

MATLAB EXPO

코드 검증을 위한 Continuous Integration 활용 방안
유용출, MathWorks Korea



Agenda

- Polyspace Products for Static Analysis
- Polyspace in Continuous Integration (CI)
- Automate Polyspace Static Analysis by Continuous Integration Tool
- Collaborative Review Environment with Polyspace and 3rd Party Tool

Agenda

- **Polyspace Products for Static Analysis**
- Polyspace in Continuous Integration (CI)
- Automate Polyspace Static Analysis by Continuous Integration Tool
- Collaborative Review Environment with Polyspace and 3rd Party Tool

Polyspace Tools

Bug Finder

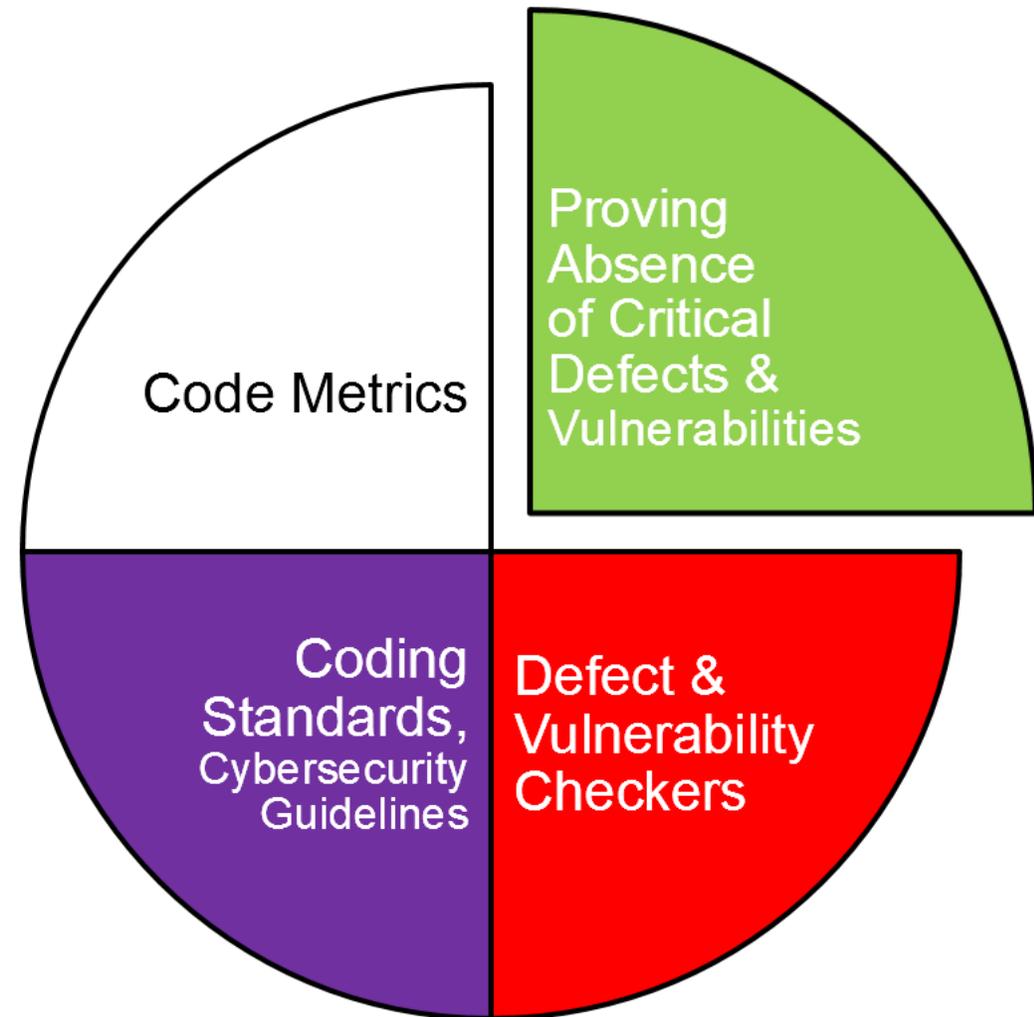


- Produce code metrics
- Check coding standards
- Find defects and vulnerabilities

Code Prover

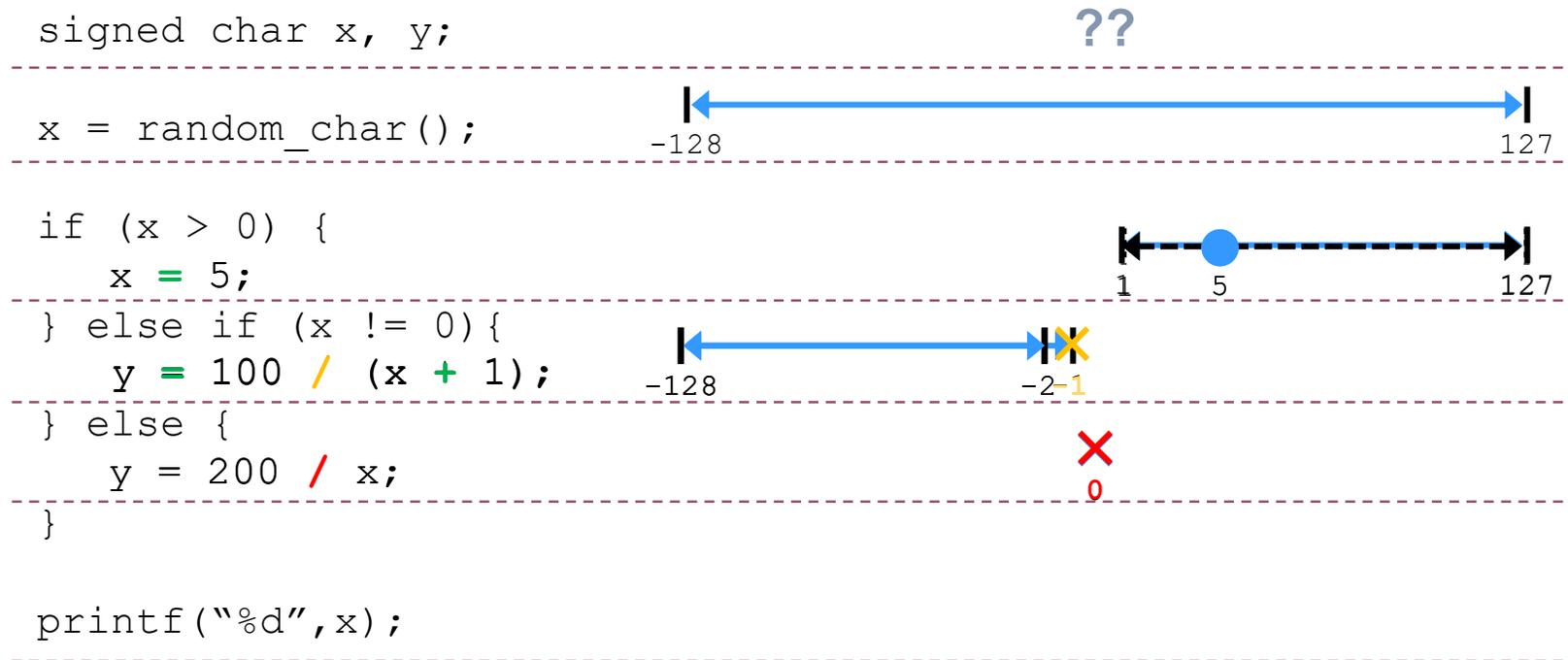


- Proves code Safe and Secure
- 33 most critical run-time checks
- Supports DO-178 and ISO 26262 mori



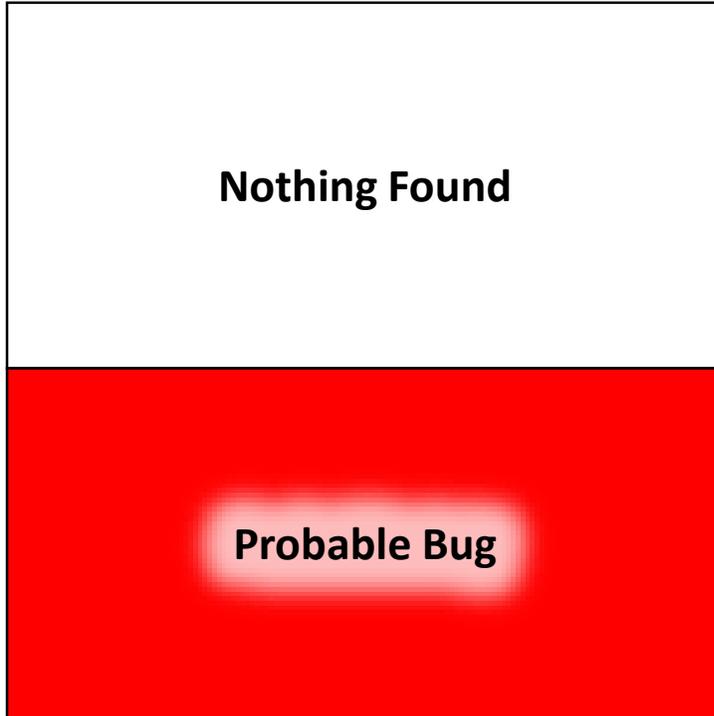
Formal Methods Prove Absence of Runtime Errors in your SW

- To prove the absence of errors, the Polyspace verification accounts for all possible execution paths using abstract interpretation.



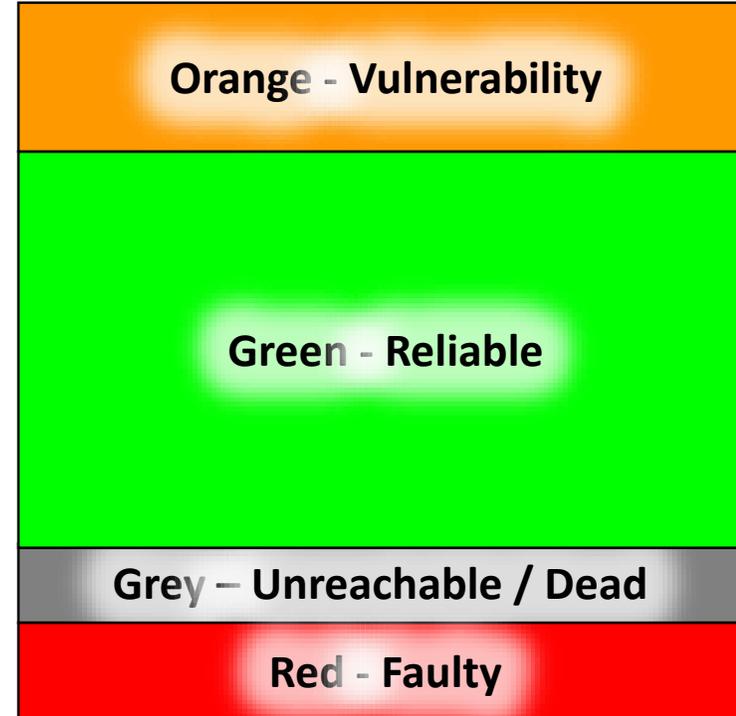
How do Bug Finder results differ from Code Prover results?

Bug Finder



vs.

Code Prover

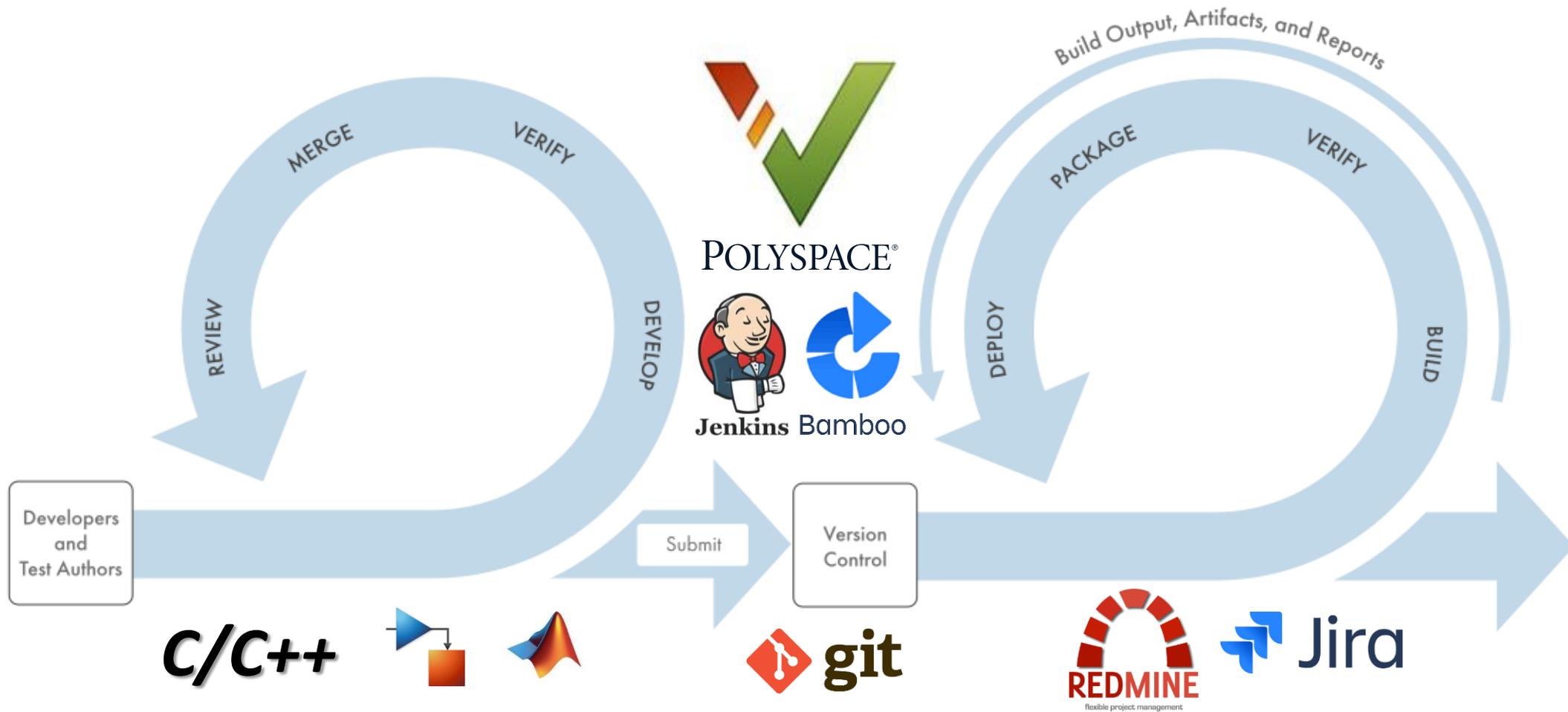


Purple - coding rule violations

Agenda

- Polyspace Products for Static Analysis
- **Polyspace in Continuous Integration (CI)**
- Automate Polyspace Static Analysis by Continuous Integration Tool
- Collaborative Review Environment with Polyspace and 3rd Party Tool

Continuous Integration (CI)



Common Questions to Adopt Polyspace in Continuous Integration

- ✓ Who manages multiple servers?
- ✓ No time to invest in static analysis?
- ✓ How to manage so many issues?
- ✓ How to configure analysis jobs in CI?
- ✓ How to share analysis results?



Common Questions to Adopt Polyspace in Continuous Integration

Who manages multiple servers?

**Technical
Difficulties**



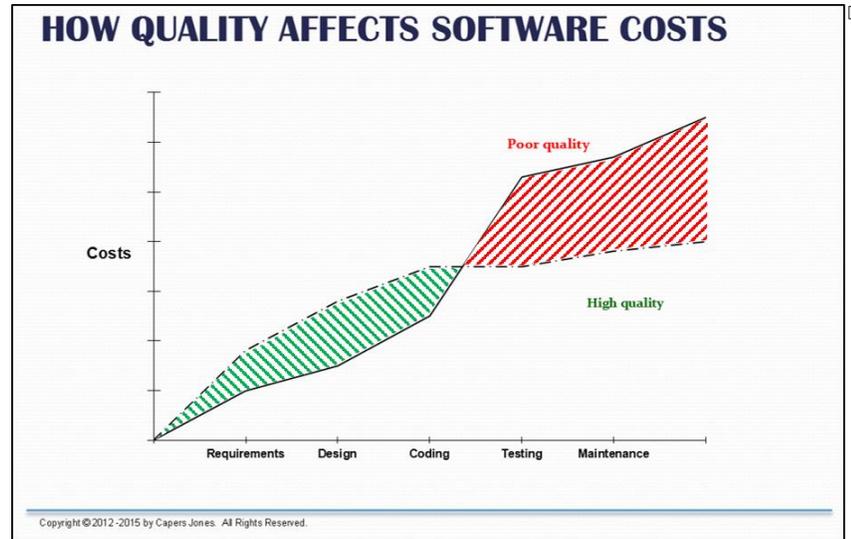
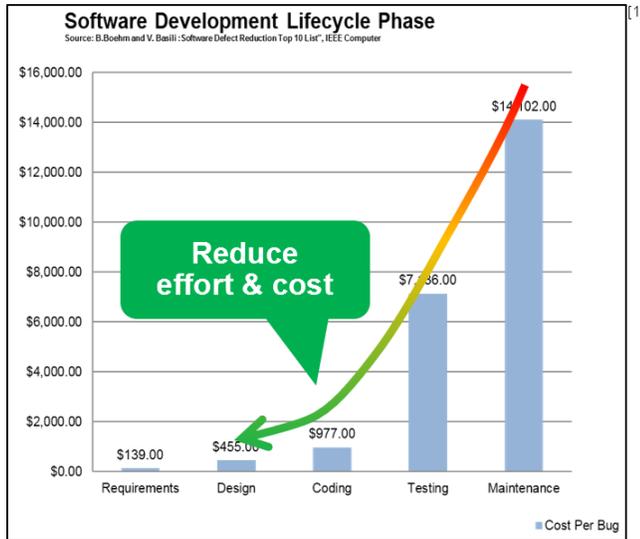
Cost



How to share analysis results?

Resolve Challenges to Adopt Polyspace in Continuous Integration

- Abundant information helps to lower technical hurdle
 - Youtube, Blogs, Articles and open communities in the internet
 - 3rd party tools provide plugins and guidelines
- CI and Static Analysis helps cost reduction



WHAT IS JENKINS? JENKINS TUTORIAL 19:53

JENKINS FULL COURSE IN 3 HOURS 2:56:07

Redmine Tutorial 18:55

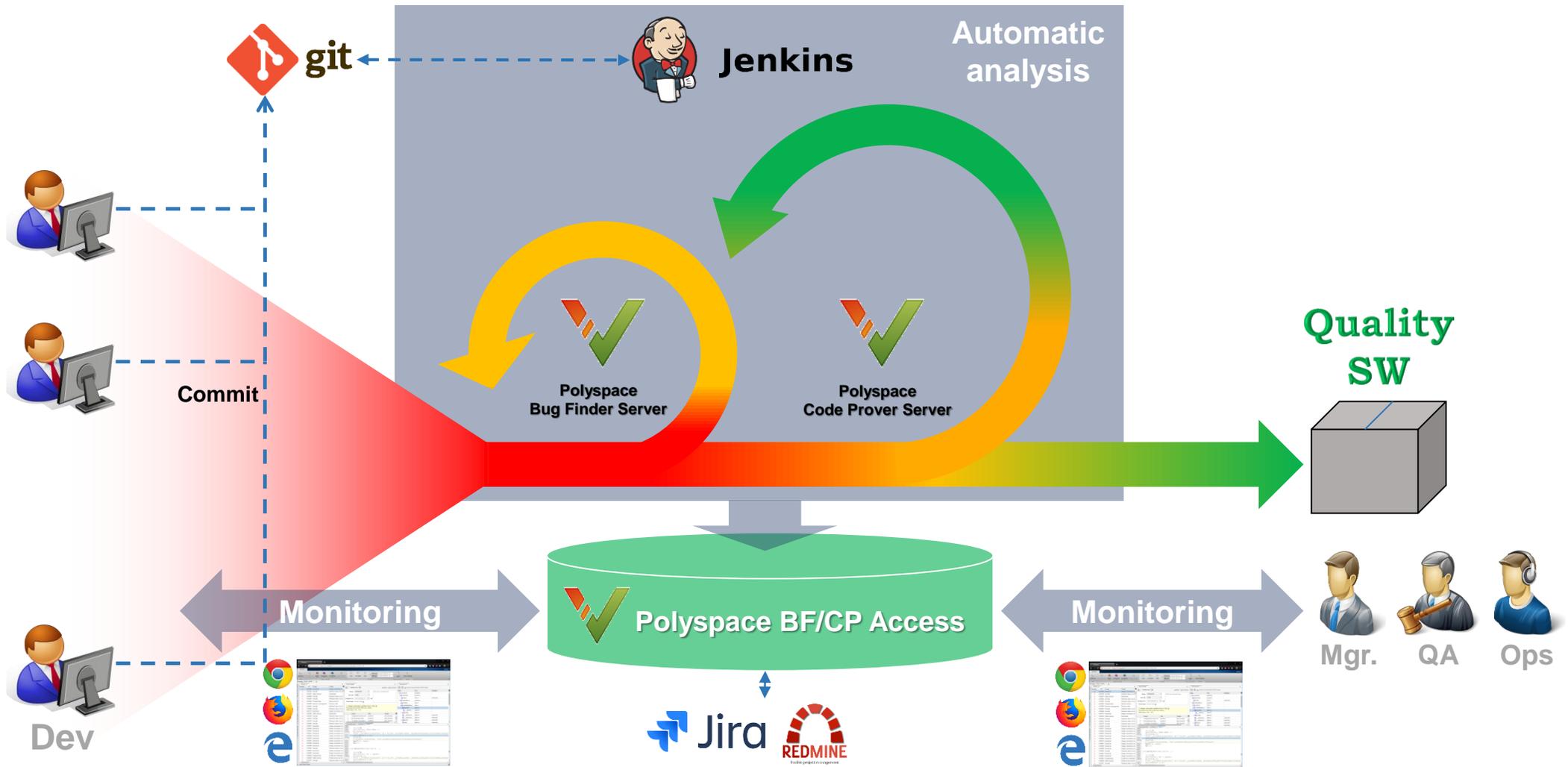
레드마인(redmine) 생활코딩 • 조회수 8.7천 http://opentutorials.org/

[1] Barry Boehm, "Software Defect Reduction Top 10 List", IEEE
 [2] Capers Jones, "Software Quality in 2016: A Survey of the state of the art"
 [3] captured from Youtube

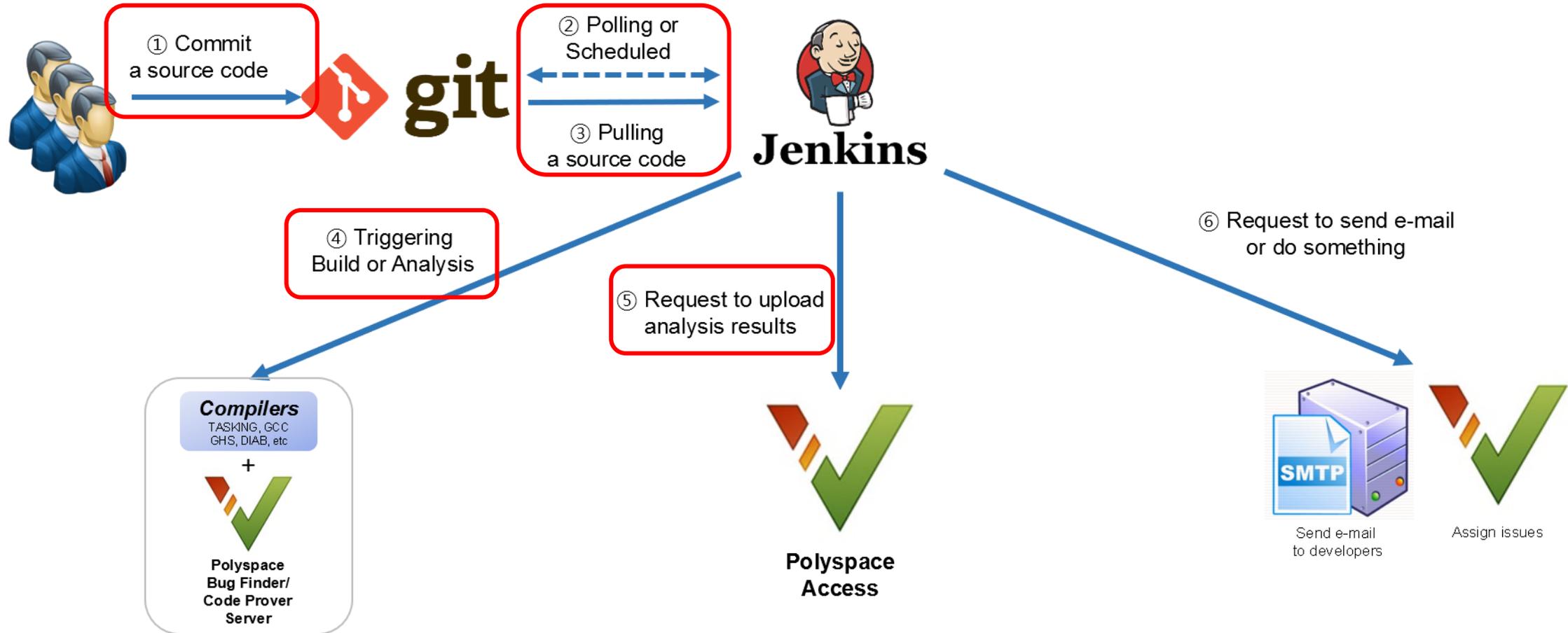
Agenda

- Polyspace Products for Static Analysis
- Polyspace in Continuous Integration (CI)
- **Automate Polyspace Static Analysis by Continuous Integration Tool**
- Collaborative Review Environment with Polyspace and 3rd Party Tool

Automatic Polyspace Analysis after Commit and Review Environment



CI tools can automate Polyspace analysis job



Configure for Source Code Repository and Triggers

Configure for source code management

- Source code repository type
- Account information
- Branch

Source Code Management

- None
- Git
- Mercurial
- Subversion

Usually, configure for triggers

- Analyze periodically
- Analyze when source codes are changed

Build Triggers

- Trigger builds remotely (e.g., from scripts)
- Build after other projects are built
- Build periodically
- Build when a change is pushed to BitBucket
- GitHub hook trigger for GITScm polling
- Poll SCM

Start Static Analysis after Pulling Source Code and Upload the Result

Configure commands for analysis

- Project creation
- Analysis option configuration
- Analysis for source codes

Build or Analysis

Request to upload analysis results

Configure commands to upload

- Project creation for the web interface
- Upload analysis results
- Assign results to owners



```
Execute shell

Command #Clean project
make clean

#Create a proeject automatically from build command
polyspace-configure -allow-override -allow-build-error -prog "BF_Resu

#Run Polyspace Bug Finder
polyspace-bug-finder-server -options-file "BF_Result.psopts" -options-

Just 7 ~ 8 lines

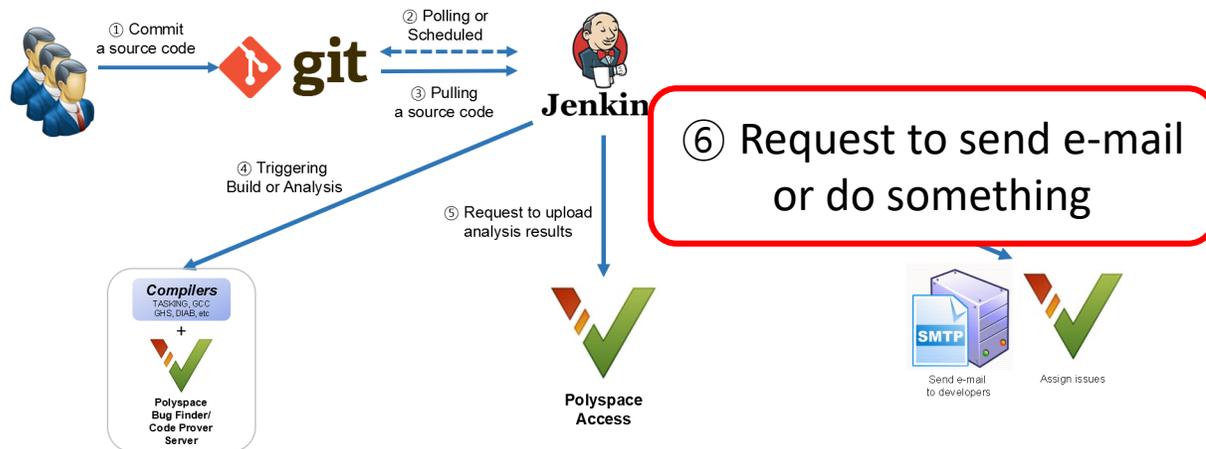
Command #Create project folders on Polyspace Access
$ps_helper_access -create-project "Team1"
$ps_helper_access -create-project "Team1/Project1"

#Upload a result to Polyspace Access
mkdir Notification
$ps_helper_access -upload ./BF_Result/ -project "BF_Result" -parent-pr

#Assign owners to unassigned issues
$ps_helper_access -set-unassigned-findings "Team1/Project1/BF_Result"
```

See [the list of available environment variables](#)

Send an E-mail about Summarized Information



Post-build Actions

Polyspace Notification

Send common e-mail

E-mail usernames

Attachment filename

Mail subject filename

Mail body filename

Send different e-mails to individual recipients

* Can customize e-mail template

Agenda

- Polyspace Products for Static Analysis
- Polyspace in Continuous Integration (CI)
- Automate Polyspace Static Analysis by Continuous Integration Tool
- **Collaborative Review Environment with Polyspace and 3rd Party Tool**

Polyspace Access - Web Interface for Collaborative Review

Summary

Open Issues

Open: 2053

New

Assigned To Me

Unassigned

Defects

Density: 11

Summary Information

Result Review

Family	ID	Type	Group	Check	Im
○	40374	Defects	Dynamic memory	Deallocation of previous...	Im
○	40426	Defects	Data flow	Non-initialized variable	Im
○	40433	Defects	Data flow	Non-initialized pointer	Im
○	40437	Defects	Data flow	Non-initialized variable	Im
○	40467	Defects	Dynamic memory	Use of previously freed ...	Im
○	40470	Defects	Dynamic memory	Invalid free of pointer	Im
○	40445	Defects	Data flow	Useless if	Im
○	40453	Defects	Data flow	Unreachable code	Im
○	35239	Defects	Data flow	Variable shadowing	Im
○	35250	Defects	Data flow	Missing return statement	Im
○	40430	Defects	Data flow	Partially accessed array	Im
○	40441	Defects	Data flow	Write without a further r...	Im
○	40449	Defects	Data flow	Write without a further r...	Im
○	40457	Defects	Data flow	Dead code	Im
○	40460	Defects	Data flow	Partially accessed array	Im
○	40464	Defects	Dynamic memory	Alignment changed afte...	Im
▽	35235	MISRA C:2012	17 Functions	17.4 All exit paths from ...	Ca
▽	40398	MISRA C:2012	22 Resources	22.2 A block of memory ...	Ca
▽	40683	MISRA C:2012	22 Resources	22.2 A block of memory ...	Ca
▽	40707	MISRA C:2012	9 Initialization	9.1 The value of an obje...	Ca
▽	40725	MISRA C:2012	9 Initialization	9.1 The value of an obje...	Ca
▽	40745	MISRA C:2012	9 Initialization	9.1 The value of an obje...	Ca
▽	31107	MISRA C:2012	21 Standard libraries	21.1 #define and #undef...	Ca
▽	31117	MISRA C:2012	Dir 4 Code design	D4.12 Dynamic memory...	Ca
▽	31118	MISRA C:2012	8 Declarations and defin	8.5 An external object or	Ca

Code Metrics: Number of Sub-Projects

Variable trace: Show In Re:

Status: Unreviewed

Severity: Unset

Assigned to: ylee

Ticket:

○ Deallocation of previously deallocated pointer (Impact: High) ? ⓘ
 Pointer is already deallocated.

	Event	File	Scope
1	Dynamic allocation	dynamicmemory.c	bug_doubledeallocation()
2	Assignment to local pointer 'pi'	dynamicmemory.c	bug_doubledeallocation()
3	Call to 'free'	dynamicmemory.c	bug_doubledeallocation()
4	○ Deallocation of previous...	dynamicmemory.c	bug_doubledeallocation()

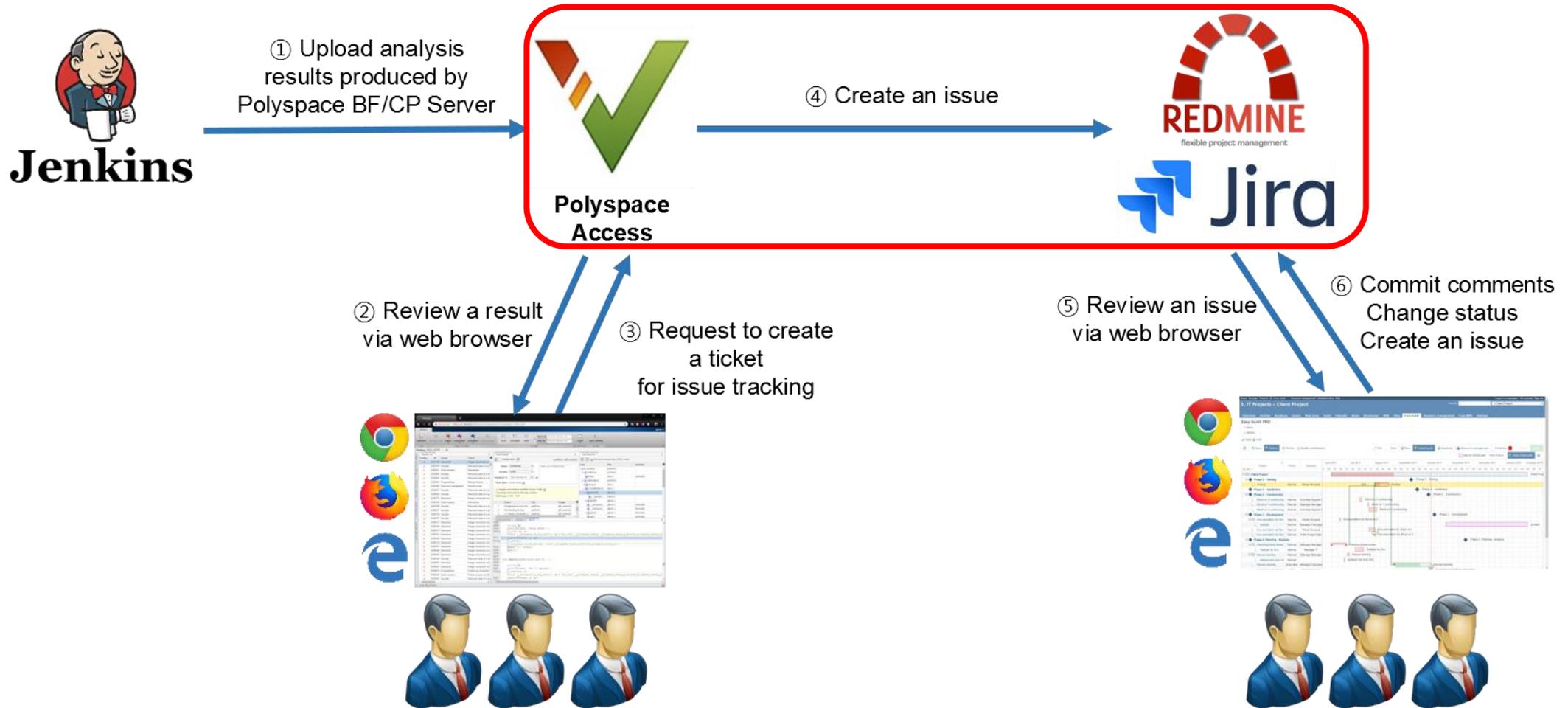
Source Code

```

programming.c x staticmemory.c x dynamicmemory.c x cryptography.c x
84  *pi = 2;
85  (void)printf("Freeing the pointer\n");
86  free(pi); /* good practice violation (missing reset
87           * freed pointer) prior to double free
88  (void)printf("Freeing the pointer\n");
89  free(pi); /* Defect: pi has already been freed
90  }
91
92  void corrected_doubledeallocation(void) {
93

```

Collaborative Review Environment



Open Analysis Result from E-mail

Email can provide ...

- Alarm about completion status
- Brief information about analysis result
- Links to analysis results

Gary Ryu	Finished Polyspace analysis Job - Defect: 3, MISRA: 14 Summary information about the Jenkins Job BFAAnalysis - 16
Gary Ryu	Finished Polyspace analysis Job - Red: 1, Gray: 1, Orange: 9 MISRA: 26 Summary information about the Jenkins Job CPAAnalysis - 11
Gary Ryu	Finished Polyspace analysis Job - Defect: 0, MISRA: 2 Summary information about the Jenkins Job BFAAnalysis - 15
Gary Ryu	Finished Polyspace analysis Job - Defect: 0, MISRA: 2 Summary information about the Jenkins Job BFAAnalysis - 14
Gary Ryu	Finished Polyspace analysis Job - Defect: 3, MISRA: 14 Summary information about the Jenkins Job BFAAnalysis - 13
Gary Ryu	Finished Polyspace analysis Job - Red: 1, Gray: 1, Orange: 9 MISRA: 26 Summary information about the Jenkins Job CPAAnalysis - 10
Gary Ryu	Finished Polyspace analysis Job - Defect: 3, MISRA: 14 Summary information about the Jenkins Job BFAAnalysis - 12

Easy to Understand Overall Polyspace Analysis Results

Dashboards provide ...

- Overview of analysis results
- Graphical charts
- Clickable fields to drill down into findings

The screenshot displays the Polyspace dashboard interface. At the top, a navigation bar includes a 'DASHBOARD' tab and a menu with options: 'Project Overview', 'Defects', 'Code Metrics', 'MISRA C:2012', and 'Quality Objectives'. Below this is a 'PROJECT EXPLORER' pane on the left, showing a tree view with folders like 'ProjectsWaitingForDeletion', 'public', and 'Team1', and sub-items 'BF_Result' and 'CP_Result'. The main area is titled 'Project Overview' and contains a 'Summary' section with 'Open Issues' (17 Open, 15 New, 0 Assigned To Me, 17 Unassigned) and a 'Defects' section with a large orange gauge and '3 To Do' items. A red box highlights the top navigation menu.

Collaborative Review Environment

Result Review provide ...

- Results List
- Result Details
- Source Code
- Call Hierarchy
- Various Filters
- Review Comments
- Issue Assignment
- Ticket Creation to Issue Tracking Tool

The screenshot displays the MATLAB Collaborative Review Environment dashboard. The top navigation bar includes 'Project Overview', 'Defects', 'Code Metrics', 'MISRA C:2012', and 'Quality Objectives'. The 'Review' button is highlighted with a red box. The main content area shows a 'PROJECT EXPLORER' on the left with a tree view of folders and files. The right pane displays a 'Summary' section with an 'Open Issues' table. The table has columns for issue status, counts, and compliance levels. A red box highlights the counts in the 'Open Issues' table.

		Mandatory	Required	Advi
Open	14	Done 0%	0%	100%
New	12	Open 2	12	0
Assigned To Me	0	0% Done		0/14 viola
Unassigned	14	9 rules violated (133 rules enabled)		

Issue Tracking between Polyspace Access and 3rd Party Tools

Result Review provide ...

- Results List
- Result Details
- Source Code
- Call Hierarchy
- Various Filters
- Review Comments
- Issue Assignment
- Ticket Creation to Issue Tracking Tool

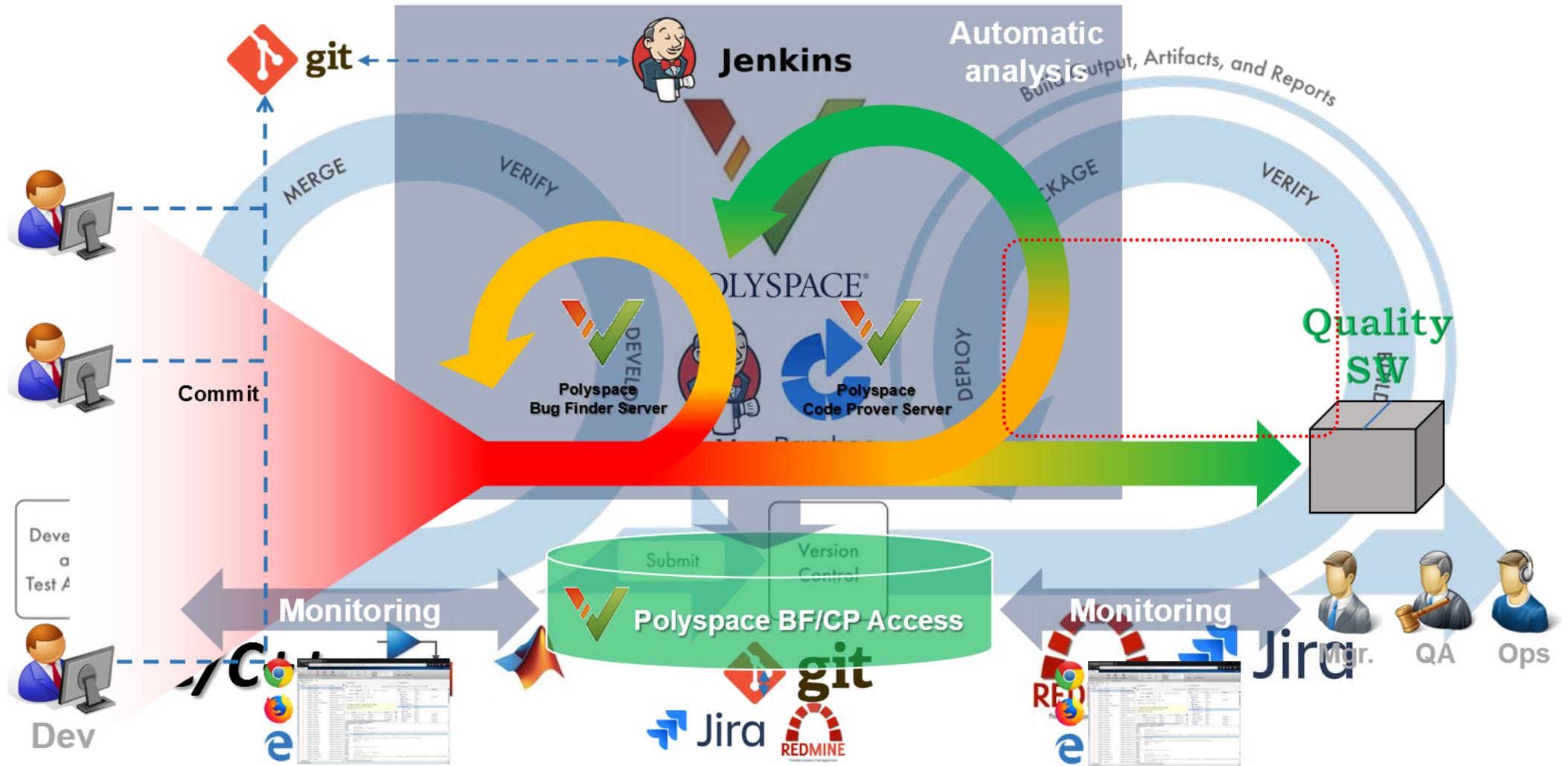
The screenshot displays the Polyspace interface for issue tracking. The top window shows the 'Results List' for the file 'seatbelt_reminder.c'. It contains a table with columns: Family, ID, Type, Group, and Check. The table lists various MISRA C:2012 rules and code metrics.

Family	ID	Type	Group	Check
▼ *	640680	MISRA C:2012	10 The essential type m...	10.4 Both operands of a
▼ *	640681	MISRA C:2012	10 The essential type m...	10.1 Operands shall not
▼ *	640682	MISRA C:2012	10 The essential type m...	10.1 Operands shall not
▼ *	640683	MISRA C:2012	10 The essential type m...	10.1 Operands shall not
▼ *	640684	MISRA C:2012	16 Switch statements	16.4 Every switch state.
▼ *	640685	MISRA C:2012	16 Switch statements	16.1 All switch statemen
▼	640742	MISRA C:2012	Dir 4 Code design	D4.14 The validity of val
★	640703	Code Metrics	Function Metrics	Higher Estimate of Size
★	640706	Code Metrics	Function Metrics	Language Scope
★	640707	Code Metrics	Function Metrics	Number of Call Levels
★	640708	Code Metrics	Function Metrics	Number of Local Non-St
★	640709	Code Metrics	Function Metrics	Number of Instructions
★	640712	Code Metrics	Function Metrics	Number of Function Par
★	640714	Code Metrics	Function Metrics	Number of Called Funct
★	640716	Code Metrics	Function Metrics	Number of Calling Func.
★	640717	Code Metrics	Function Metrics	Number of Local Static .
★	640718	Code Metrics	Function Metrics	Cyclomatic Complexity
★	640720	Code Metrics	Function Metrics	Number of Call Occurre.
★	640721	Code Metrics	Function Metrics	Number of Return State
★	640723	Code Metrics	Function Metrics	Number of Paths
★	640724	Code Metrics	File Metrics	Estimated Function Cou
★	640726	Code Metrics	Function Metrics	Lower Estimate of Size .
★	640727	Code Metrics	File Metrics	Comment Density
★	640729	Code Metrics	Function Metrics	Number of Executable L
★	640730	Code Metrics	Function Metrics	Number of Lines Within

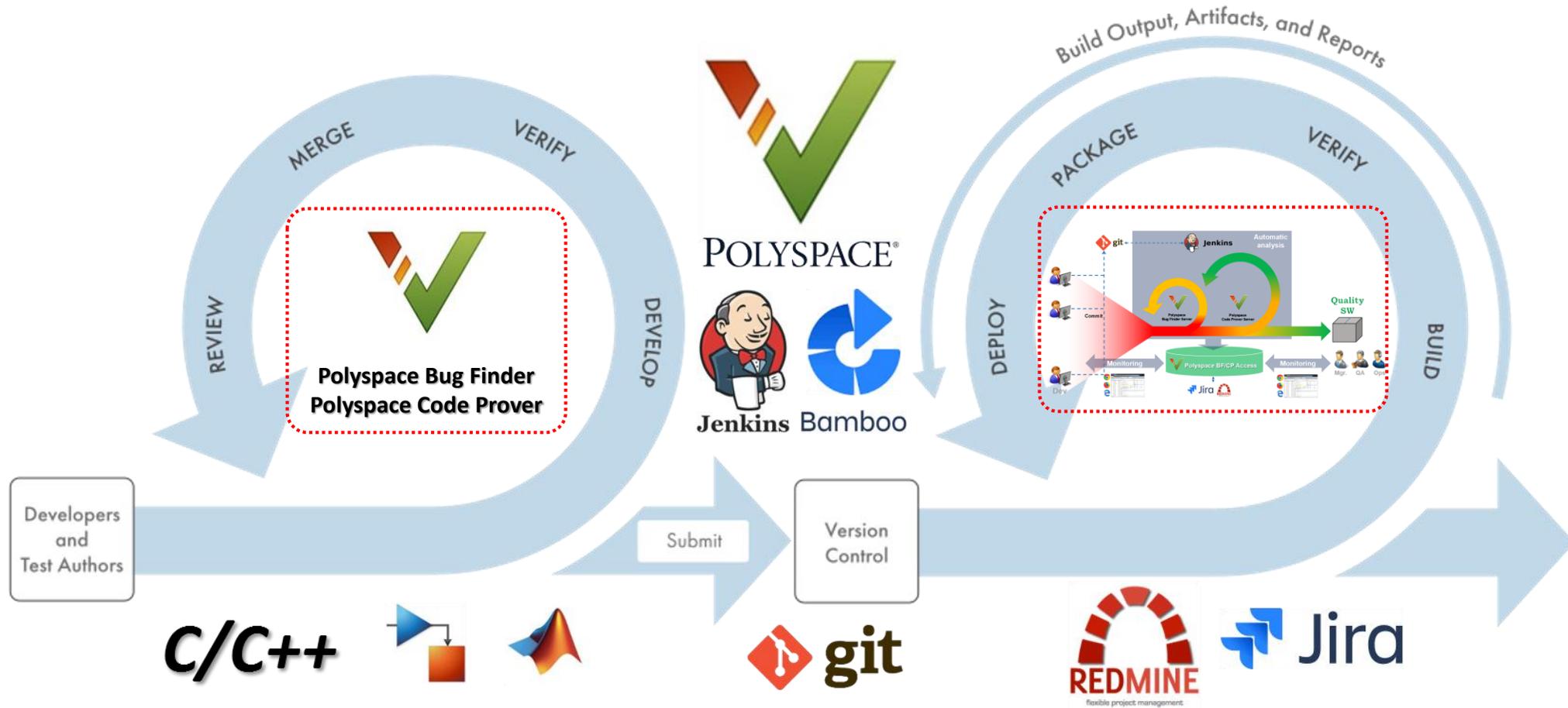
The 'Result Details' window on the right shows the status of a selected issue: MISRA C:2012 10.4 (Required). The status is 'Unreviewed' and severity is 'Unset'. It includes a text area for comments and buttons for 'Create' and 'Attach' tickets. Below this, there is a table with columns 'Event' and 'File'. At the bottom, the 'Source Code' window shows the C code snippet where the issue was detected:

```
49 11 (((uint16_t)inputs&0x00ff) == 1) {
50     input_SeatBeltFasten = 1;
51 } else {
52     input_SeatBeltFasten = 0;
53 }
54
55 switch (((uint16_t)inputs&0xFF00)>>8) {
56 case 0:
57     input_KEY = 0;
58     break;
59 case 1:
60     input_KEY = 1;
```

Polyspace in Continuous Integration (CI)



Completed Workflow for Static Analysis with Polyspace



<https://kr.mathworks.com/videos/code-verification-for-cc-with-polyspace-121361.html>

Thank You !!