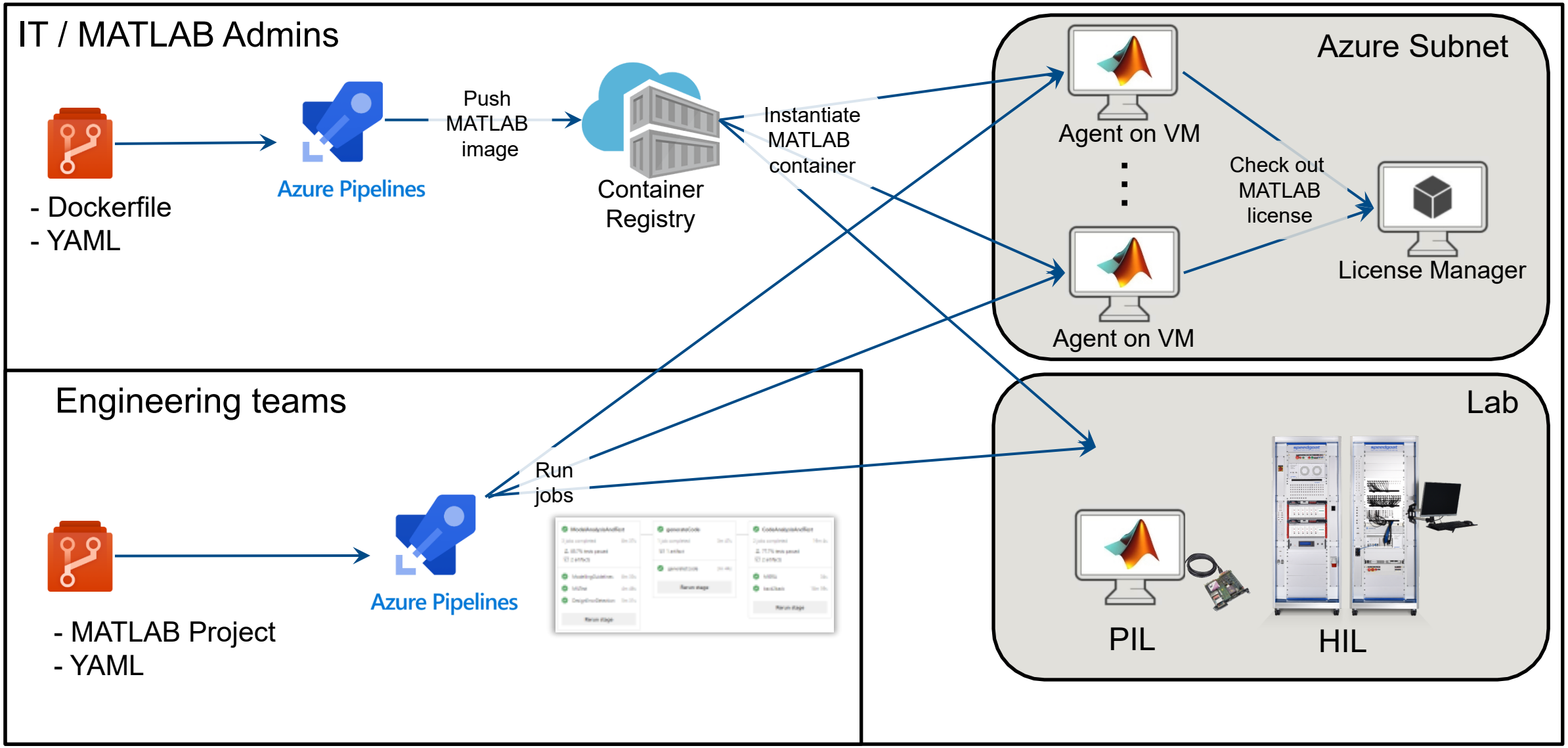
Azure Pipeline 的持续集成工作流
使用参考架构进行快速设置



Pipeline 的持续集成工作流
推送变更到云端触发MBD的云端构建



架构图示例


IT / MATLAB Admins
- Dockerfile
- YAML
Azure Pipelines
Push MATLAB image
Container Registry
Instantiate MATLAB container
Azure Subnet
Agent on VM
Agent on VM
Check out MATLAB license
License Manager
Engineering teams
- MATLAB Project
- YAML
Azure Pipelines
Run jobs
Lab
PIL
HIL

持续集成实践中的建议

建立成熟的问题检测和处置流程

Maturity	Description of static analysis
Level 0	No analysis performed
Level 1	Analysis once per release
Level 2	Analysis multiple times per release, critical issues added to backlog
Level 3	Fully automated & frequent analysis, all critical issues resolved
Level 4	New issues break the build, critical and high issues resolved, IDE-plugins are used

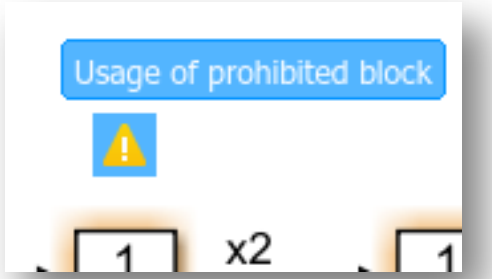
SAFe® PROVIDED BY SCALED AGILE

https://www.scaledagileframework.com/accelerating-flow-with-devsecops-and-the-software-factory/

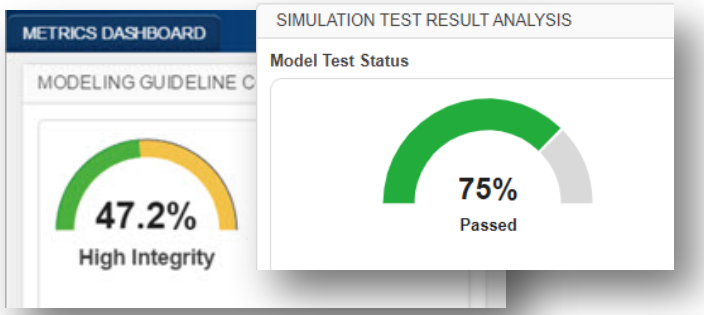
CI pipeline

Process

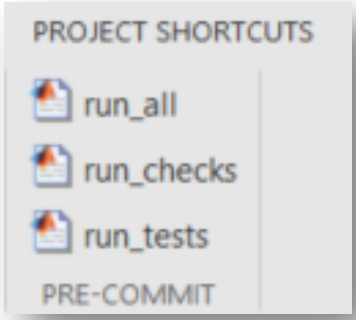
Pre-commit checks



Edit time checks



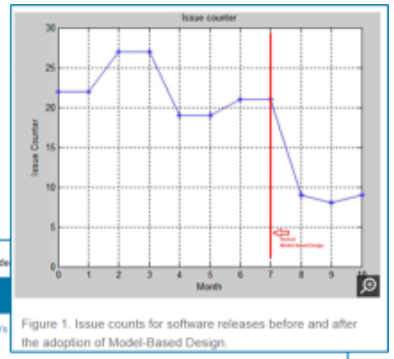
Quality Dashboards

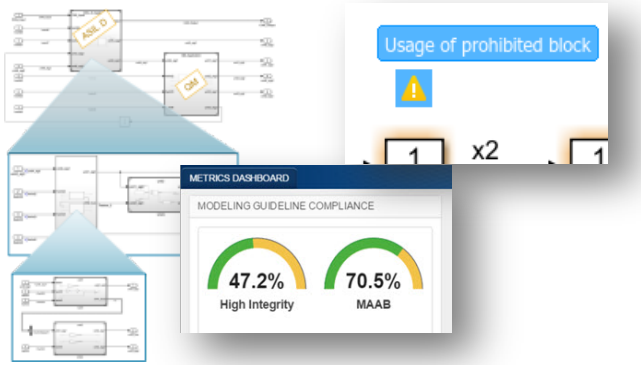


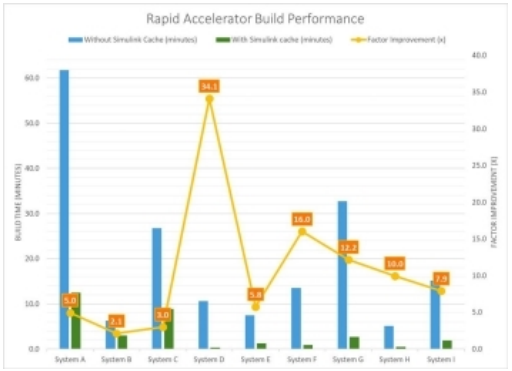
MATLAB Project shortcuts

保持沟通联络 – 利用MathWorks的CI和DevOps专业知识

- 提高自动化程度和CI成熟度
  - 将MBD与CI系统、版本管理系统等其他系统集成
  - 尽早检查错误，建立成熟的问题检测和处置流程
- 改进系统架构，掌控复杂度
  - 为测试自动化和功能安全和网络安全进行模块化
- 解决性能瓶颈



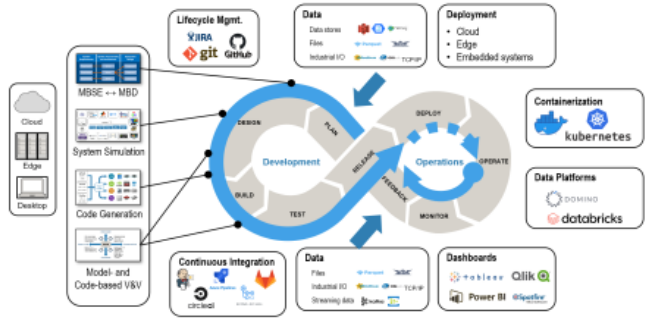




要点回顾

- 将基于模型设计和Polyspace融入CI和DevOps应对软件开发转型的挑战
- 在云上部署CI，以提高MBD的灵活性
- 保持联络以强化CI流程和最佳实践

	Throughput		Stability	
DORA metrics impact	Lead Time For Change	Deployment Frequency	Change Failure Rate	Mean Time to Recovery
CI/CD pipelines	x	x	x	x
toolchain upgrades	x	x		x
system architecture		x	x	
cache files on CI	x	x		



感谢您的聆听！