MATLAB EXPO 2021

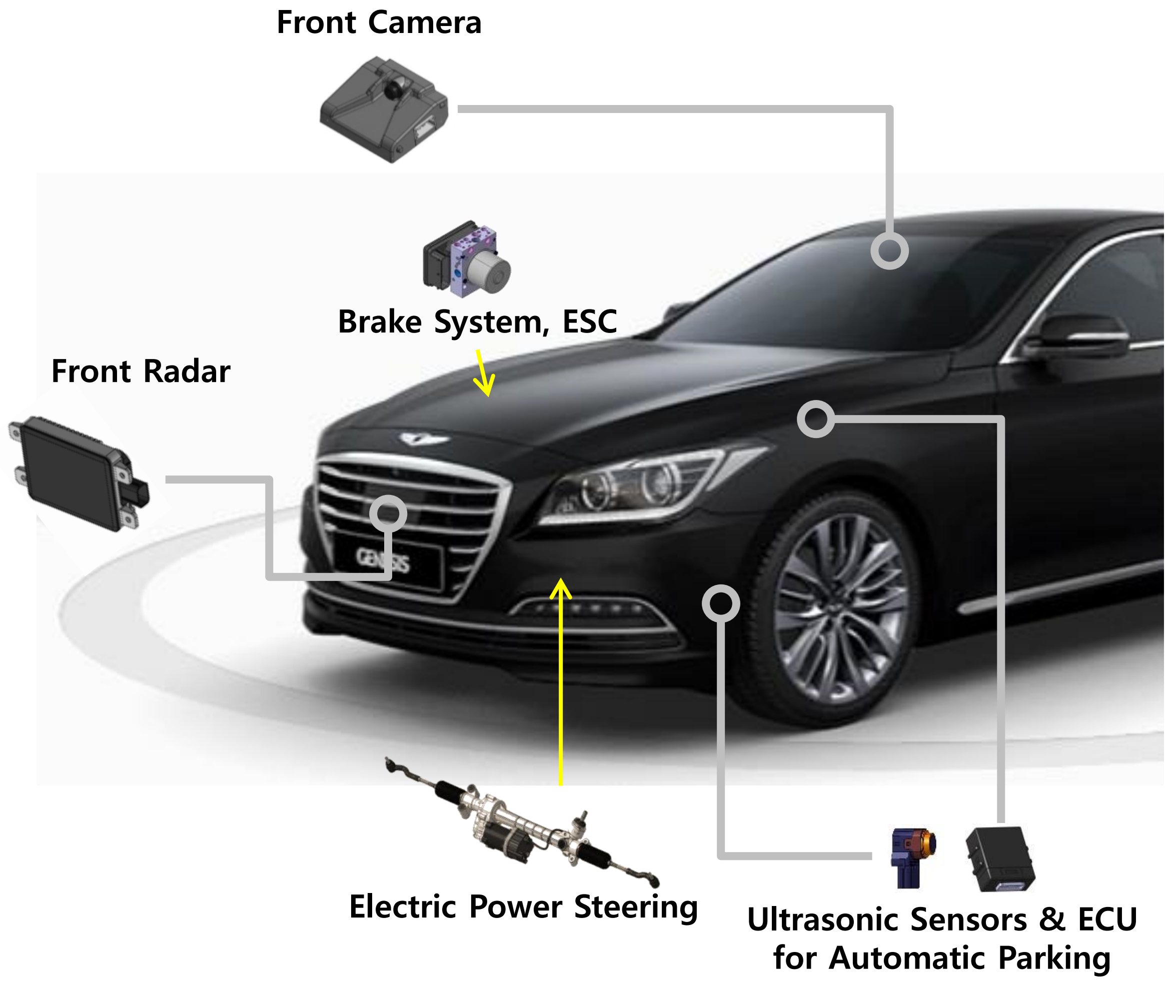
# Polyspace Server Products 및 Polyspace Access Products를 활용한 SW 정적검증 자동화

*이민채, ㈜만도*

## Contents

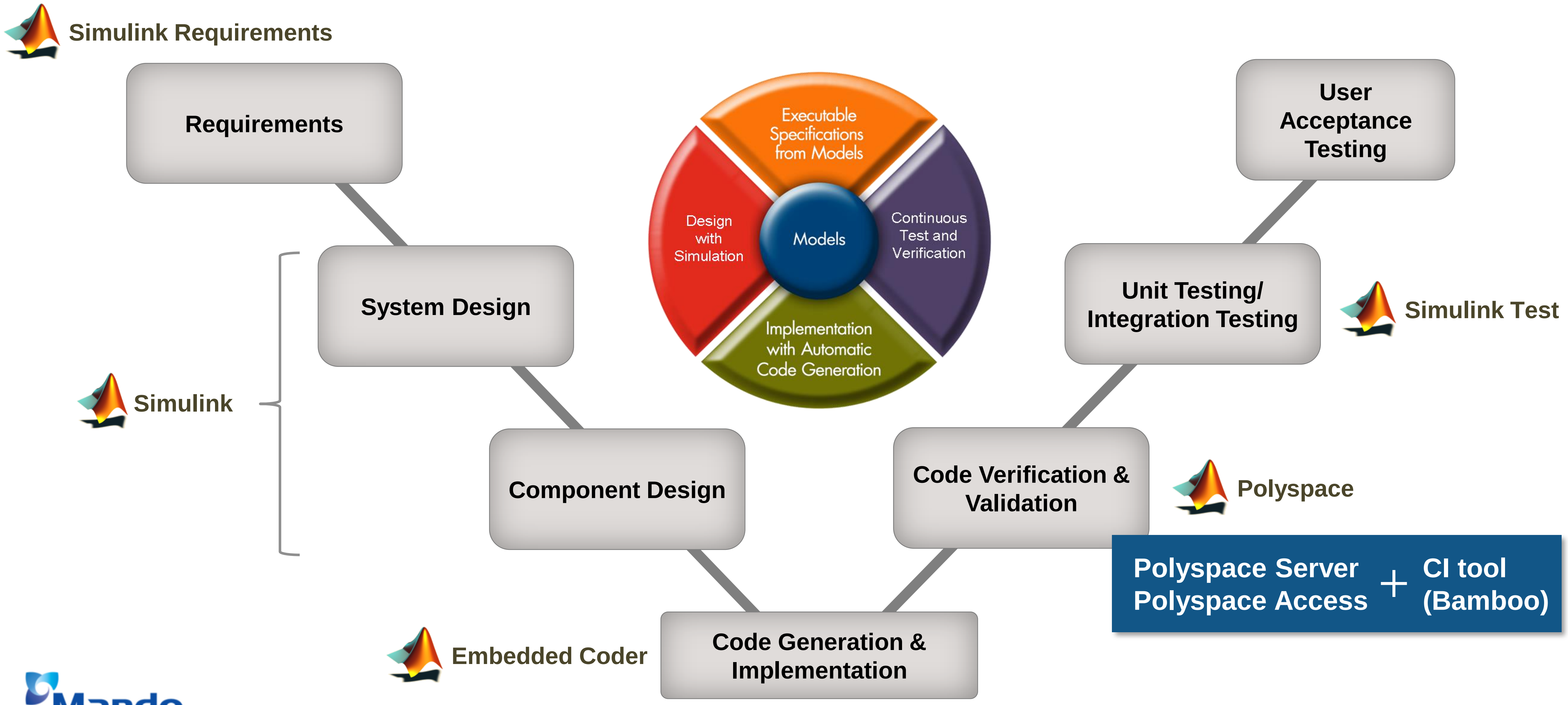
1. Introduction to Mando and Presenter
2. Project Overview
3. Project Goals and Challenges
4. How did we get the “Automated Static Analysis and Collaborative Review” platform
5. Achievements and Outlook
6. Future Works
7. Conclusions

## Mando - ADAS BU



Mando

# Development Process with MBD and Hand Code


Simulink Requirements
Requirements
System Design
Simulink
Component Design
Embedded Coder
Code Generation & Implementation
Executable Specifications from Models
Design with Simulation
Models
Continuous Test and Verification
Implementation with Automatic Code Generation
User Acceptance Testing
Unit Testing/ Integration Testing
Simulink Test
Code Verification & Validation
Polyspace
Polyspace Server
Polyspace Access
+
CI tool (Bamboo)

## 발표자 소개

이민채 책임연구원

㈜만도 / ADAS BU

- 연구 분야
  - ADAS 및 자율주행 주행상황 판단 시스템
  - 차량동역학 기반 제어시스템 설계 및 구현
  - 자동차 SW 플랫폼, C/C++/Python, SW 검증
- 학력
  - 한양대학교 전자전기컴퓨터공학부 학사
  - 한양대학교 자동차공학과 석사
  - 한양대학교 자동차공학과 박사
- 경력
  - 한양대학교 자동차공학과 자동차전자제어연구소 (ACE Lab, 2006~2013)
  - 자율주행자동차 경진대회 우승 (현대자동차, 2010/2012)
  - 2013 무인 자율주행 자동차 경진대회 대상 (한국자동차공학회, 2013)
  - ㈜만도 Global R&D Center 책임연구원 (2014 ~ 현재)

Mando

## Project Overview

Static Code Analysis for Automotive Software

What is Static Code Analysis?

- Coding Guidelines
  - MISRA C: Software development guidelines for the C programming language developed by MISRA (Motor Industry Software Reliability Association)
- Run-Time Error Detection
  - Run-Time Error: Problems that appear during the execution of a program
  - Division by Zero, Overflow/Underflow, Use of Uninitialized Variables, …
- Code Metrics
  - A statistical measurement of code complexity, size, coupling and cohesion

## Project Overview
### Polyspace Products

- Polyspace Bug Finder and Polyspace Code Prover
  - Polyspace® Bug Finder™ ***identifies run-time errors, concurrency issues, security vulnerabilities, and other defects*** in C and C++ embedded software.
  - Polyspace® Code Prover™ is a sound static analysis tool that ***proves the absence of overflow, divide-by-zero, out-of-bounds array access, and other run-time errors*** in C and C++ source code.

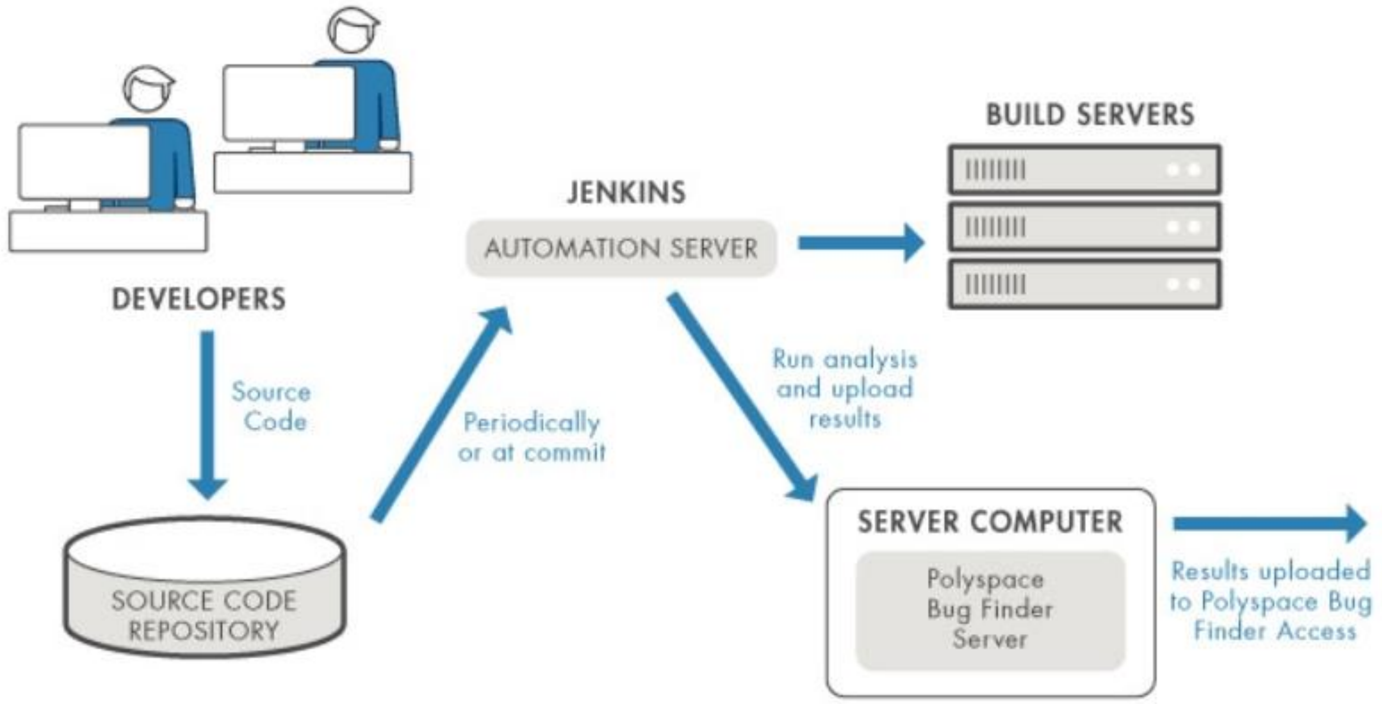
*for this project*

| Desktop | Server | Web |
|---|---|---|
| Polyspace Bug Finder | *Polyspace Bug Finder Server* | *Polyspace Bug Finder Access* |
| Polyspace Code Prover | Polyspace Code Prover Server | Polyspace Code Prover Access |

Mando

## Project Overview
### Continuous Integration and Static Code Analysis

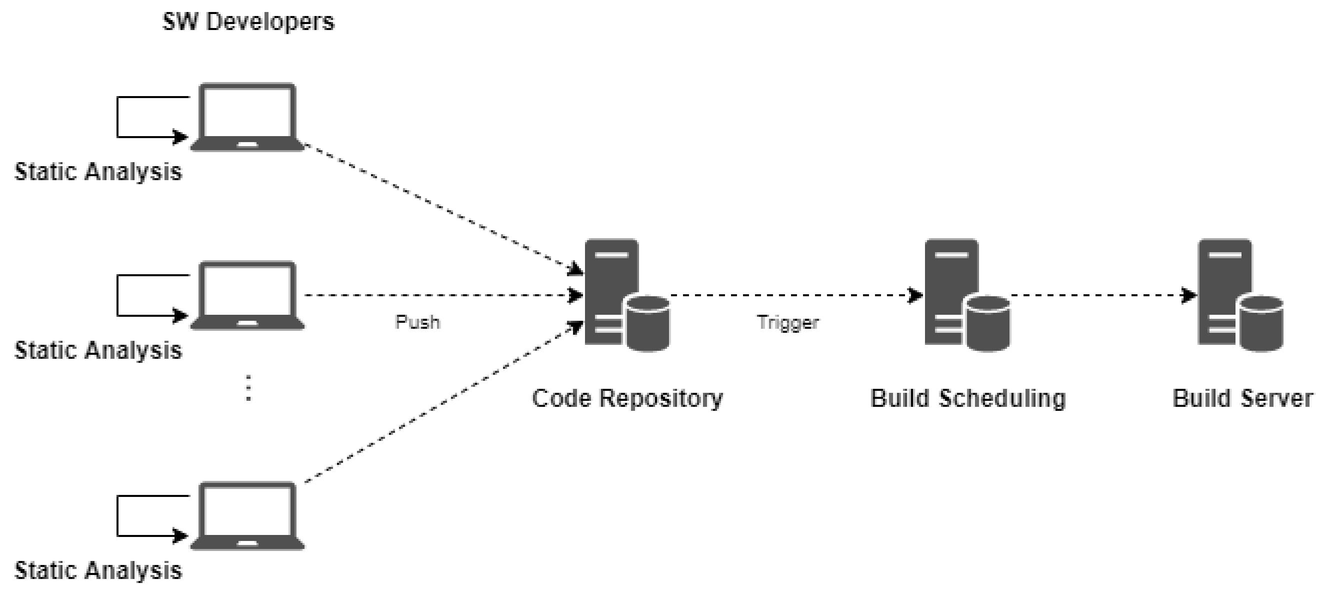
- Continuous Integration
  - Automating the integration of code changes from multiple contributors
  - Jenkins, Bamboo, GitLab, …
- Integrating Polyspace with continuous integration environment


DEVELOPERS
Source Code
SOURCE CODE REPOSITORY
Periodically or at commit
JENKINS
AUTOMATION SERVER
BUILD SERVERS
Run analysis and upload results
SERVER COMPUTER
Polyspace Bug Finder Server
Results uploaded to Polyspace Bug Finder Access

## Project Goals and Challenges

### Conventional Development Process

- SW developer used standalone static code analysis tools



→ Static code analysis is required to perform early in development, before software testing begins.

Mando

MATLAB EXPO

## Project Goals and Challenges

### Development Process with CI and Polyspace

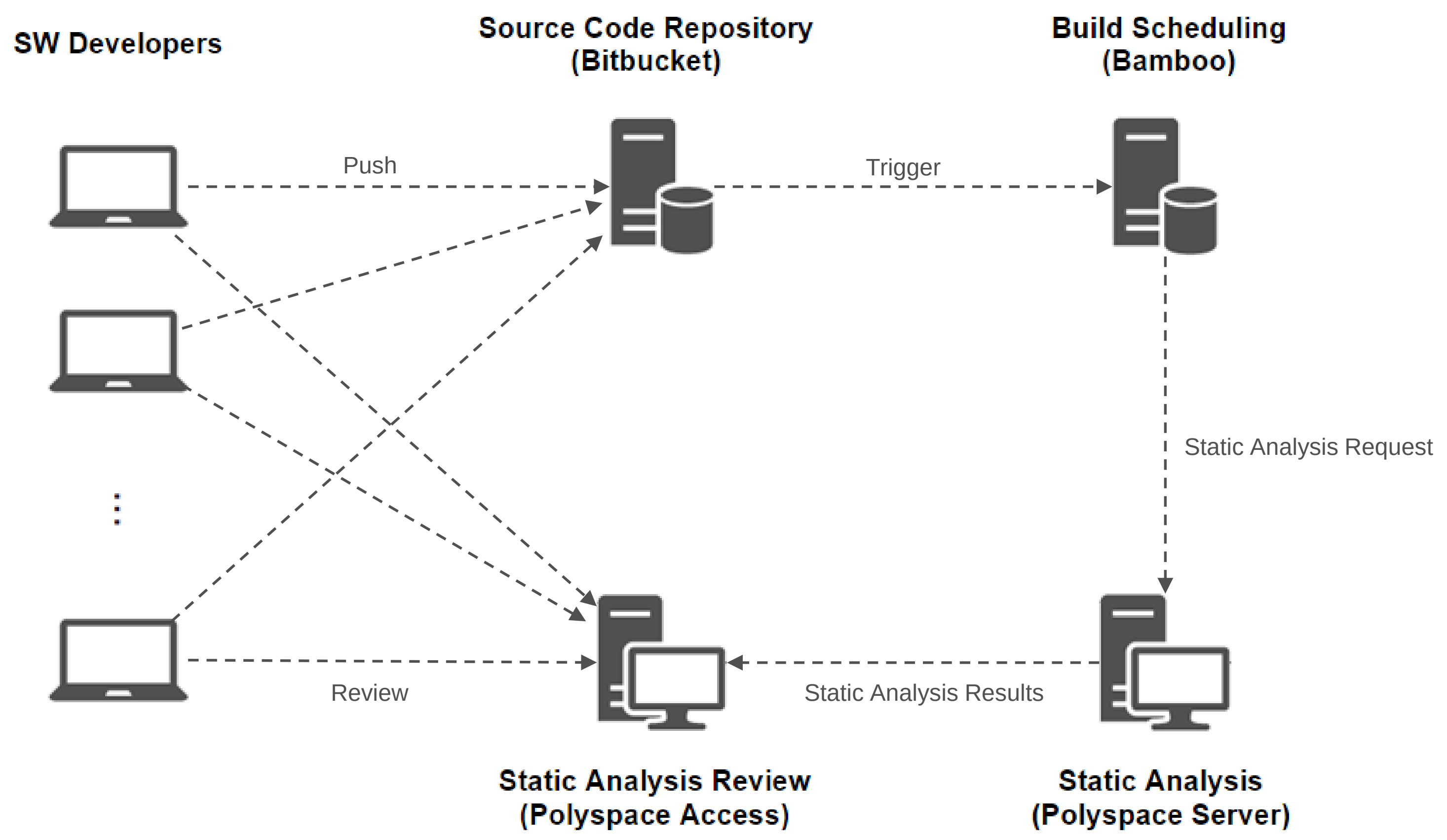
- Atlassian Bitbucket® and Bamboo® are used for continuous integration platform
  - Bitbucket(GIT) for software code repository
  - Bamboo for build and static code analysis triggering and scheduling

- Mathworks Polyspace products are used for static code analysis tool
  - Polyspace Bug Finder Server for static code analysis with CI tools
  - Polyspace Bug Finder Access for web based result review

Mando

## Project Goals and Challenges
### Development Process with CI and Polyspace

- System architecture of automated static code analysis platform


SW Developers
Source Code Repository (Bitbucket)
Build Scheduling (Bamboo)
Push
Trigger
Static Analysis Request
Review
Static Analysis Results
Static Analysis Review (Polyspace Access)
Static Analysis (Polyspace Server)

## Automated Static Analysis and Collaborative Review

### Preparations for automated static analysis

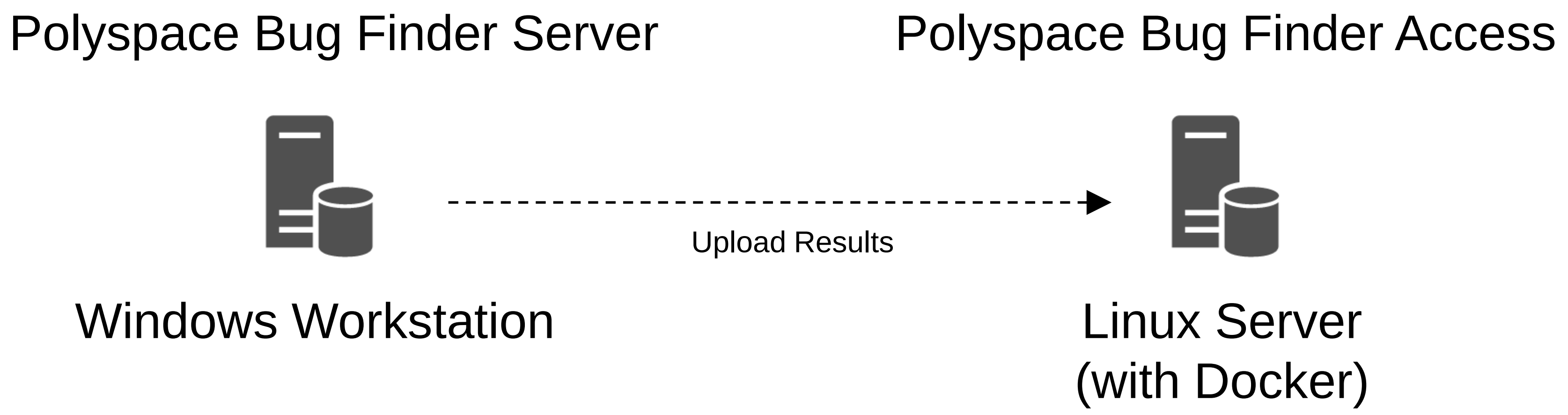
- Hardware for analysis and web server
  - A CI server(Jenkins, Bamboo, …) is required to trigger a static analysis
  - An analysis server is required to run Polyspace Bug Finder Server (w/ license server)
  - A web server is required to run Polyspace Bug Finder Access

- Software for static analysis
  - Software compile options or compile environments for Polyspace project setup

# Automated Static Analysis and Collaborative Review

## Preparations for automated static analysis

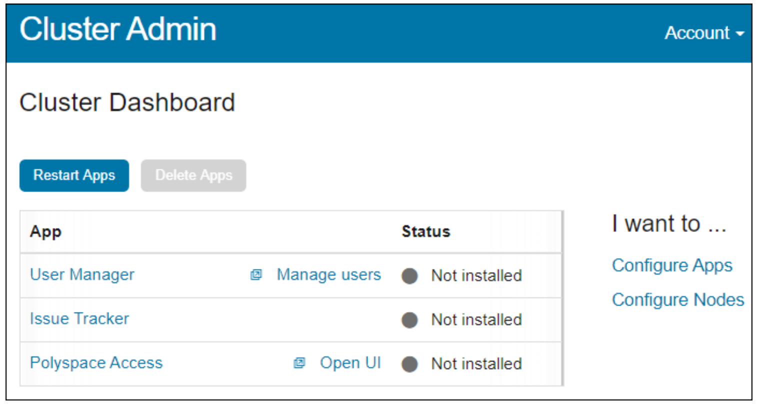
- Installation of Polyspace Server and Access
  - https://www.mathworks.com/help/polyspace_bug_finder_server/gs/install-products-required-for-polyspace-analysis-on-server.html



## Automated Static Analysis and Collaborative Review

### Preparations for automated static analysis

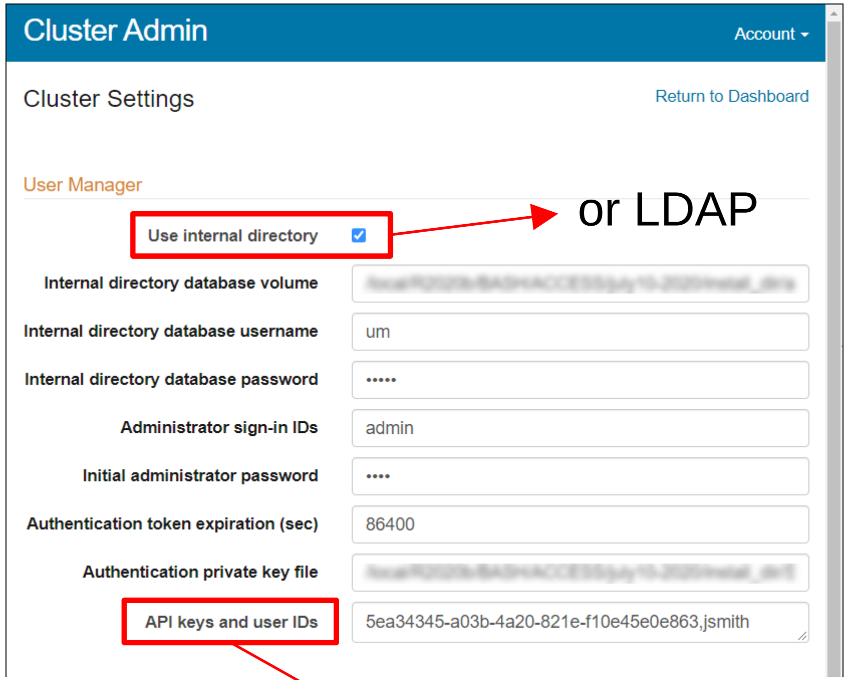
- Configuration of Polyspace Access
  - Installation command: admin-docker-agent
  - Open web browser and go to URL specified in the command-line output



Cluster Dashboard

| App | | Status |
|---|---|---|
| User Manager | Manage users | ● Not installed |
| Issue Tracker | | ● Not installed |
| Polyspace Access | Open UI | ● Not installed |

I want to ...
Configure Apps
Configure Nodes



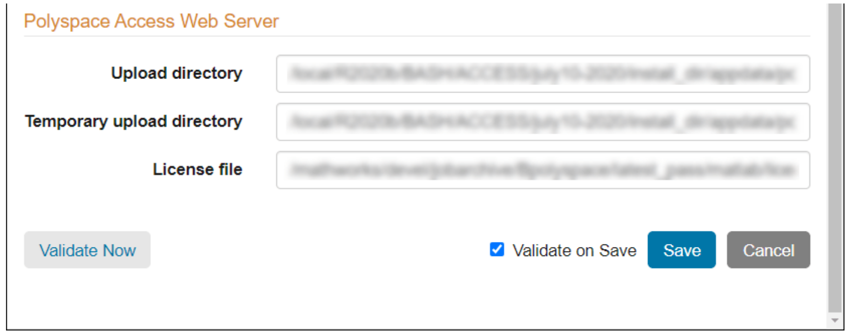
Cluster Settings

User Manager

| | |
|---|---|
| Use internal directory | ✓ |
| Internal directory database volume | [redacted] |
| Internal directory database username | um |
| Internal directory database password | ••••• |
| Administrator sign-in IDs | admin |
| Initial administrator password | •••• |
| Authentication token expiration (sec) | 86400 |
| Authentication private key file | [redacted] |
| API keys and user IDs | 5ea34345-a03b-4a20-821e-f10e45e0e863,jsmith |

or LDAP

for upload user authentication



Polyspace Access Web Server

| | |
|---|---|
| Upload directory | [redacted] |
| Temporary upload directory | [redacted] |
| License file | [redacted] |

✓ Validate on Save

## Automated Static Analysis and Collaborative Review

### First step – Command line based static analysis

- Polyspace Bug Finder Server is used for command line based static analysis (w/o GUI)

- To check if the installation of Polyspace Bug Finder Server was successful
  - Open a command window. Navigate to polyspaceserverroot\polyspace\bin
  - Run “polyspace-bug-finder-server –help”

```
C:\Program Files\Polyspace Server\R2021a\polyspace\bin>

C:\Program Files\Polyspace Server\R2021a\polyspace\bin>polyspace-bug-
finder-server -help
```

## Automated Static Analysis and Collaborative Review

First step – Command line based static analysis

- Syntax for Polyspace Bug Finder Server
  - sources sourceFiles [OPTIONS]
  - sources-list-file listOfSources [OPTIONS]
  - option-file optFile

```
> polyspace-bug-finder-server -source-list-file source_files.txt -option-file options.txt
```

## Automated Static Analysis and Collaborative Review

### First step – Command line based static analysis

- Create ‘source_files.txt’ file with your options
  - Specify your sources in the text file, on each line, specify the path to a source file
  - You can specify an absolute path or a path relative to the folder from which you are running the analysis

```
C:\Sources\myfile.c
C:\Sources2\myfile2.c
```

## Automated Static Analysis and Collaborative Review

### First step – Command line based static analysis

- Create ‘options.txt’ file with your options

```
#These are the options for MyBugFinderProject
-lang c
-prog MyBugFinderProject
-author jsmith
-sources "mymain.c,funAlgebra.c,funGeometry.c"
-target x86_64
-compiler generic
-dos
-misra2 required-rules
-do-not-generate-results-for all-headers
-checkers default
-disable-checkers concurrency
-results-dir C:\Polyspace\MyBugFinderProject
```

We don’t need these lines.
See next pages…

# Automated Static Analysis and Collaborative Review

## First step – Command line based static analysis

- Combining command line arguments and option files

Branch name as a project name

```
> SET project_name=${bamboo.planRepository.branch}

> polyspace-bug-finder-server -prog %project_name% -source-list-file ./source_files.txt
-option-file ./project_options.txt -option-file ./include_path.txt -results-dir ./Result
```

Multiple option files to separate options with respect to characteristics

# Automated Static Analysis and Collaborative Review

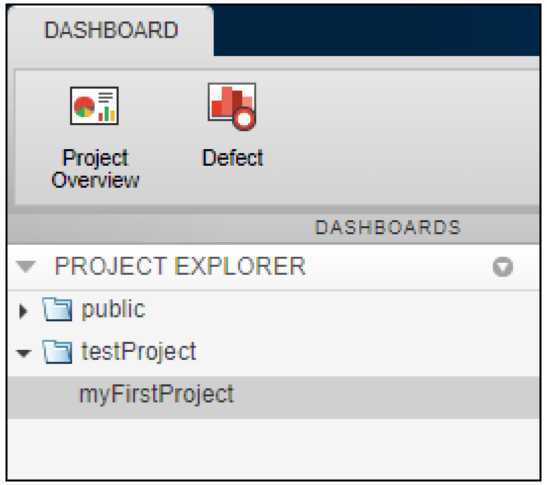
## Second step – Upload the results to Polyspace Access server

- Create project to Polyspace Access server

```
> polyspace-access -host hostName -port portNumber -login username -encrypted-
password pwd -create-project testProject
```

- Upload results to Polyspace Access server

```
> polyspace-access -host hostName -port portNumber -login
username -encrypted-password pwd -upload . -project
myFirstProject -parent-project testProject
```



## Automated Static Analysis and Collaborative Review

Second step – Upload the results to Polyspace Access server

- Create project to Polyspace Access server

```
> ${bamboo.polyspace_access_app} -host ${bamboo.polyspace_server_ip} -port ${bamboo.polyspace_server_port} -
protocol http -api-key ${bamboo.polyspace_api_key} -create-project ${bamboo.polyspace_location_on_access}

> ${bamboo.polyspace_access_app} -host ${bamboo.polyspace_server_ip} -port ${bamboo.polyspace_server_port} -
protocol http -api-key ${bamboo.polyspace_api_key} -upload .\Result -parent-project
${bamboo.polyspace_location_on_access} -project %project_name%
```

- Bamboo user defined variables
  - bamboo.polyspace_access_app
  - bamboo.polyspace_server_ip, bamboo.polyspace_server_port
  - bamboo.polyspace_api_key
  - bamboo.polyspace_location_on_access

## Automated Static Analysis and Collaborative Review

### Third step – Generate report

- Run report generator

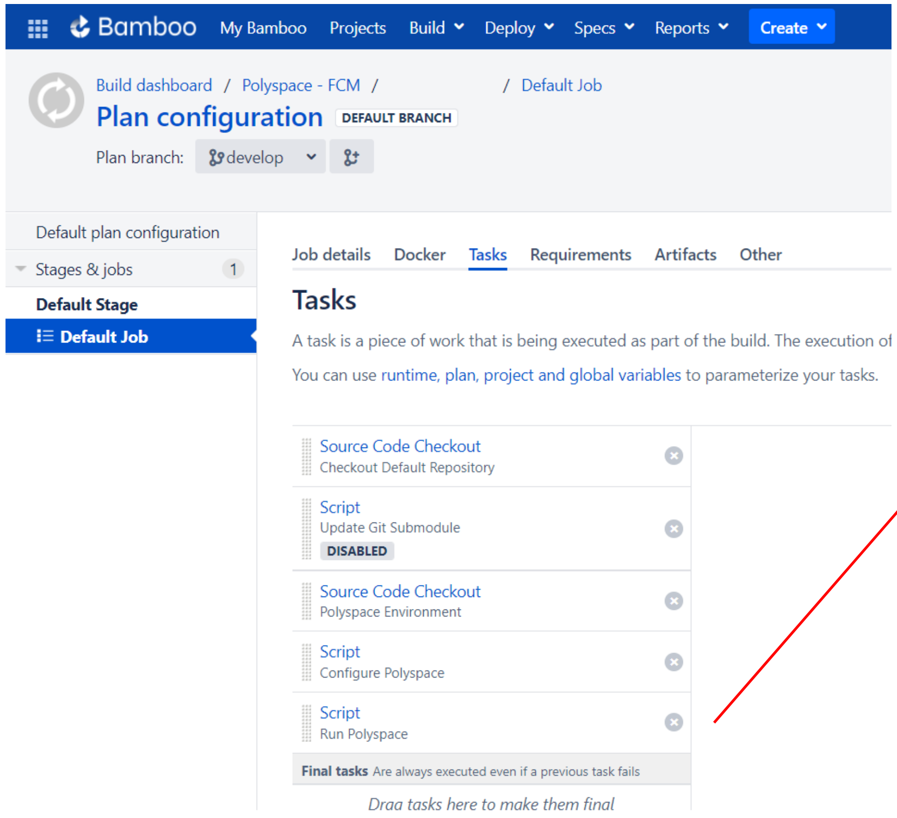
```
> ${bamboo.polyspace_report_generator} -generate-results-list-file -results-dir ./result

> ${bamboo.polyspace_report_generator} -template ${bamboo.polyspace_report_template_dir}/developer.rpt -results-
dir ./result

> ${bamboo.polyspace_report_generator} -template
${bamboo.polyspace_report_template_dir}/bug_finder/BugFinderSummary.rpt -results-dir ./result
```

- Report template
  - C:\Program Files\Polyspace\R2021a\toolbox\polyspace\psrptgen\templates

# Automated Static Analysis and Collaborative Review

## Integration with Bamboo

- Bamboo plan configuration



Build dashboard / Polyspace - FCM / / Default Job

**Plan configuration** DEFAULT BRANCH

Plan branch: develop

Default plan configuration

Stages & jobs 1

**Default Stage**

**Default Job**

Job details | Docker | Tasks | Requirements | Artifacts | Other

**Tasks**

A task is a piece of work that is being executed as part of the build. The execution of

You can use runtime, plan, project and global variables to parameterize your tasks.

Source Code Checkout
Checkout Default Repository

Script
Update Git Submodule
DISABLED

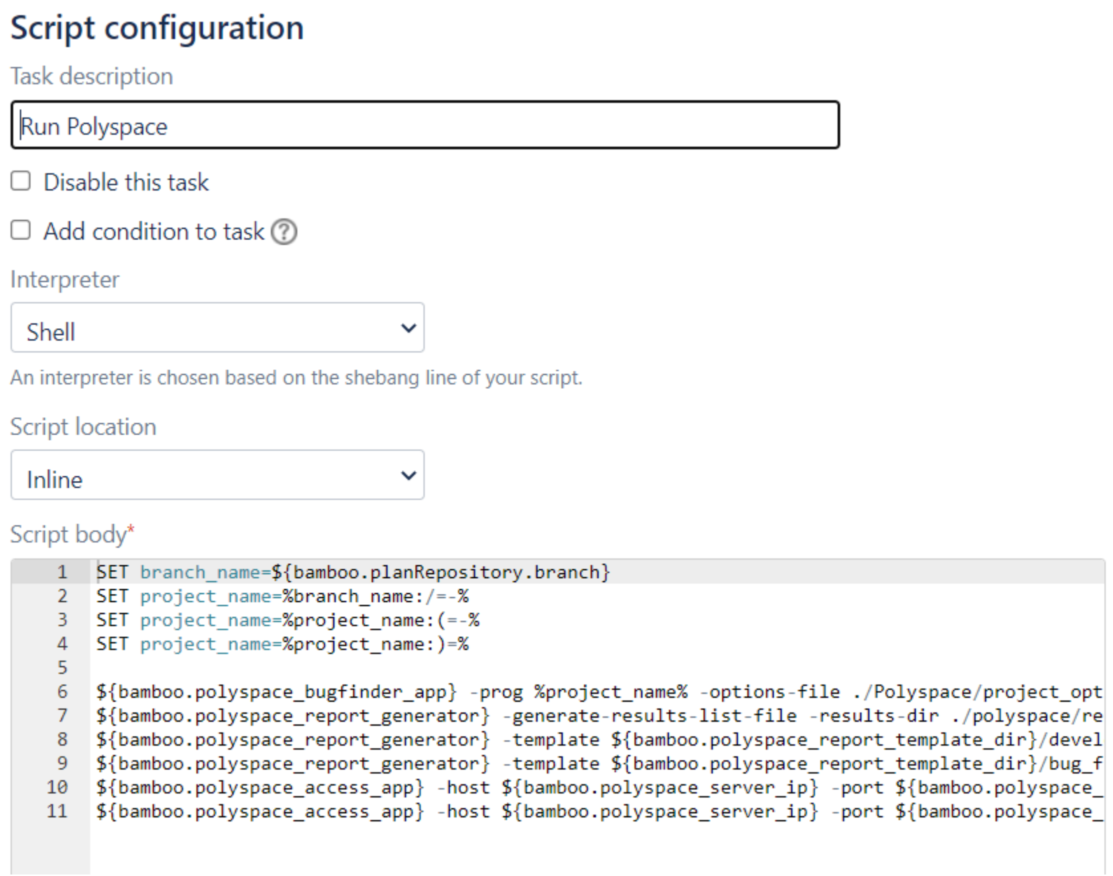
Source Code Checkout
Polyspace Environment

Script
Configure Polyspace

Script
Run Polyspace

**Final tasks** Are always executed even if a previous task fails

*Drag tasks here to make them final*



**Script configuration**

Task description: Run Polyspace

☐ Disable this task

☐ Add condition to task

Interpreter: Shell

An interpreter is chosen based on the shebang line of your script.

Script location: Inline

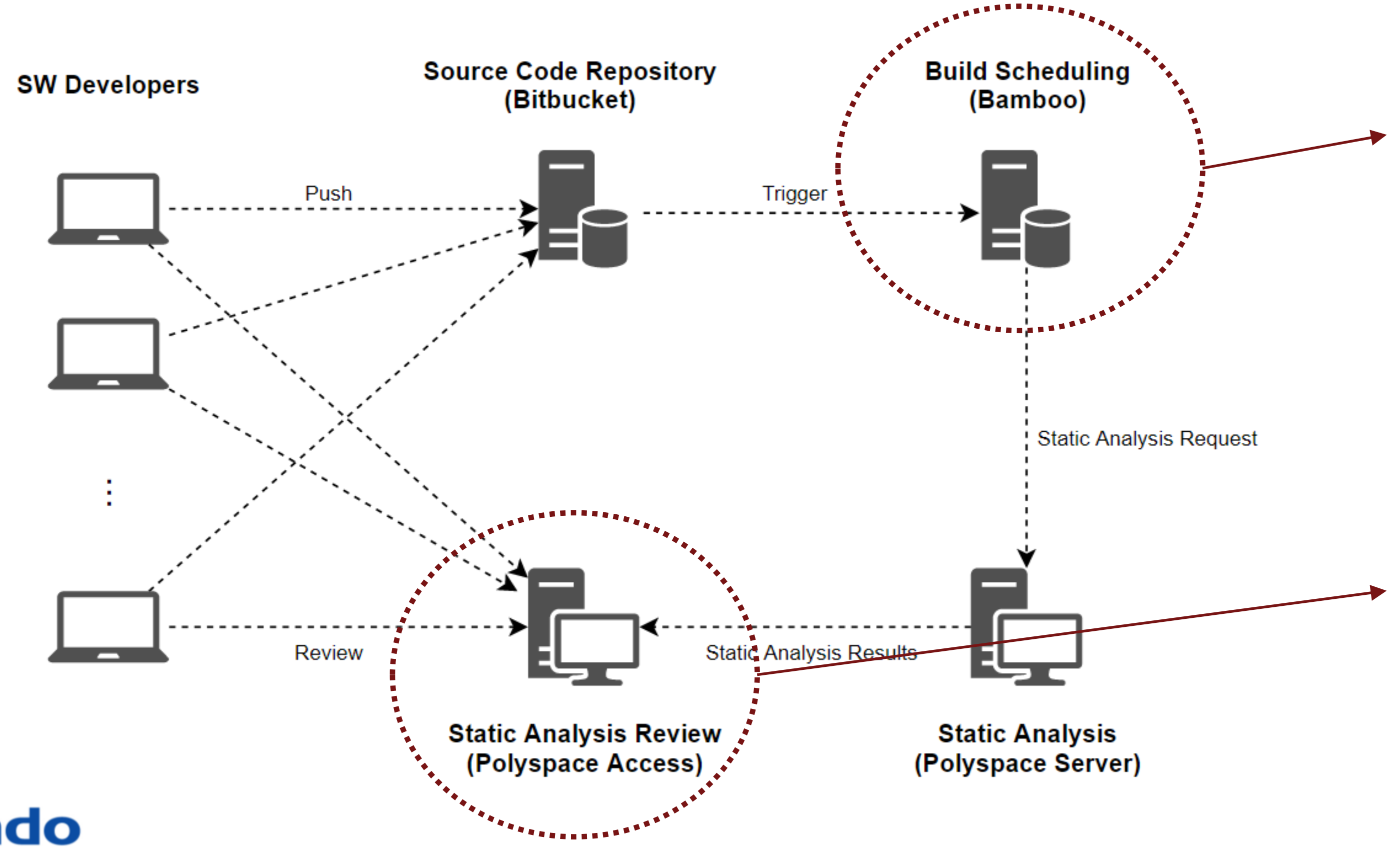
Script body*

```
SET branch_name=${bamboo.planRepository.branch}
SET project_name=%branch_name:/=-%
SET project_name=%project_name:(=-%
SET project_name=%project_name:)=%

${bamboo.polyspace_bugfinder_app} -prog %project_name% -options-file ./Polyspace/project_opt
${bamboo.polyspace_report_generator} -generate-results-list-file -results-dir ./polyspace/re
${bamboo.polyspace_report_generator} -template ${bamboo.polyspace_report_template_dir}/devel
${bamboo.polyspace_report_generator} -template ${bamboo.polyspace_report_template_dir}/bug_f
${bamboo.polyspace_access_app} -host ${bamboo.polyspace_server_ip} -port ${bamboo.polyspace_
${bamboo.polyspace_access_app} -host ${bamboo.polyspace_server_ip} -port ${bamboo.polyspace_
```

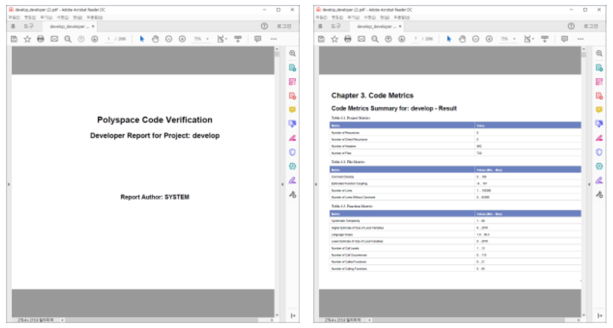
## Achievements and Outlook

### Web based result review and report generation

- Polyspace Server uploads the results to Polyspace Access
  - The generated documents are registered to bamboo server as an artifact.


SW Developers
Source Code Repository (Bitbucket)
Build Scheduling (Bamboo)
Push
Trigger
Static Analysis Request
Review
Static Analysis Results
Static Analysis Review (Polyspace Access)
Static Analysis (Polyspace Server)

**Static Analysis Report (For OEM and developers)**



Polyspace Code Verification

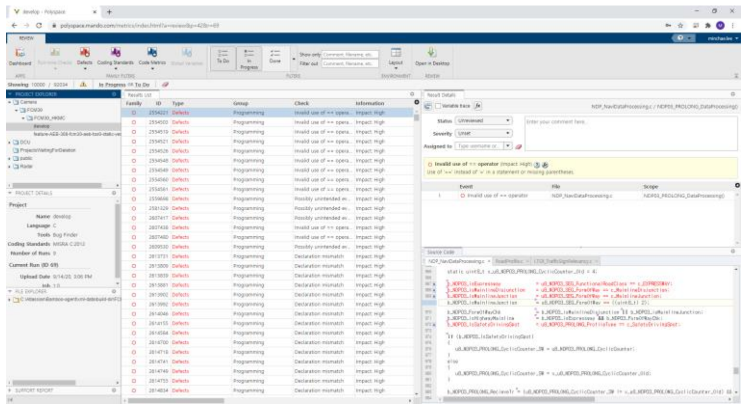
Developer Report for Project: develop

Report Author: SYSTEM

Chapter 3. Code Metrics

Code Metrics Summary for: develop - Result

**Web based static analysis review**



MATLAB EXPO

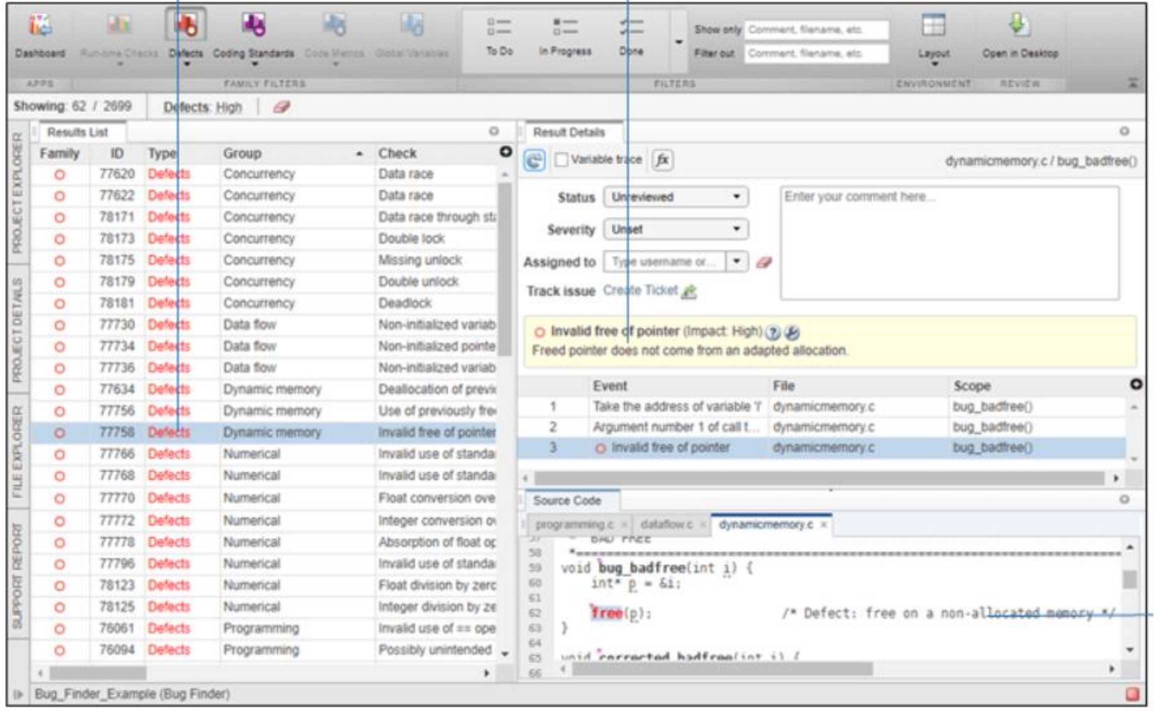
# Achievements and Outlook

## Web based result review and report generation

- Open web browser and go to ‘Polyspace Access’ web site

Select a result.

Read result explanation.



Showing: 62 / 2699 | Defects: High

Results List

| Family | ID | Type | Group | Check |
|---|---|---|---|---|
| ○ | 77620 | Defects | Concurrency | Data race |
| ○ | 77622 | Defects | Concurrency | Data race |
| ○ | 78171 | Defects | Concurrency | Data race through st |
| ○ | 78173 | Defects | Concurrency | Double lock |
| ○ | 78175 | Defects | Concurrency | Missing unlock |
| ○ | 78179 | Defects | Concurrency | Double unlock |
| ○ | 78181 | Defects | Concurrency | Deadlock |
| ○ | 77730 | Defects | Data flow | Non-initialized variab |
| ○ | 77734 | Defects | Data flow | Non-initialized pointe |
| ○ | 77736 | Defects | Data flow | Non-initialized variab |
| ○ | 77634 | Defects | Dynamic memory | Deallocation of previ |
| ○ | 77756 | Defects | Dynamic memory | Use of previously fre |
| ○ | 77758 | Defects | Dynamic memory | Invalid free of pointer |
| ○ | 77766 | Defects | Numerical | Invalid use of standa |
| ○ | 77768 | Defects | Numerical | Invalid use of standa |
| ○ | 77770 | Defects | Numerical | Float conversion ove |
| ○ | 77772 | Defects | Numerical | Integer conversion o |
| ○ | 77778 | Defects | Numerical | Absorption of float op |
| ○ | 77796 | Defects | Numerical | Invalid use of standa |
| ○ | 78123 | Defects | Numerical | Float division by zerc |
| ○ | 78125 | Defects | Numerical | Integer division by ze |
| ○ | 76061 | Defects | Programming | Invalid use of == ope |
| ○ | 76094 | Defects | Programming | Possibly unintended |

Bug_Finder_Example (Bug Finder)

Result Details

Variable trace | dynamicmemory.c / bug_badfree()

Status: Unreviewed

Severity: Unset

Assigned to: Type username or...

Track issue: Create Ticket

Enter your comment here...

○ Invalid free of pointer (Impact: High)
Freed pointer does not come from an adapted allocation.

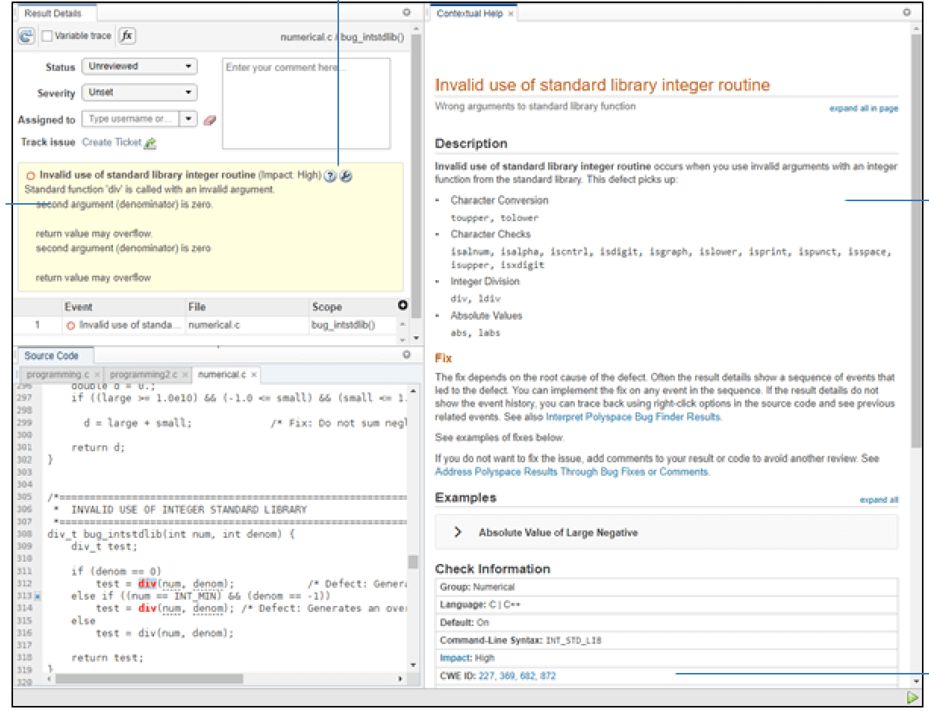
|  | Event | File | Scope |
|---|---|---|---|
| 1 | Take the address of variable 'i' | dynamicmemory.c | bug_badfree() |
| 2 | Argument number 1 of call t... | dynamicmemory.c | bug_badfree() |
| 3 | ○ Invalid free of pointer | dynamicmemory.c | bug_badfree() |

Source Code

programming.c × | dataflow.c × | dynamicmemory.c ×

```
 * BAD FREE
 *=================================================
void bug_badfree(int i) {
    int* p = &i;

    free(p);                    /* Defect: free on a non-allocated memory */
}

void corrected_badfree(int i) {
```

Open contextual help.



Result Details

Variable trace | numerical.c / bug_intstdlib()

Status: Unreviewed

Severity: Unset

Assigned to: Type username or...

Track issue: Create Ticket

Enter your comment here...

○ Invalid use of standard library integer routine (Impact: High)
Standard function 'div' is called with an invalid argument.
second argument (denominator) is zero.

return value may overflow.
second argument (denominator) is zero

return value may overflow

|  | Event | File | Scope |
|---|---|---|---|
| 1 | ○ Invalid use of standa... | numerical.c | bug_intstdlib() |

Source Code

programming.c × | programming2.c × | numerical.c ×

```
    double d = 0.;
    if ((large >= 1.0e10) && (-1.0 <= small) && (small <= 1.

      d = large + small;            /* Fix: Do not sum negl

    return d;
}


/*=====================================================
 *  INVALID USE OF INTEGER STANDARD LIBRARY
 *=====================================================
div_t bug_intstdlib(int num, int denom) {
    div_t test;

    if (denom == 0)
        test = div(num, denom);            /* Defect: Genera
    else if ((num == INT_MIN) && (denom == -1))
        test = div(num, denom); /* Defect: Generates an ove
    else
        test = div(num, denom);

    return test;
}
```

Contextual Help ×

Invalid use of standard library integer routine

Wrong arguments to standard library function — expand all in page

**Description**

**Invalid use of standard library integer routine** occurs when you use invalid arguments with an integer function from the standard library. This defect picks up:

- Character Conversion
  `toupper, tolower`
- Character Checks
  `isalnum, isalpha, iscntrl, isdigit, isgraph, islower, isprint, ispunct, isspace, isupper, isxdigit`
- Integer Division
  `div, ldiv`
- Absolute Values
  `abs, labs`

**Fix**

The fix depends on the root cause of the defect. Often the result details show a sequence of events that led to the defect. You can implement the fix on any event in the sequence. If the result details do not show the event history, you can trace back using right-click options in the source code and see previous related events. See also Interpret Polyspace Bug Finder Results.

See examples of fixes below.

If you do not want to fix the issue, add comments to your result or code to avoid another review. See Address Polyspace Results Through Bug Fixes or Comments.

**Examples** — expand all

> Absolute Value of Large Negative

**Check Information**

Group: Numerical
Language: C | C++
Default: On
Command-Line Syntax: INT_STD_LIB
Impact: High
CWE ID: 227, 369, 682, 872

Read detailed explanation with examples.

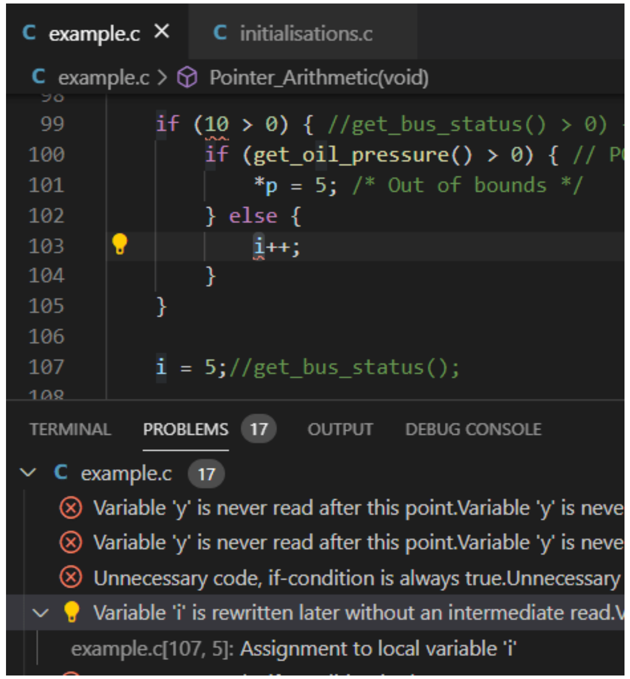
Check external standards.

Mando

## Future Works

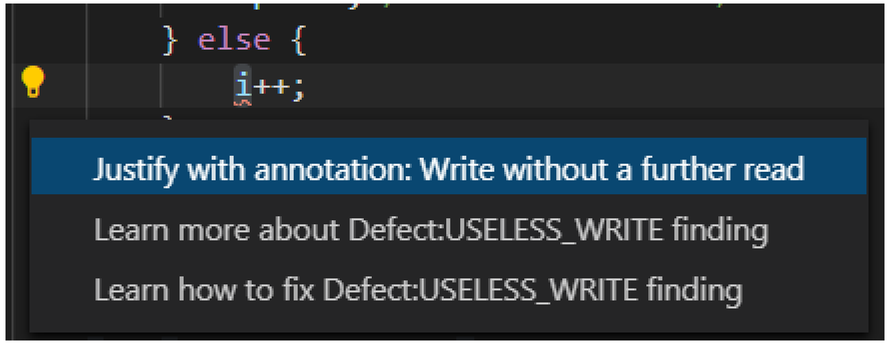
### Polyspace as You Code

- Polyspace as You Code is a Visual Studio Code extension
  - Run a single-file analysis on software developer’s computer
  - Analysis results appears on VS Code window



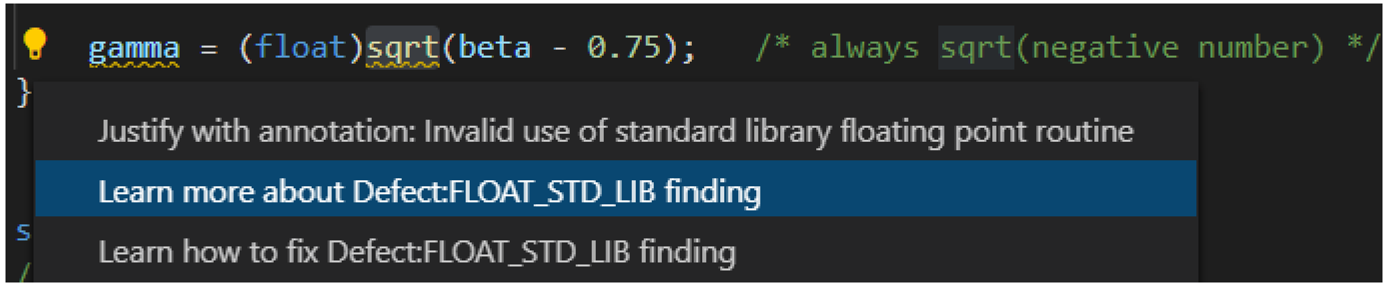
```
 99        if (10 > 0) { //get_bus_status() > 0)
100            if (get_oil_pressure() > 0) { // P
101                *p = 5; /* Out of bounds */
102            } else {
103                i++;
104            }
105        }
106
107        i = 5;//get_bus_status();
```

```
C example.c 17
  Variable 'y' is never read after this point.Variable 'y' is neve
  Variable 'y' is never read after this point.Variable 'y' is neve
  Unnecessary code, if-condition is always true.Unnecessary
  Variable 'i' is rewritten later without an intermediate read.V
    example.c[107, 5]: Assignment to local variable 'i'
```



```
    } else {
        i++;
```

Justify with annotation: Write without a further read
Learn more about Defect:USELESS_WRITE finding
Learn how to fix Defect:USELESS_WRITE finding



```
    gamma = (float)sqrt(beta - 0.75);    /* always sqrt(negative number) */
}
```

Justify with annotation: Invalid use of standard library floating point routine
Learn more about Defect:FLOAT_STD_LIB finding
Learn how to fix Defect:FLOAT_STD_LIB finding

Mando

## Conclusions

- Pros
  - Polyspace can analyze MISRA and defects at once
  - Various interface to analyze/review static analysis results
  - No additional costs for many lines of code or component extension

- Cons
  - Customization of MISRA rules categories such as Mandatory, Required, and Advisory
  - More detailed configuration for interrupts priority in Multitasking
  - Slow Polyspace Code Prover makes adoption difficult

## Conclusions

- Improved development process with Polyspace and CI
  - Bamboo and Polyspace based SW static analysis is applied
    - Static analysis script runs automatically after code push

- Collaborative review on web site
  - Software developers can review the results on Polyspace Access web site
    - Analysis reports are generated for developers and OEMs
  - Findings can be assigned to relevant person on JIRA
    - JIRA issues can be created in Polyspace Access

## Questions?