

Which Configuration Option Should I Change?

Sai Zhang, Michael D. Ernst
University of Washington

Presented by: K1vanç Muşlu



Developers

I have released a new software version ...



I cannot get used to the UI

I do not know how to configure it



Users

...

Diagnosis of User-Fixable Software Errors

- **Goal:**
 - enable users to fix software errors
- **Challenges:**
 - Errors can be crashing or non-crashing
 - Users much less understand source code
 - Developer tools are of little use



A new software version

Our previous work [ISSTA'13]
Help users adapt to the new UI

I cannot get used to the UI



Users



A new software version

**I do not know how to
configure it**



Users

This paper:

***How to help users configure
the new software version***

(i.e., diagnosis of configuration errors)

Software system often requires *configuration*



Configuration errors:

- Users use wrong values for options
- The software exhibits unintended behaviors

```
#
# * Fine Tuning
#
key_buffer      = 16M
max_allowed_packet = 16M
thread_stack    = 192K
thread_cache_size = 8
#
# * Query Cache Configuration
#
query_cache_limit = 1M
query_cache_size  = 16M
#
```

Example:

`--port_num = 100.0`

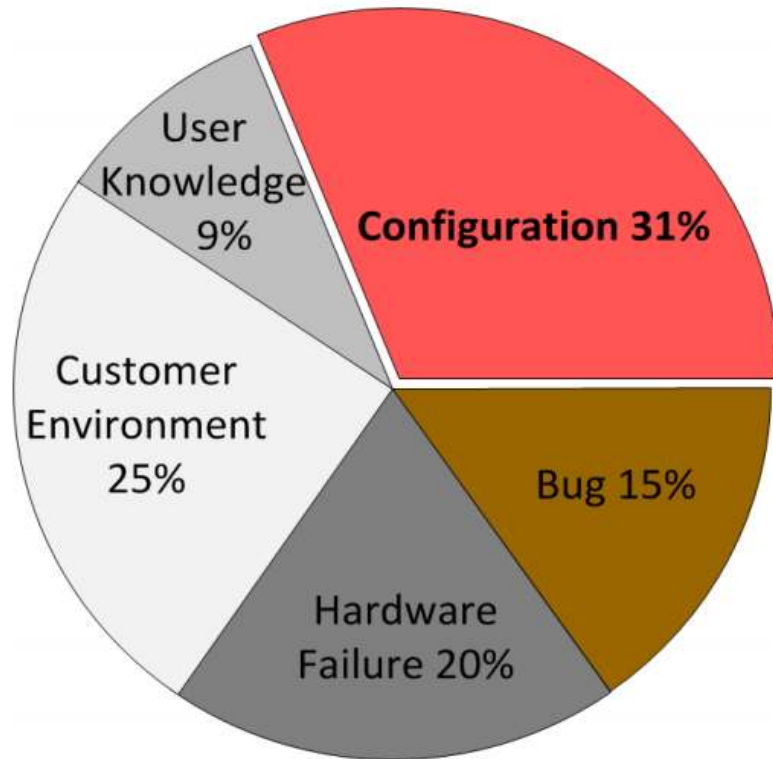


Should be a valid integer

```
Options include:
--help          Display this help text
-v or --verbose Print messages about ISendfile actions
-s             Silent, opposite of verbose
-h            No banner for this job
-F=format      Format is one of the following:
    f - formatted, l - leave control characters, o - Postscript
    p - use 'pr' format, r - FORTRAN, c - CIF, d - dvi, g - plot
    n - ditroff, t - troff, v - raster
-C=class      Class is used on banner page; up to 31 characters
-T=title      Job title
```



Configuration errors are *common and severe*



Root causes of **high-severity** issues in a major storage company [Yin et al, SOSP'11]



Configuration errors can have **disastrous** impacts (downtime costs **3.6%** of revenue)

Configuration errors are *difficult* to diagnose

- Error messages are **absent** or **ambiguous**
 - e.g.,

Server Error in '/' Application.
Description: An error occurred during the parsing of a resource

(after setting `--port_num = 100.0` in webs server)
- **Infeasible** to automatically search for a good configuration
 - Need to know the spec of a valid configuration option value (e.g., regex, date time, integer value range)
 - Huge search space
 - Need to specify a testing oracle for automation
- **Cannot** directly use existing debugging techniques
[Zhang et al., ICSE'13]

Goal: diagnosing configuration errors for *evolving* software



Old version



New version



a **different** output



Requires configuration!



To maintain the desired behavior on the new version
Which configuration option should I change?

Diagnosing configuration errors with ConfSuggester

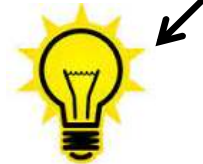


Old version



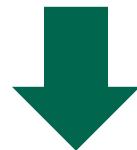
New version

a **different** output



Key idea:

The execution trace on the old version as the “intended behavior”



Our technique: **ConfSuggester**



Suspicious configuration options

Design constraints for ConfSuggester

- **Accessible**: no assumption about user background (e.g., users cannot read or write code annotations)
- **Easy-to-use**: fully automated
- **Portable**: no changes to OS or runtime environment
- **Accurate**: few false positives

Outline

- Example
- A Study of Configuration Evolution
- The ConfSuggester Technique
- Evaluation
- Related Work
- Contributions

Outline

- • Example
- A Study of Configuration Evolution
- The ConfSuggester Technique
- Evaluation
- Related Work
- Contributions



A popular performance testing tool

The screenshot shows a Microsoft Internet Explorer browser window displaying the results of a JMeter load test. The address bar shows the file path: C:\JMeter\Save\result\Test4.html. The page title is "Load Test Results - Microsoft Internet Explorer provided by Marsh Canada Limited".

The main content area is titled "summary" and contains a table with the following data:

Tests	Failures	Success Rate	Average Time	Min Time	Max Time
90	0	100.00%	310 ms	0 ms	2703 ms

Below the summary is a section titled "Pages" with a table listing individual page performance:

URL	Tests	Failures	Success Rate	Average Time	Min Time	Max Time
/CanysAanda/processAuthentication.do	10	0	100.00%	216 ms	171 ms	423 ms
/CanysAanda/handlePostAction.do	10	0	100.00%	44 ms	31 ms	47 ms
/claim/cmsstartup.do	10	0	100.00%	344 ms	201 ms	391 ms

The "Details for Page "/claim/cmsstartup.do"" section is expanded, showing a table of thread operations:

Thread	Operation	Time	Success
Thread Group-1	0	312ms	True
Thread Group-2	2	313ms	True
Thread Group-1	3	360ms	True
Thread Group-2	4	359ms	True
Thread Group-1	5	320ms	True
Thread Group-2	6	326ms	True
Thread Group-1	7	371ms	True
Thread Group-2	8	344ms	True
Thread Group-1	9	350ms	True
Thread Group-2	10	331ms	True

At the bottom of the page, there is a table with the following data:

/claim/jsp/common/marsh_header.jsp	10	0	100.00%	25 ms	0 ms	63 ms
/claim/searchClaimForm.do	10	0	100.00%	343 ms	219 ms	484 ms
/claim/CMSApplet/CMSHttpMessenger.jar	10	0	100.00%	34 ms	0 ms	76 ms
/claim/addClaim.do	20	0	100.00%	866 ms	15 ms	2703 ms
/claim/logoutClaims.do	10	0	100.00%	127 ms	94 ms	187 ms



Managers

Use Jmeter to monitor a website's performance



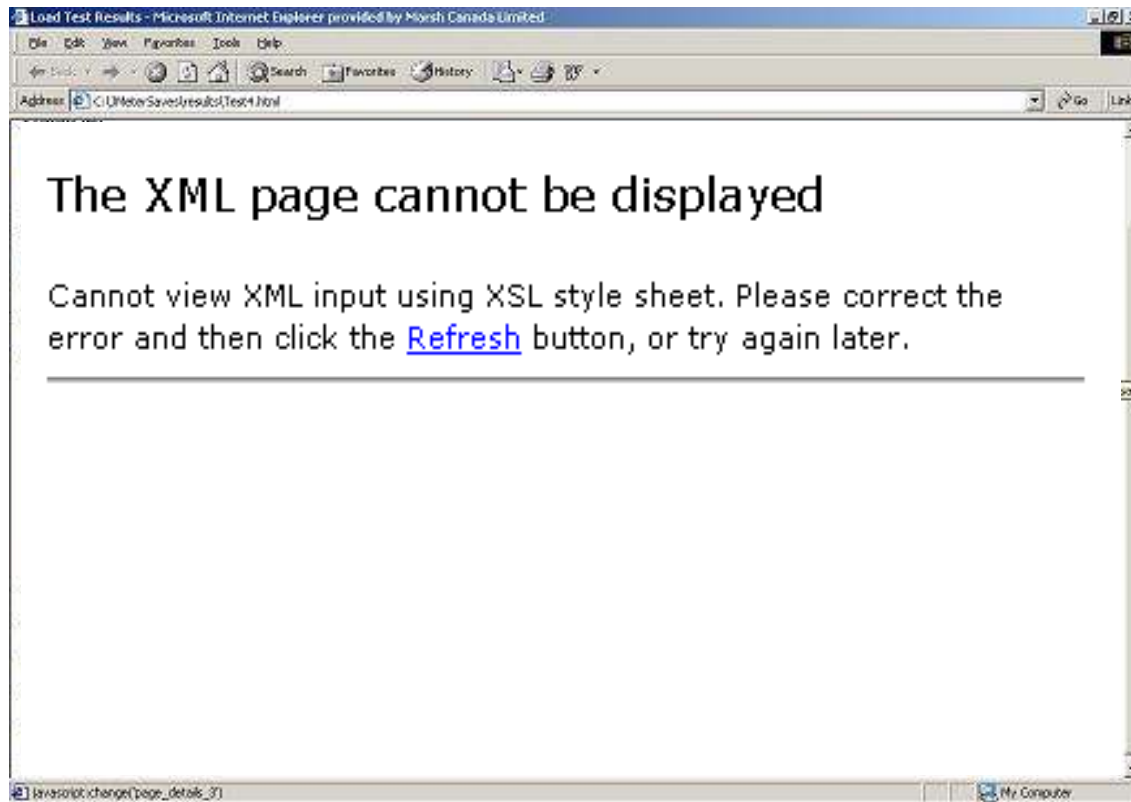
Version 2.8



All regression tests passed



Version 2.9



Managers

Use Jmeter to monitor a website's performance



Version 2.8



No regression bugs.



All regression tests passed



Version 2.9



Causes XML parsing error

The new version behaves as **designed**, but **differently** from a user expects.



Version 2.8



All regression tests passed



Version 2.9



ConfSuggester



... Suspicious configuration options

output_format

Resolve the problem: set **output_format = XML**



Version 2.8



All regression tests passed



Version 2.9



Load Test Results - Microsoft Internet Explorer provided by Marsh Canada Limited

Address: C:\JMeter\Save\result\Test4.html

Tests	Failures	Success Rate	Average Time	Min Time	Max Time
90	0	100.00%	318 ms	0 ms	2703 ms

Pages

URL	Tests	Failures	Success Rate	Average Time	Min Time	Max Time
/CanysisAandA/processAuthentication.do	10	0	100.00%	216 ms	171 ms	422 ms
/CanysisAandA/handlePostAction.do	10	0	100.00%	44 ms	31 ms	47 ms
/claim/cmsstartup.do	10	0	100.00%	344 ms	201 ms	391 ms

Details for Page "/claim/cmsstartup.do"

Thread	Iteration	Time	Success
Thread_Group-1	0	312ms	True
Thread_Group-2	0	313ms	True
Thread_Group-1	1	360ms	True
Thread_Group-2	1	359ms	True
Thread_Group-1	2	370ms	True
Thread_Group-2	2	328ms	True
Thread_Group-1	3	371ms	True
Thread_Group-2	3	344ms	True
Thread_Group-1	4	359ms	True
Thread_Group-2	4	331ms	True
Thread_Group-1	5	331ms	True
Thread_Group-2	5	331ms	True
Thread_Group-1	6	344ms	True
Thread_Group-2	6	359ms	True
Thread_Group-1	7	359ms	True
Thread_Group-2	7	359ms	True
Thread_Group-1	8	359ms	True
Thread_Group-2	8	359ms	True
Thread_Group-1	9	359ms	True
Thread_Group-2	9	359ms	True
Thread_Group-1	10	331ms	True
Thread_Group-2	10	331ms	True

/claim/jsp/common/marsh_header.jsp	10	0	100.00%	25 ms	0 ms	65 ms
/claim/searchClaimForm.do	10	0	100.00%	343 ms	219 ms	404 ms
/claim/CMSApplet/CMSHttpNessenger.jar	10	0	100.00%	34 ms	0 ms	76 ms
/claim/addClaim.do	20	0	100.00%	966 ms	15 ms	2703 ms
/claim/logoutClaims.do	10	0	100.00%	127 ms	94 ms	187 ms

Outline

- Example
- ➔ • A Study of Configuration Evolution
- The ConfSuggester Technique
- Evaluation
- Related Work
- Contributions

Do configuration changes arise in software evolution?

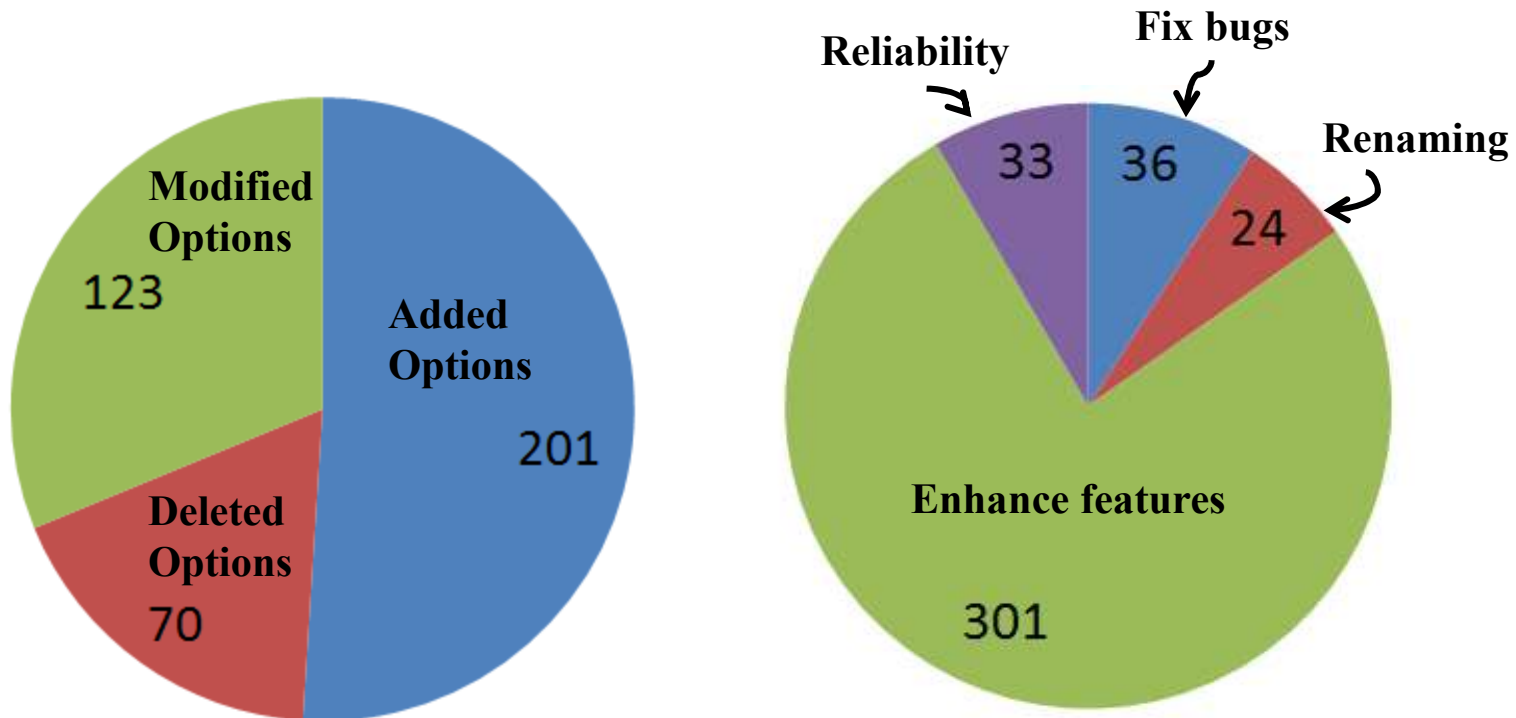
- 8 open-source programs



- **40 versions** released in the past **6 years**
- Searched for “**configuration changes**”-related messages in **7022** commits and **28** change logs
 - Count the number of changes made to configuration options

Results

- Configuration changes arise in **every** version of **all** software systems (**394** configuration changes in total)



- Configuration change can lead to **unexpected** behaviors (details later)

Outline

- Example
- A Study of Configuration Evolution
- • The ConfSuggester Technique
- Evaluation
- Related Work
- Contributions

Key insights of ConfSuggester

- **Control flow** propagates most configuration options' effects
- The execution traces on the old version can serve as the “**intended behavior**”
 - The **control flow difference** and **their impacts** provides diagnosis clues

```
/* a configuration option in JMeter */
```

```
String output_format = readFromCommandLine();
```

```
...
```

```
if ((output_format == "XML"))
```

```
    saveAsXML();
```

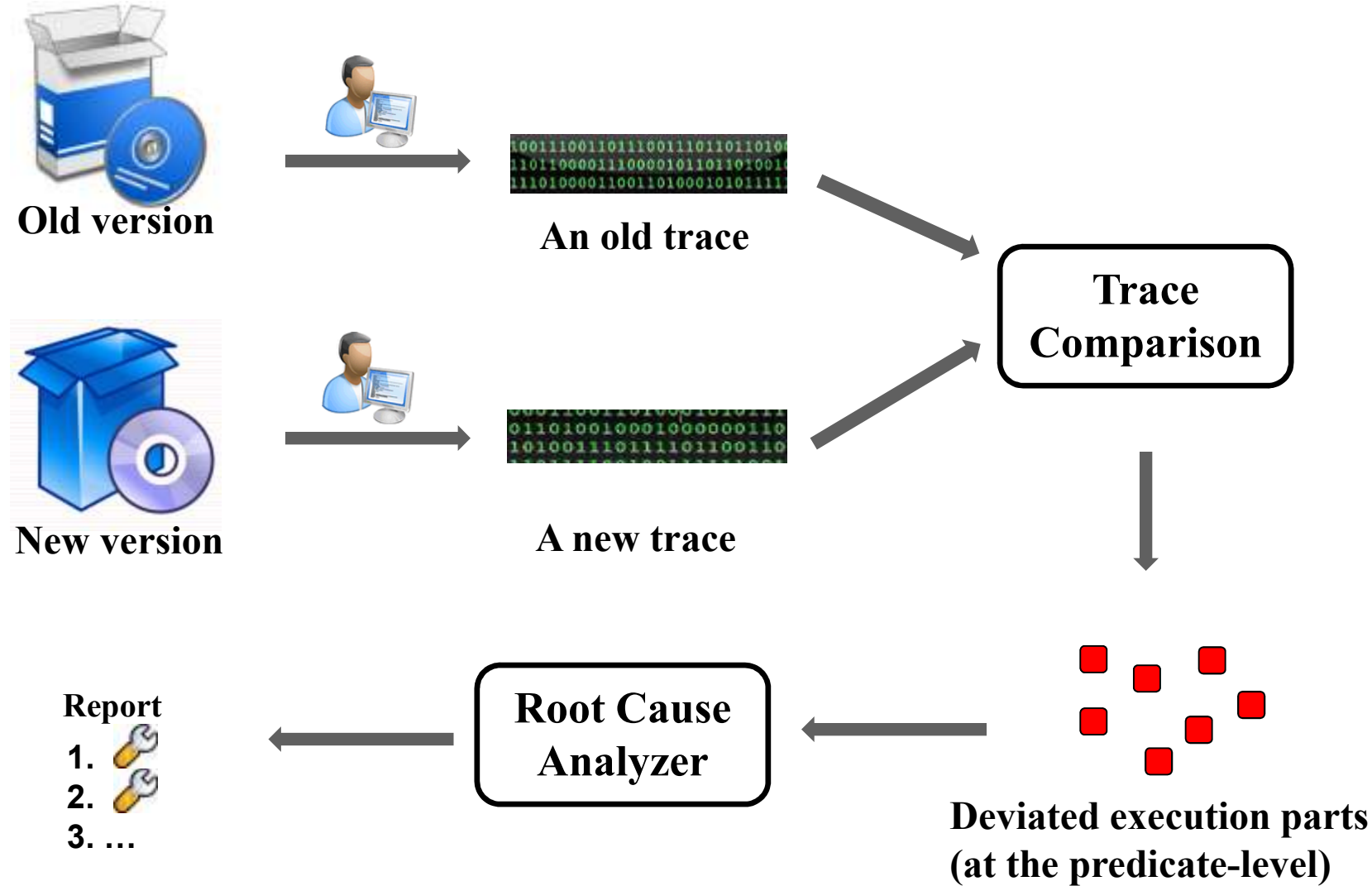
```
    } else {
```

```
        saveAsCSV();
```

```
    }
```

The evaluation result of this predicate affects the next 1000+ instructions

Workflow of ConfSuggester



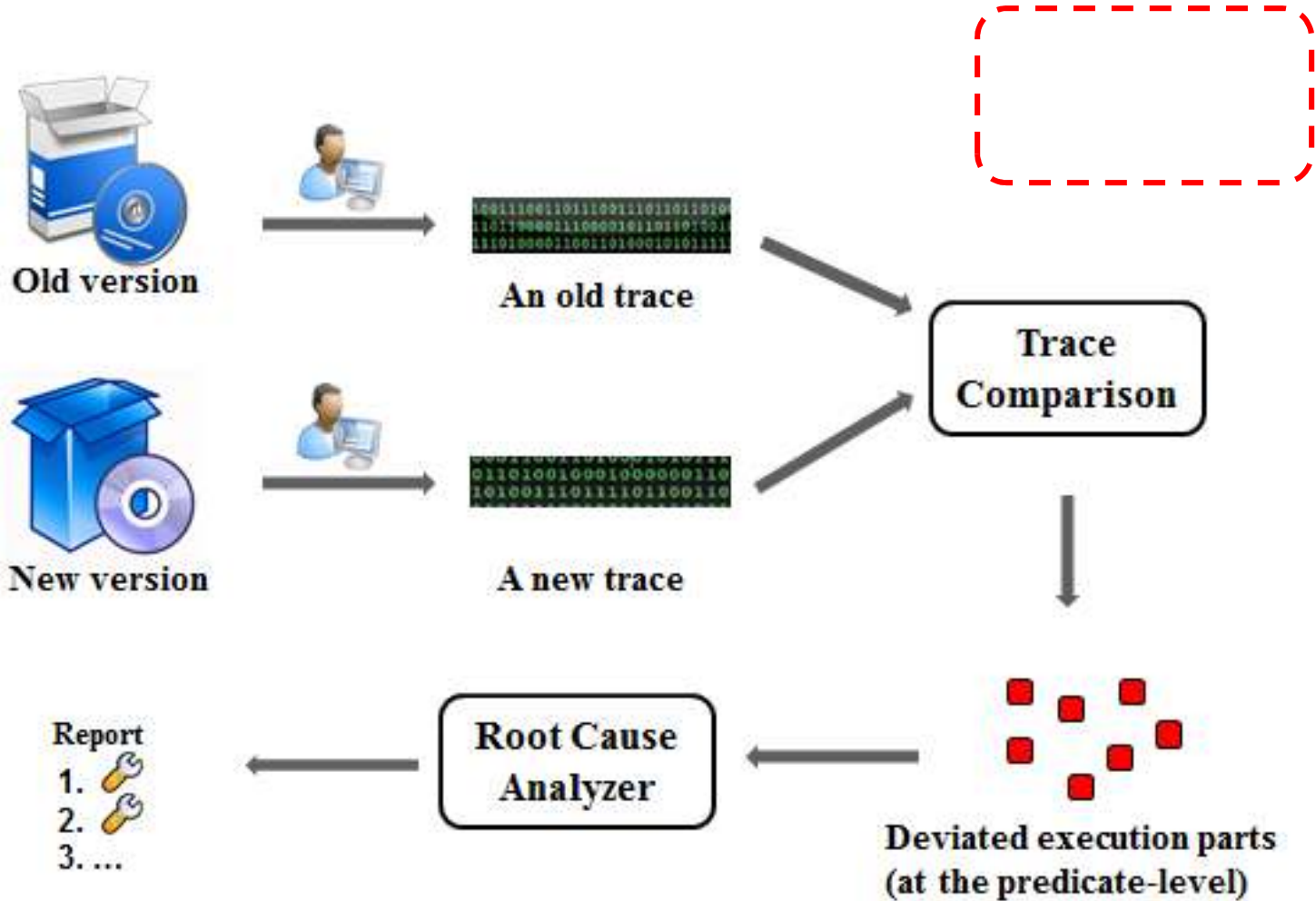
Workflow of ConfSuggester

User demonstration:
show the error

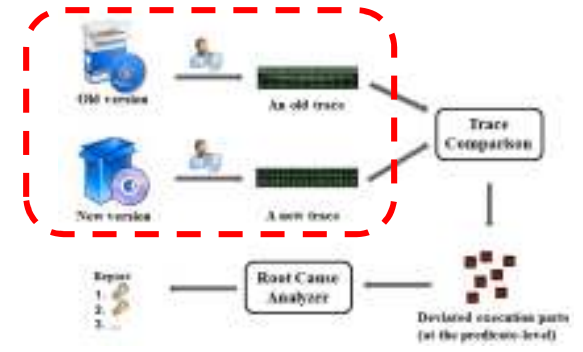
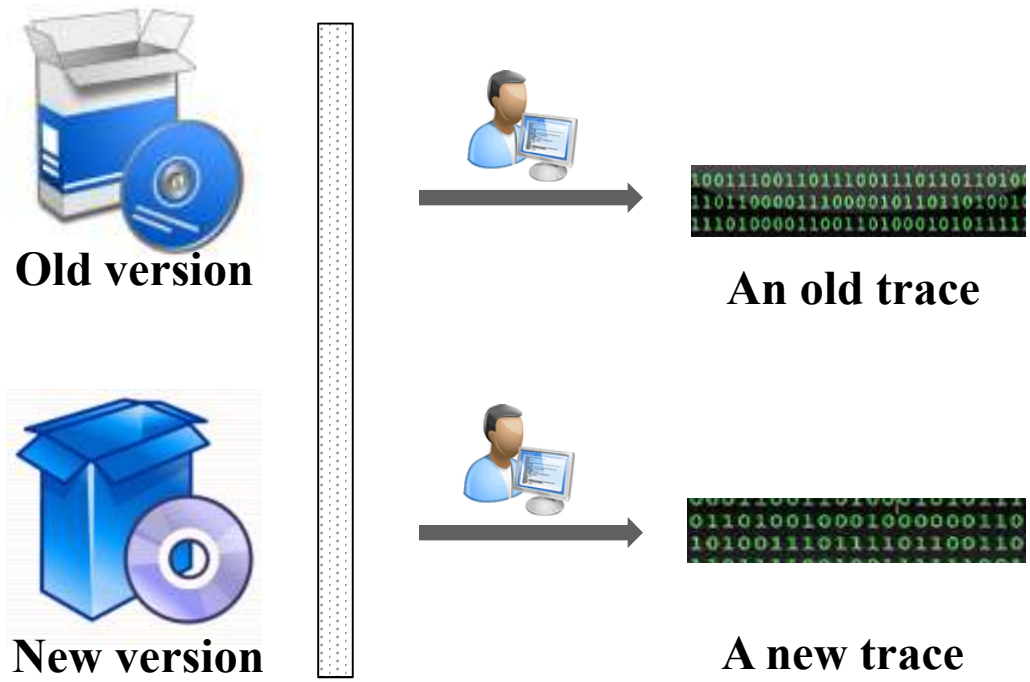
Dynamic analysis:
understand the
behavior

Static analysis:
compute the solution

Workflow of ConfSuggester



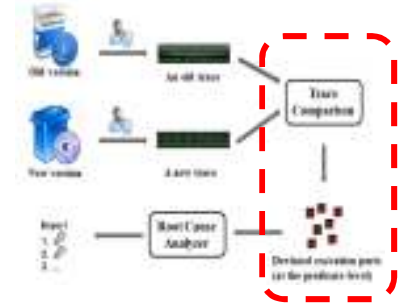
User demonstration



Code instrumentation, monitoring:

- 1. predicate execution frequency and result**
- 2. execution of each other instruction**

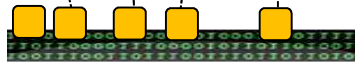
Execution trace comparison



An old trace



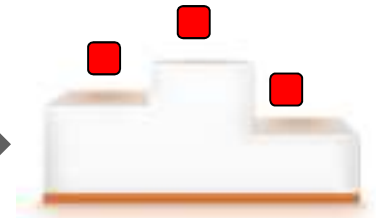
A new trace



■ : a predicate



■ : a deviated predicate

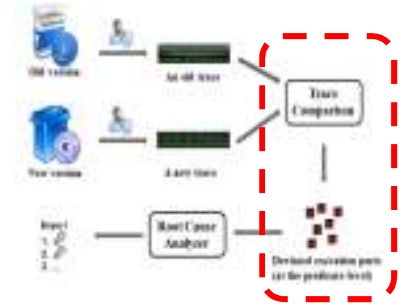


**Matching
predicates**

**Identifying
deviated
predicates**

**Ranking
deviated
predicates**

Matching predicate across traces



- JDiff algorithm [[Apiwattanapong'07](#)]
 - Tolerate small changes between versions

```
...  
if (output_format == "XML") {  
    saveAsXML();  
} else {  
    saveAsCSV();  
}  
...
```

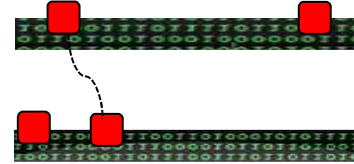
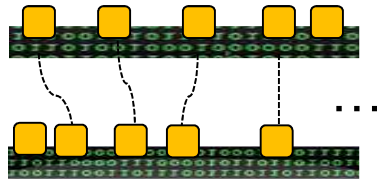
Old version

```
...  
if (isValidFormat(output_format)) {  
    //check validity  
}  
  
if (output_format == "XML") {  
    checkXMLParser();  
    saveAsXML();  
} else {  
    saveAsCSV();  
}  
...
```

New version

Identifying deviated predicates

Goal:



An old trace

A new trace

■ : a predicate

■ : a deviated predicate

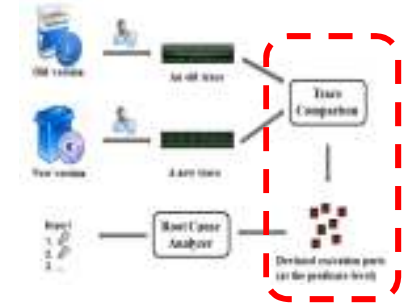
a predicate p 's **behavior** in an execution trace t :

$$\phi(p, t) = \frac{2}{\frac{1}{\text{exec frequency}} + \frac{1}{\text{true ratio}}}$$

a predicate p 's **behavior difference** across executions:

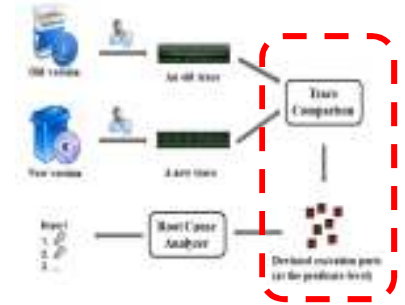
$$\text{deviation}(p, t_{\text{old}}, t_{\text{new}}) = | \phi(p, t_{\text{old}}) - \phi(p, t_{\text{new}}) |$$

p is a **deviated predicate**, if $\text{deviation}(p, t_{\text{old}}, t_{\text{new}}) > \delta$



Ranking deviated predicates

Rank predicates by their impacts



A predicate p's deviation impact

$$= \text{deviation}(p, t_{\text{old}}, t_{\text{new}}) \times (\text{controlled_instructions}(p, t_{\text{old}}) + \text{controlled_instructions}(p, t_{\text{new}}))$$

Defined in the previous slide

predicate p:

```
...  
if (output_format == "XML") {  
    saveAsXML();  
} else {  
    saveAsCSV();  
}  
...
```

Old trace

of instructions executed

saveAsXML()

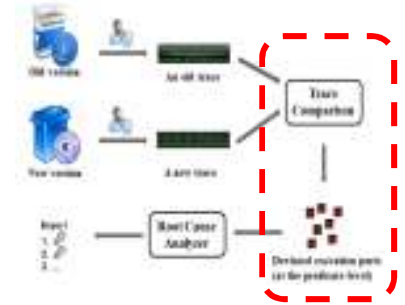
Old trace

if(...)

saveAsCSV()

Ranking deviated predicates

Rank predicates by their impacts



A predicate p's deviation impact

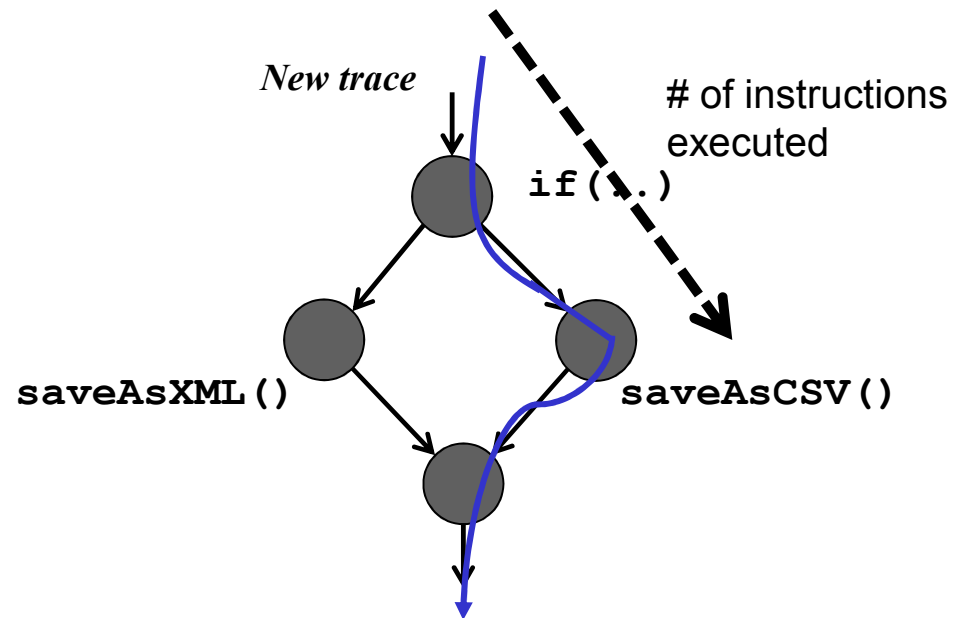
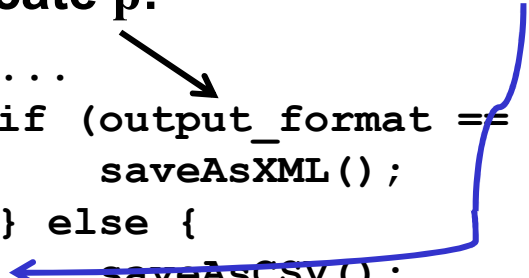
$$= \text{deviation}(p, t_{\text{old}}, t_{\text{new}})$$

$$\times (\text{controlled_instructions}(p, t_{\text{old}}) + \text{controlled_instructions}(p, t_{\text{new}}))$$

predicate p:

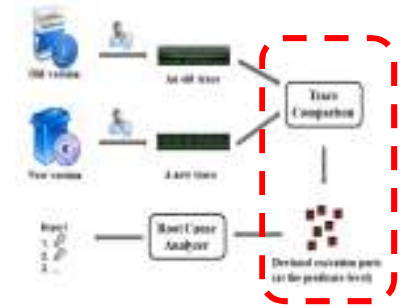
```
...
if (output_format == "XML") {
    saveAsXML();
} else {
    ← saveAsCSV();
}
...
```

New trace



Ranking deviated predicates

Rank predicates by their impacts



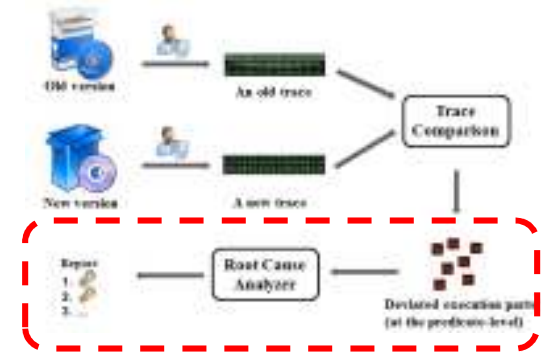
A predicate p's deviation impact

$$= \text{deviation}(p, t_{\text{old}}, t_{\text{new}})$$

$$\times (\text{controlled_instructions}(p, t_{\text{old}}) + \text{controlled_instructions}(p, t_{\text{new}}))$$

Approximate the impact of a predicate's behavior change to the subsequent program execution.

Root Cause Analyzer



Find **configuration options** affecting the deviated predicate

- Using static thin slicing [Sridharan '07]

```
//a configuration option in JMeter
String output_format = ...;
```

```
...
if (output_format == "XML") {
    saveAsXML() ;
```

```
} else {
    saveAsCSV() ;
```

```
}
```

Find the affecting predicate

Compute a backward thin slice from here


The behavior of this predicate deviates

Report

1. 🛠️
2. 🛠️
3. ...

output_format

Outline

- Example
- A Study of Configuration Evolution
- The ConfSuggester Technique
-  • Evaluation
- Related Work
- Contributions

8 configuration errors from 6 subjects

Subject	LOC	#Options	ΔLOC	#Config errors
Randooop	18587	57	1893	1
Weka	275035	14	1458	1
Synoptic	19153	37	1658	2
JChord	26617	79	3085	2
JMeter	91797	55	3264	1
Javalanche	25144	35	9261	1

Non-trivial code changes

Reproduced from change logs and user reports.

ConfSuggester's accuracy

- Measure accuracy by the rank of the actual root cause in ConfSuggester's output

1. 
2. 
3. ...

ConfSuggester's accuracy

- Measure accuracy by the rank of the actual root cause in ConfSuggester's output

1. 
2. 
3. ...

Technique	Average Root Cause Rank
Baseline	23.3

- **Baseline:**
 - Users select options in an arbitrary order
 - Half of the total number of available options

ConfSuggester's accuracy

- Measure accuracy by the rank of the actual root cause in ConfSuggester's output

1. 
2. 
3. ...

Technique	Average Root Cause Rank
Baseline	23.3
ConfAnalyzer [Rabkin'11]	22

- **ConfAnalyzer:**
 - Use program slicing for error diagnosis

ConfSuggester's accuracy

- Measure accuracy by the rank of the actual root cause in ConfSuggester's output

1. 
2. 
3. ...

Technique	Average Root Cause Rank
Baseline	23.3
ConfAnalyzer [Rabkin'11]	22
ConfDiagnoser [Zhang'13]	15.3

- **ConfDiagnoser:**
 - Use trace comparison (on the same version) for error diagnosis

ConfSuggester's accuracy

- Measure accuracy by the rank of the actual root cause in ConfSuggester's output

1. 
2. 
3. ...

Technique	Average Root Cause Rank
Baseline	23.3
ConfAnalyzer [Rabkin'11]	22
ConfDiagnoser [Zhang'13]	15.3
ConfSuggester (this paper)	1.9

- **ConfSuggester:**
 - **6** errors: root cause ranks **1st**
 - **1** error: root cause ranks **3rd**
 - **1** error: root cause ranks **6th**


ConfSuggester's efficiency

- **User demonstration**
 - 6 minutes per error, on average

- **Error diagnosis**
 - 4 minutes per error, on average



Outline


- Example
- A Study of Configuration Evolution
- The ConfSuggester Technique
- Evaluation
-  • Related Work
- Contributions

Related work on configuration error diagnosis

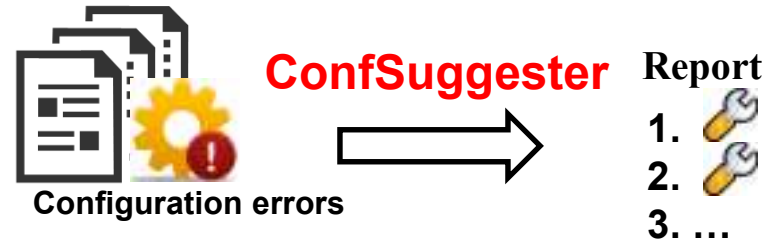
- Tainting-based techniques
 - Dynamic tainting [Attariyan'08], static tainting [Rabkin'11]
Focuses exclusively on crashing errors
- Search-based techniques
 - Delta debugging [Zeller'02], Chronus [Whitaker'04]
Requires a correct state for comparison, or OS-level support
- Domain-specific techniques
 - PeerPressure [Wang'04], RangeFixer [Xiong'12]
Targets a specific kind of configuration errors, and does not support a general language like Java

A common limitation: do not support configuration error diagnosis in software evolution.

Outline

- Example
- A Study of Configuration Evolution
- The ConfSuggester Technique
- Evaluation
- Related Work
-  • Contributions

Contributions



- A technique to diagnose configuration errors for **evolving** software
*Compare **relevant predicate** behaviors between executions from **two versions***

- ✓ **Accessible**: no assumption about user background
- ✓ **Easy-to-use**: fully automated
- ✓ **Portable**: no changes to OS or runtime environment
- ✓ **Accurate**: few false positives

- The ConfSuggester tool implementation

<http://config-errors.googlecode.com>

