

Why do Modern Web Applications Fail and What Can We Do About it?



Karthik Pattabiraman

Electrical and Computer Engineering

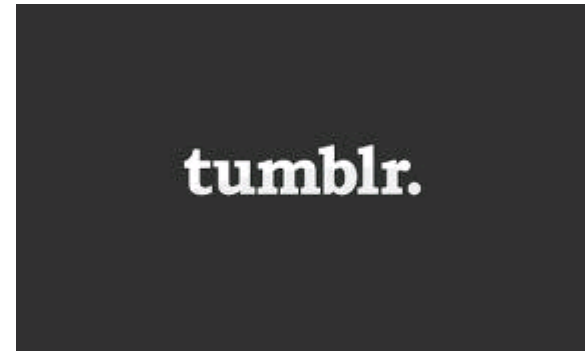
University of British Columbia (UBC)

<http://blogs.ubc.ca/karthik/>

My Research

- **Building error resilient and secure software systems**
- **Three areas**
 - Software resilience [DSN'17][DSN'18A][DSN'18B][SC'18][DSN'16][SC'16][DSN'15]
 - Web applications' reliability [ICSE'18][ICSE'16][ICSE'15][ASE15][ICSE'14A][ICSE'14B]
 - IoT Devices security [FSE'17][ACSAC'16][EDCC'16][HASE'14]
- **This talk: Web Applications' Reliability**

Modern Web Applications: Examples



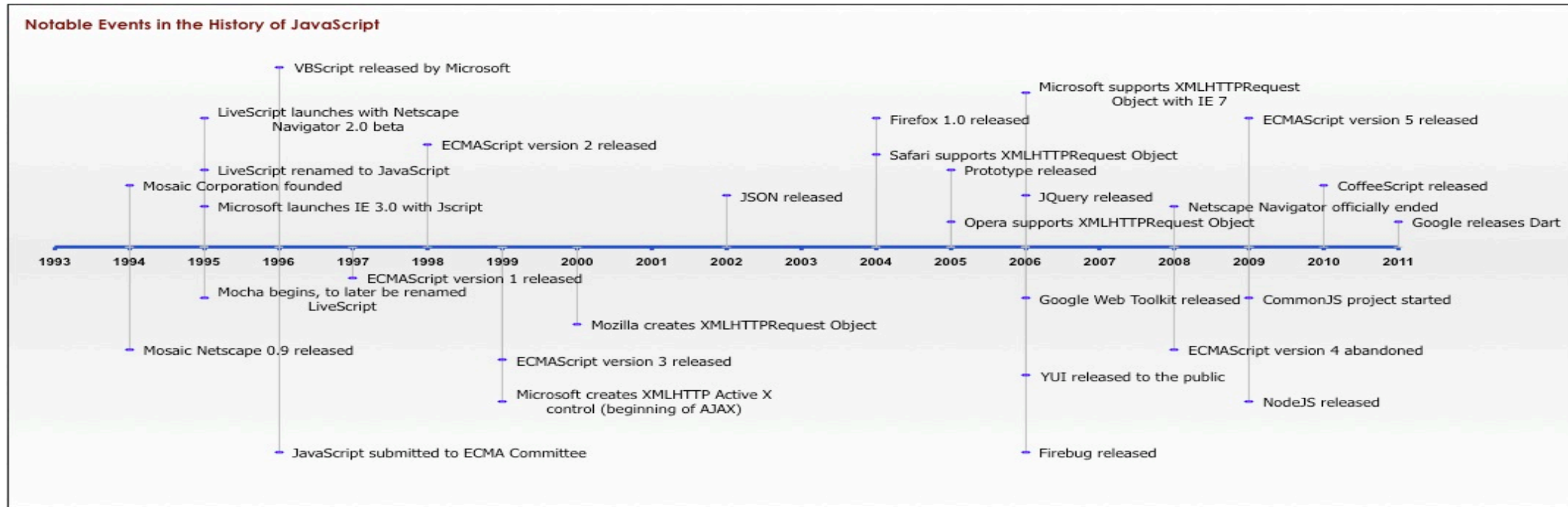
Modern Web Applications: JavaScript

- JavaScript: Implementation of ECMAScript standard
 - Client-Side JavaScript: used to develop web apps
- Executes in client's browser – send AJAX messages
- Responsible for web application's core functionality
- Not easy to write code in – has many “evil” features



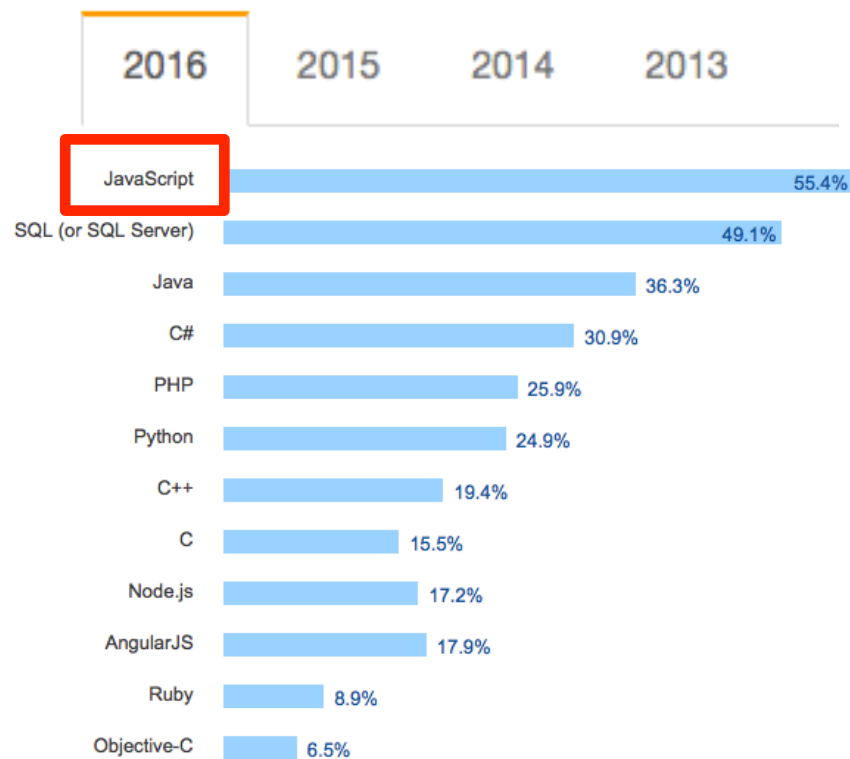
JavaScript: History

Brief History of JavaScript (Source: TomBarker.com)

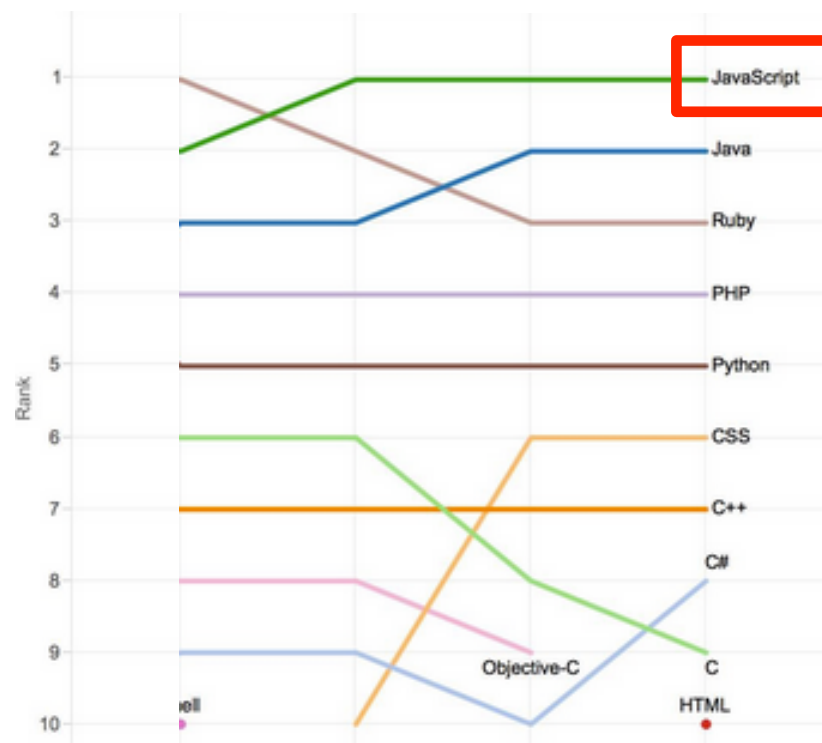


JavaScript (JS) had to “look like Java” only less so, be Java’s dumb kid brother or boy-hostage sidekick. Plus, I had to be done **in ten days** or something worse than JS would have happened – Brendan Eich (Inventor of JavaScript)

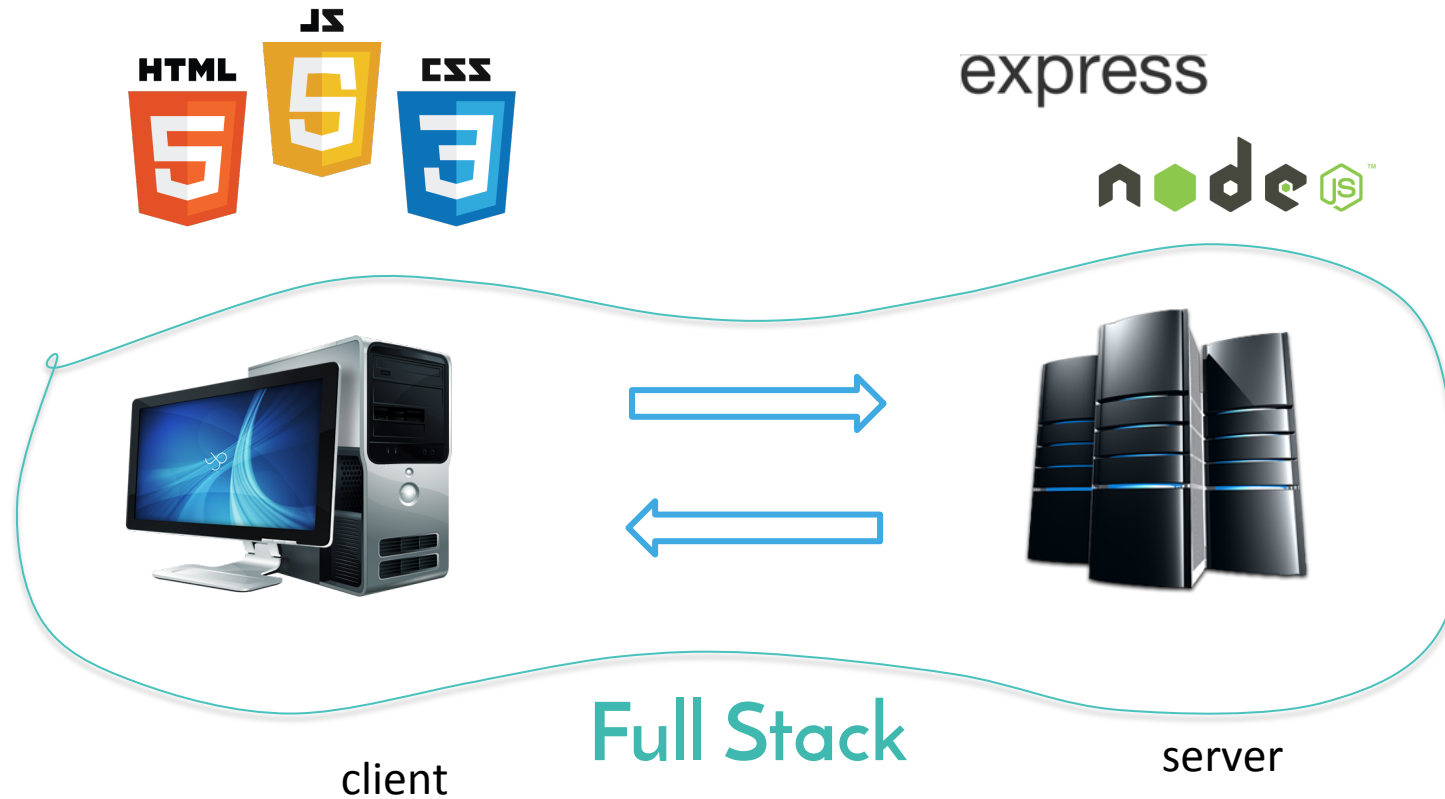
JavaScript: Most popular language



JavaScript: Top languages on GitHub



JavaScript and the Web



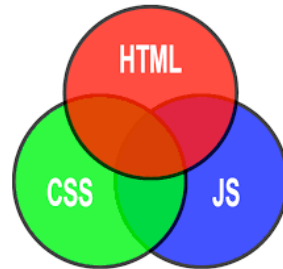
Analyzing JS Code: Challenges



JS has loose semantics



Lack of standard programming style & JS frameworks



Frequent cross-language interactions

Studies of JavaScript Web Applications

Performance and parallelism:

JSMeter [Ratanaworabhan-2010],
[Richards-2009], [Fortuna-2011]

Reliability

?

Security and Privacy:

[Yue-2009], Gatekeeper[Guarnieri-2009],
[Jang-2010]



Goal: Study and improve the reliability of JavaScript web applications

Does Reliability Matter ?

- Snapshot of iFeng.com: Leading media website in China

an error occurred when processing this directive

[an error occurred while processing this directive]

李克强宣布广州亚残运会开幕

火炬手攀登点燃主火炬|数开幕式十宗“最”
亚残运开幕解密|广州亚残运会开幕式特写

广州亚运会圆满闭幕 高清大图

[组图]仁川十分钟：Rain连唱三曲|暖场演出
童谣《月光光》 拉开序幕|大郅出任中国旗手

女排上演绝地逆转战胜韩国夺冠

周苏红发威女排逆转|韩国输球再斥裁判丑陋
女排逆转令洪钢哽咽|俞觉敏：我为队员骄傲

[高清]冠军球员搭讪礼仪小姐

裁判引导韩朝摔跤手赛场握手|摔跤精彩瞬间
男篮绝杀伊朗进决赛|朝鲜女足失冠背向升旗

- “铁血女将”黄蕴瑶暂列亚运英雄榜之首
- 中华台北选手罹癌参赛 携奖牌返家无遗憾
- 日本男女足亚运齐称霸 统治亚洲足坛获证
- 霍启刚温文尔雅态度和蔼 与郭晶晶差别大
- 快讯：广州亚运会发生第二起兴奋剂事件
- 阿联首绝杀韩国队 将与日本争男足金牌
- 韩朝射箭选手只关注比赛 不知两国冲突



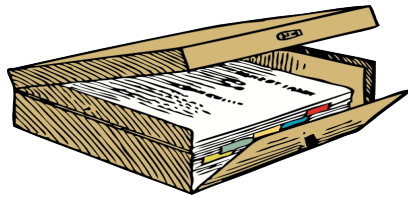
王治郅闭幕式上挥舞国旗入场

2 of 3

Talk Outline

- Motivation and Goals
- Empirical Study of reliability
- Reliability Improvements
- Conclusions and Future Directions

Bug Report Study: Methodology



Collected 502 **fixed** bug reports from 19 web applications over 10 years (2004-2014)



Qualitatively analyzed and classified bug reports manually and reading the commits



Aggregated data for further analysis

Bug Report Study: 19 Objects (15 applns, 4 libraries)



Bug Report Study: Research Questions

- What mistakes **cause** JavaScript faults?



- What **impact** do JavaScript faults have?



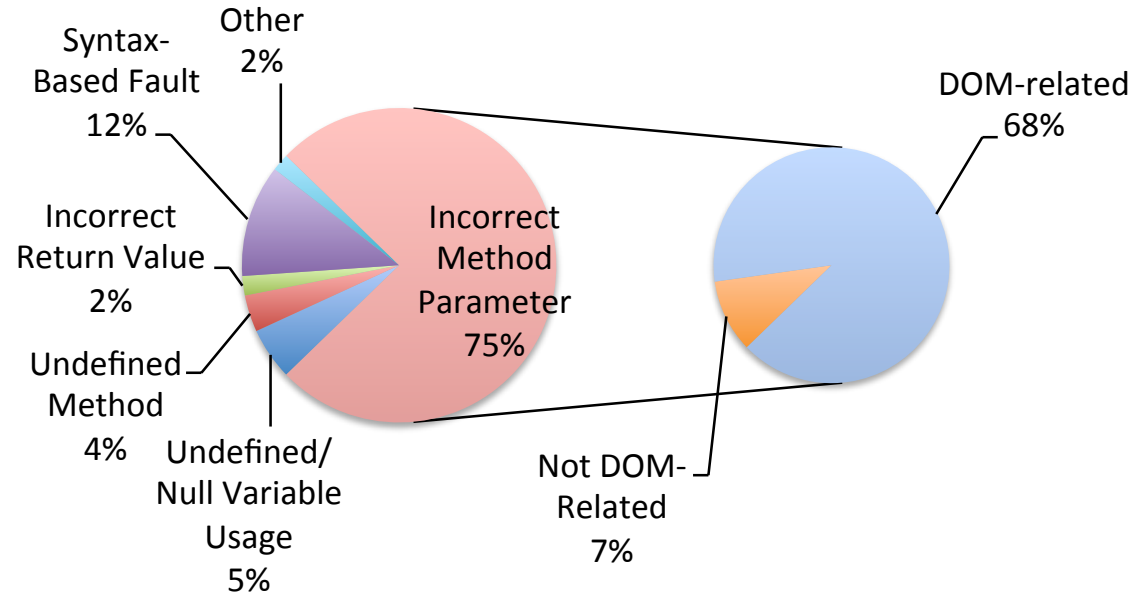
- How long does it take to fix these errors?



- How many of these faults are browser-specific ?

- How many faults can be caught by strong typing ?

Bug Report Study: Categories



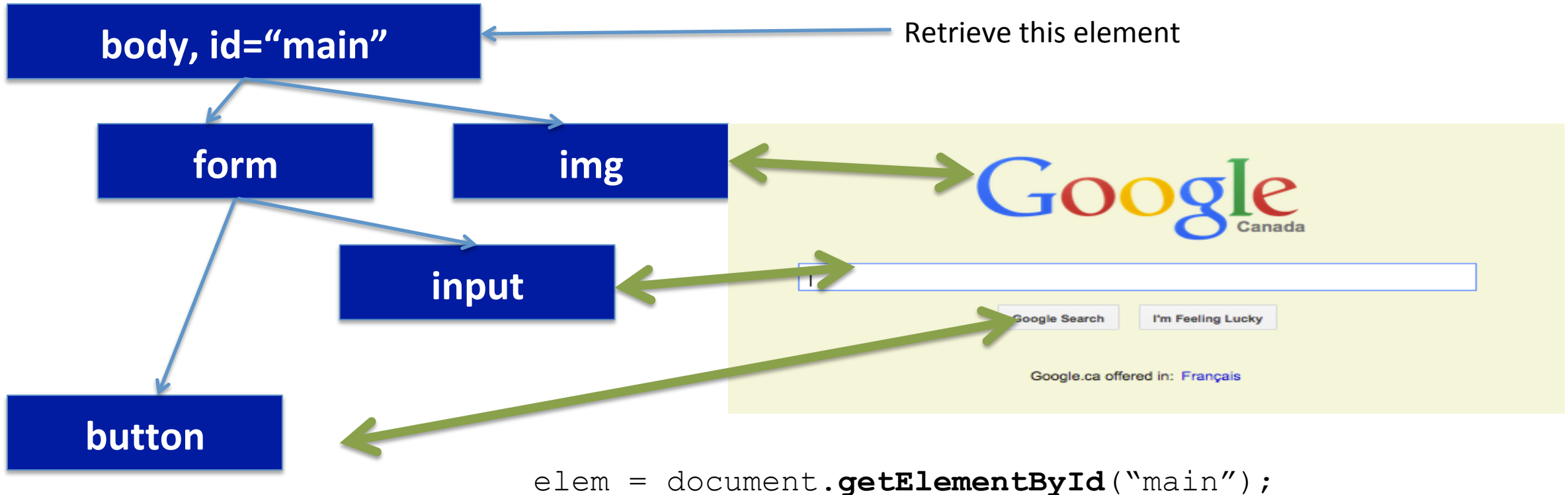
Incorrect Method Parameter Fault: Unexpected or invalid value passed to JS method or assigned to JS property

DOM-Related Fault: The method is a DOM API method
- Account for more than two-thirds of JavaScript Faults

Bug Report Study: DOM-Related Faults

DOM (Document Object Model)

Webpage



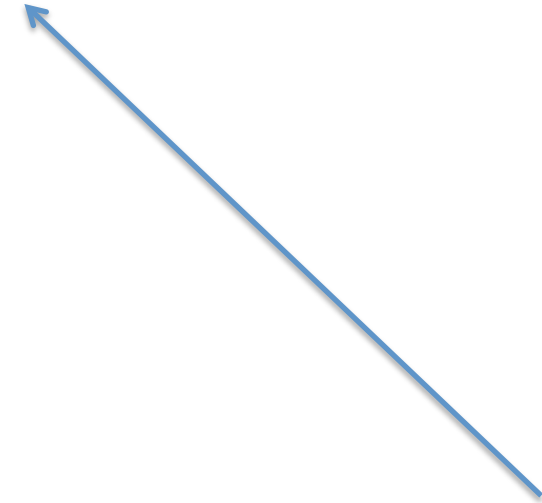
Programming Error that propagates to a DOM method parameter

DOM-Related Fault: Example

```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("##" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```

DOM-Related Fault: Example

```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("#" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```



Retrieved string
via XHR

DOM-Related Fault: Example

```
var elem, retrievedStr = [Retrieved via XHR];  
var dotsInStr = retrievedStr.split(".").length;  
if (dotsInStr == 0) {  
    var prefix = "id_";  
    elem = $("#" + prefix + retrievedStr);  
}  
else {  
    elem = $(retrievedStr);  
}  
elem[0].focus();
```

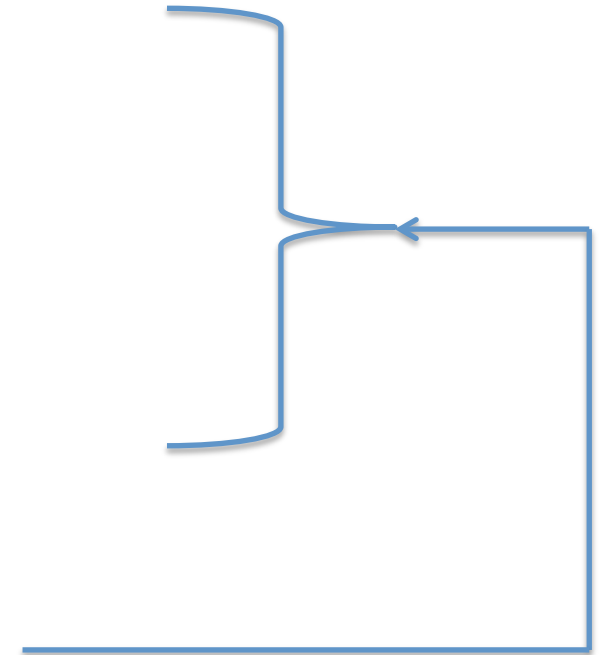


Find the number
of dots in the
string

DOM-Related Fault: Example

```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("#" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```

If there are no dots, prepend “id_” to the string and access it via `$()`. Otherwise, leave it as is, and access it via `$()`.



DOM-Related Fault: Example

```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("#" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```

UNDEFINED EXCEPTION!

Retrieved string of “editor” would go here even though it has no dots, which would erroneously cause `$("#")` to use selector “editor”, which doesn’t match any elements.

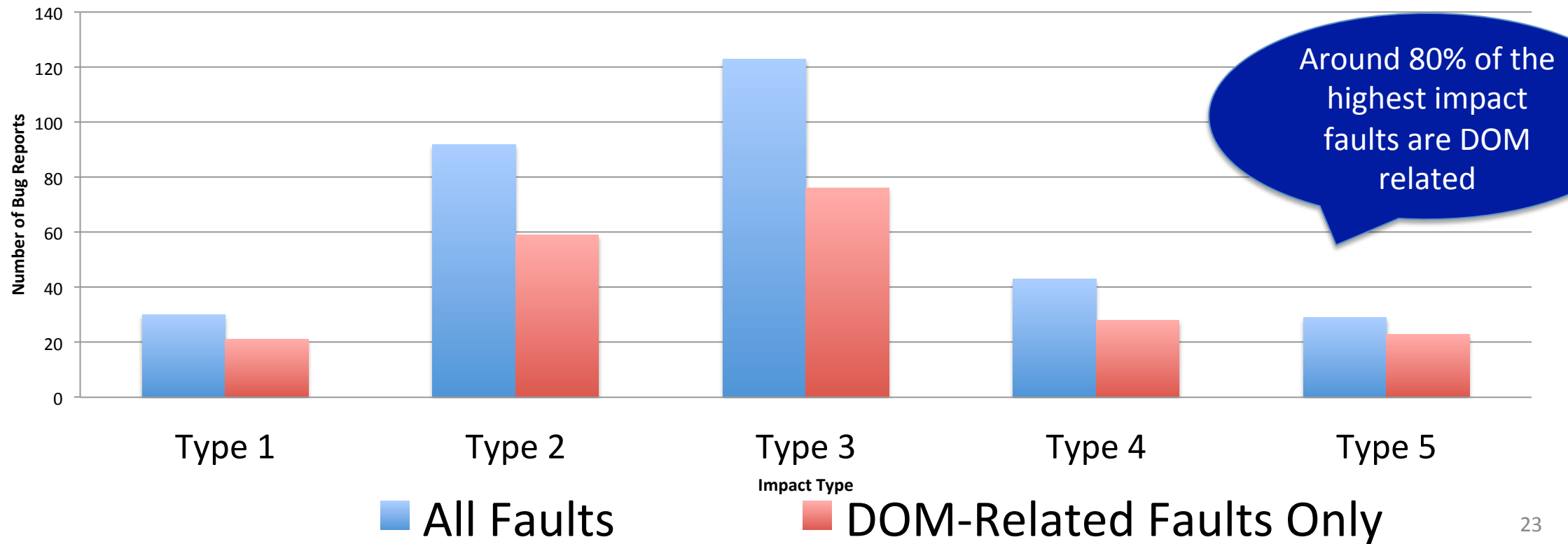
DOM-Related Fault: Example

```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("#" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```

BUG: The assigned value should be `retrievedStr.split(".").length - 1`, as `length()` always returns at least 1.

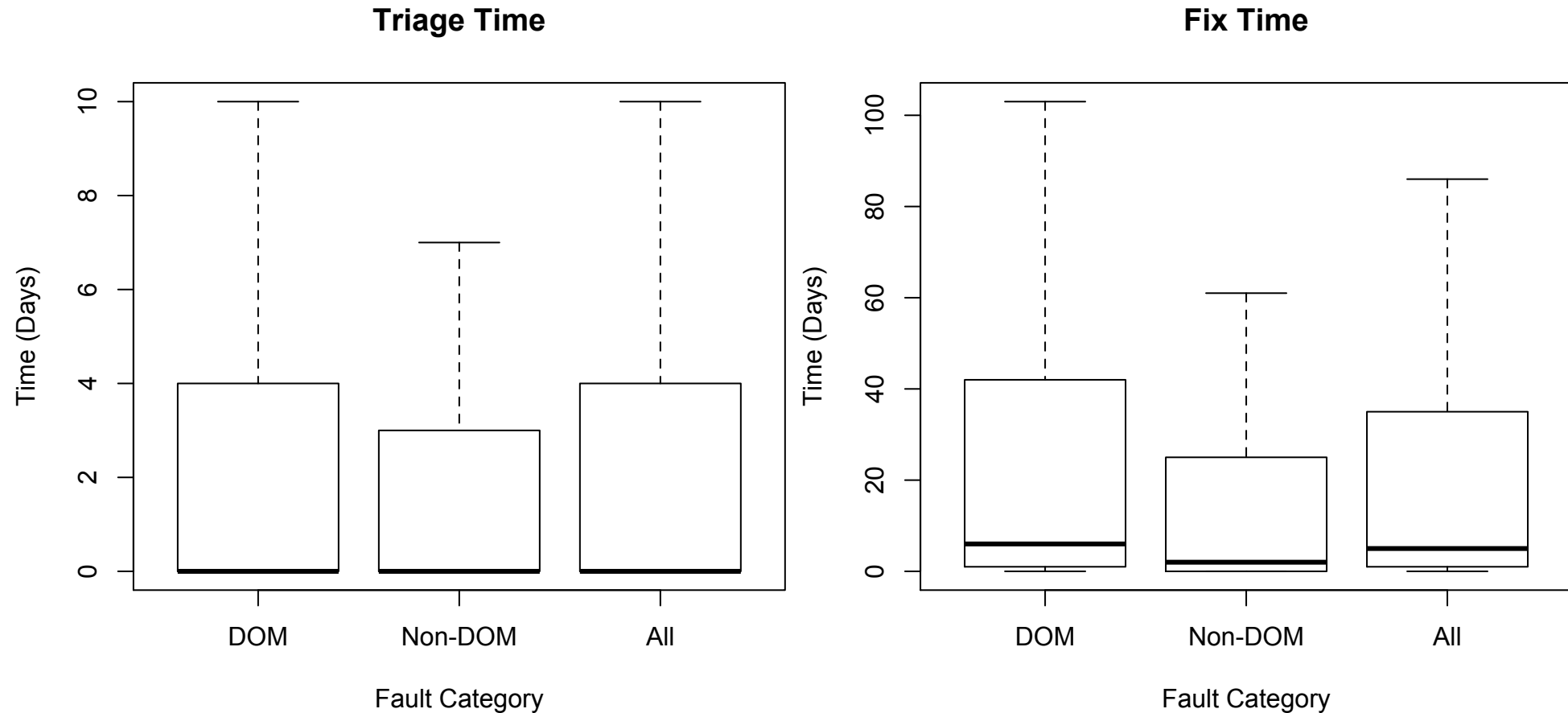
Bug Report Study: Fault Impact

- Impact Types – Based on Bugzilla [ICSE'11]
 - Type 1 (lowest impact – e.g., cosmetic changes)
 - Type 5 (highest impact – e.g., data loss bugs)



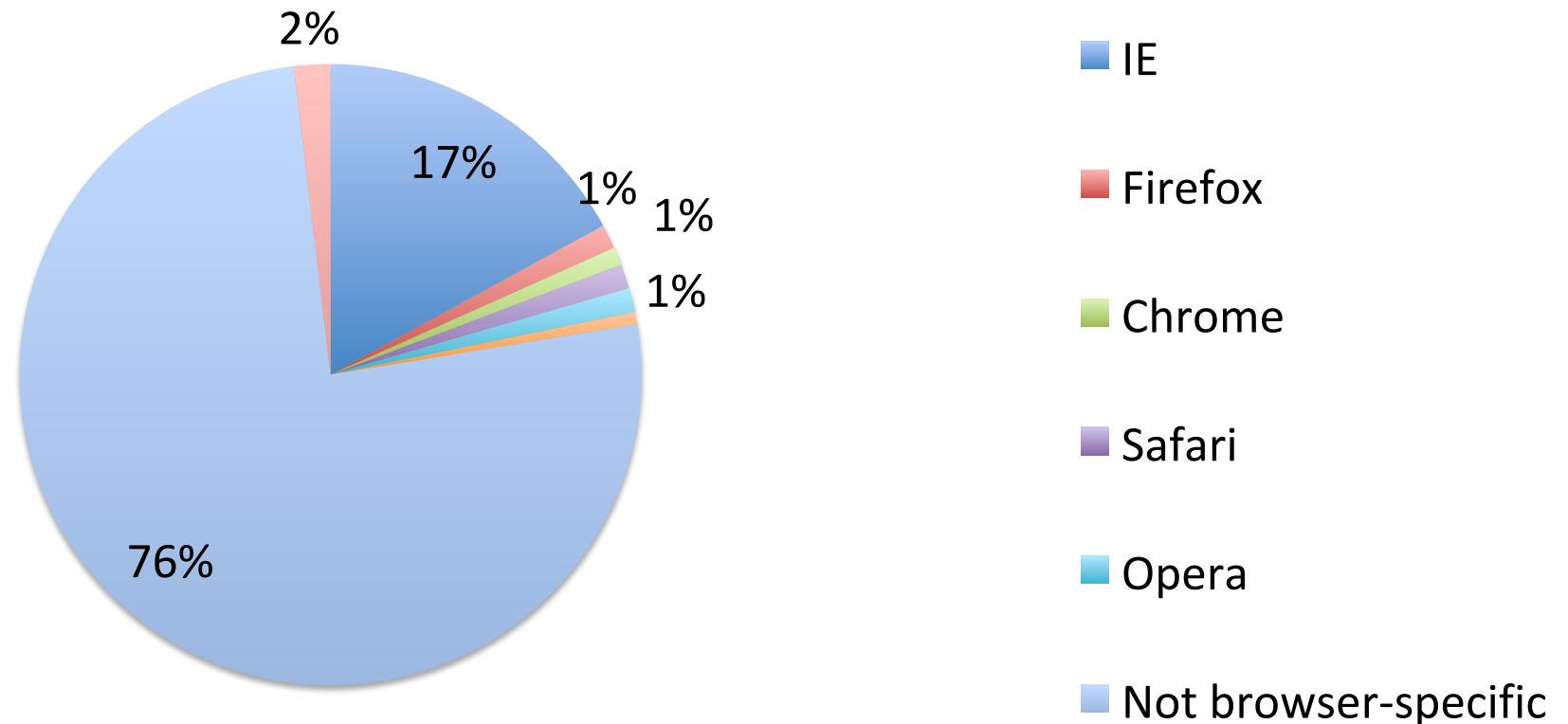
Bug Report Study: Fix Times

- **Triage Time:** Time it took to assign or comment on the bug
- **Fix Time:** Time it took to fix the bug since it was triaged



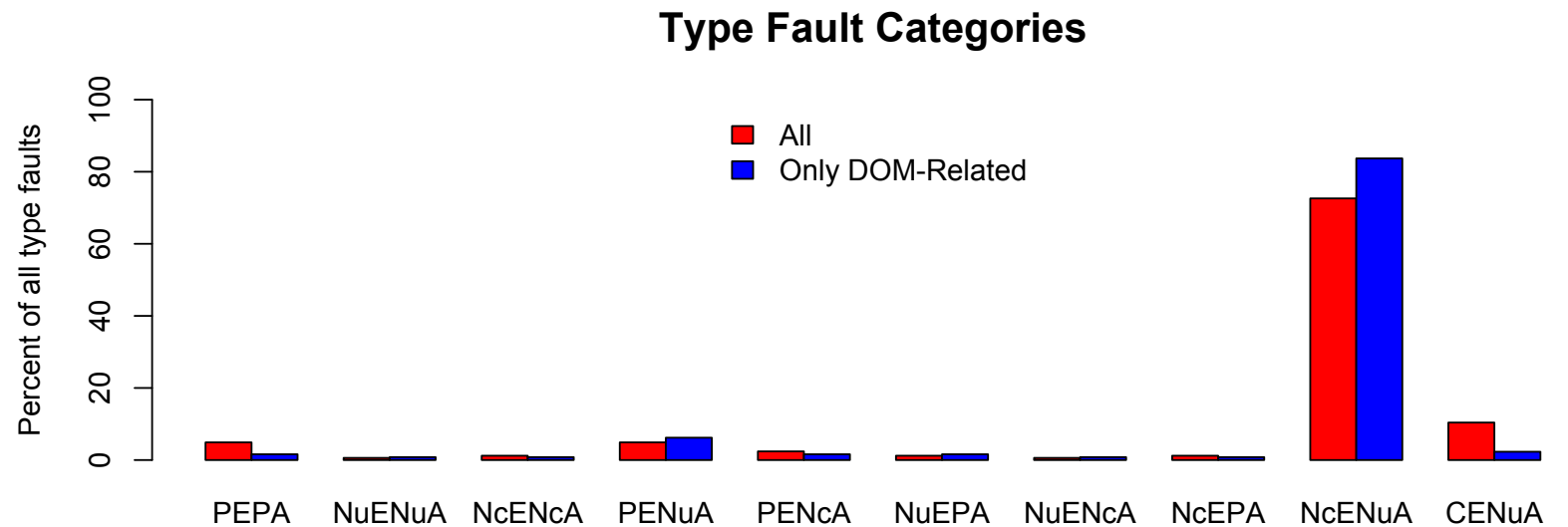
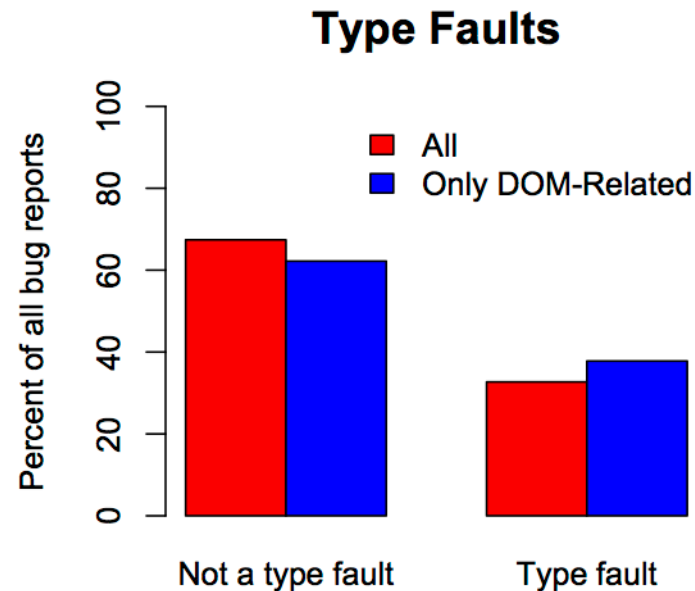
Bug Report Study: Browser Specificity

Most JavaScript faults are not browser-specific

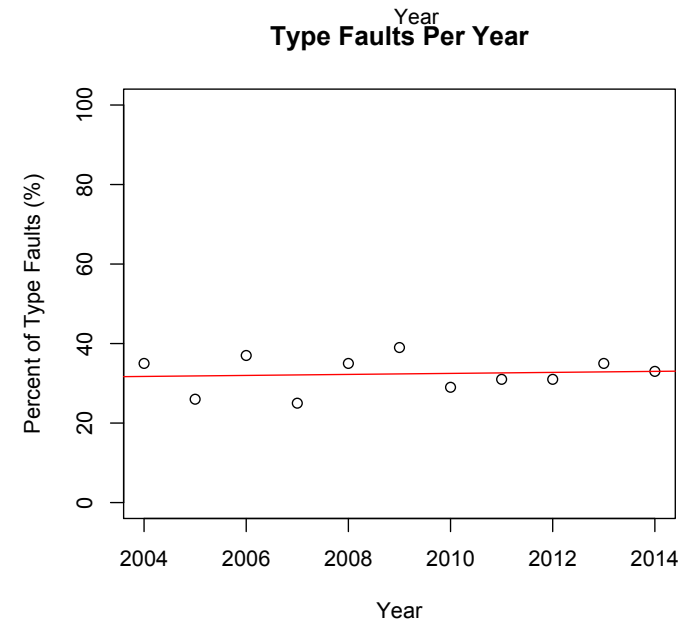
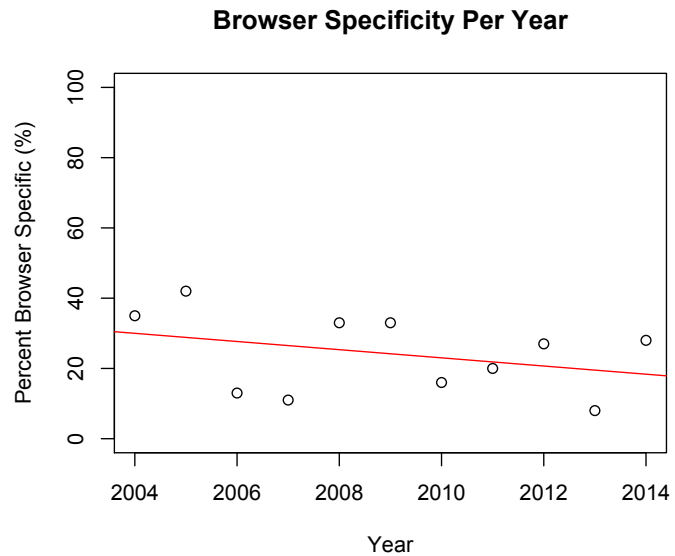
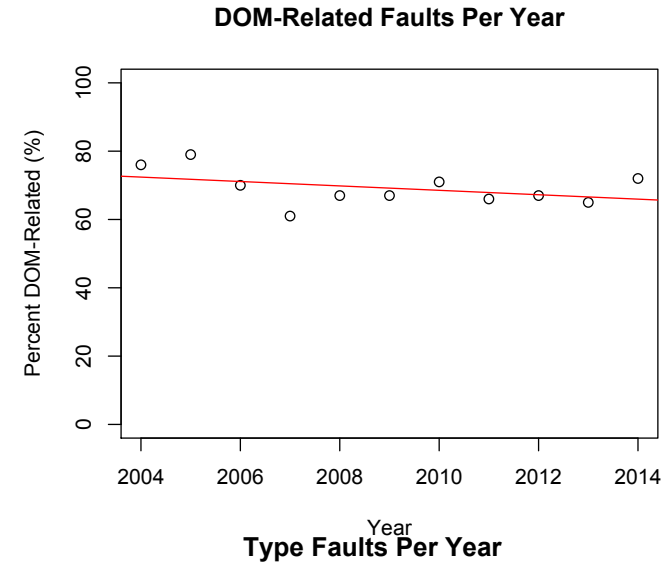
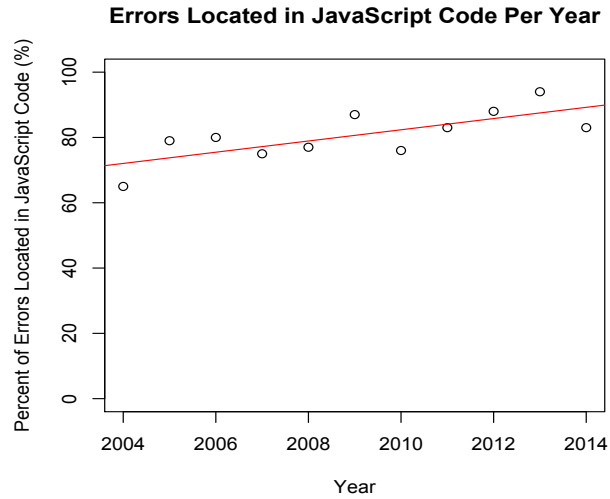


Bug Report Study: Type Faults

- Most DOM-related faults are NOT type faults
 - Among the type faults, a single category dominates



Bug Report Study: Temporal Trends



Bug Report Study: Summary

- **Bug report study of 19 applications: JS faults**
 - Over 500 bug reports analyzed; only fixed bugs considered
- **DOM-related faults dominate JavaScript faults**
 - Responsible for nearly two-thirds of all JavaScript faults
 - Responsible for 80% of highest impact faults
 - Take 50% longer time to fix for developers
 - Majority are not specific to web browser platform
 - Most DOM-related faults are NOT type errors
 - DOM-related faults have been steady over 10 years

Talk Outline

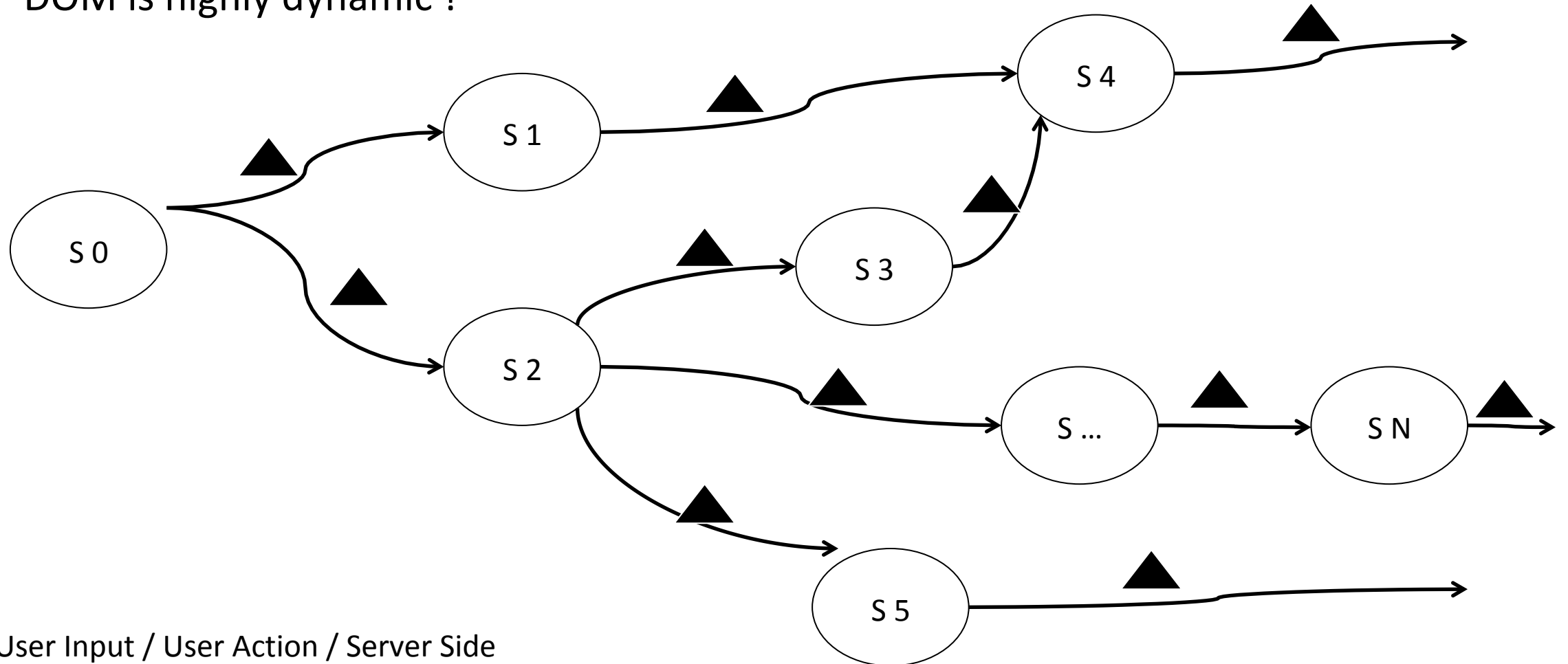
- Motivation and Goals
- Empirical Study of reliability
- **Reliability Improvements**
- Conclusions and Future Directions

Web Applications: Existing Techniques

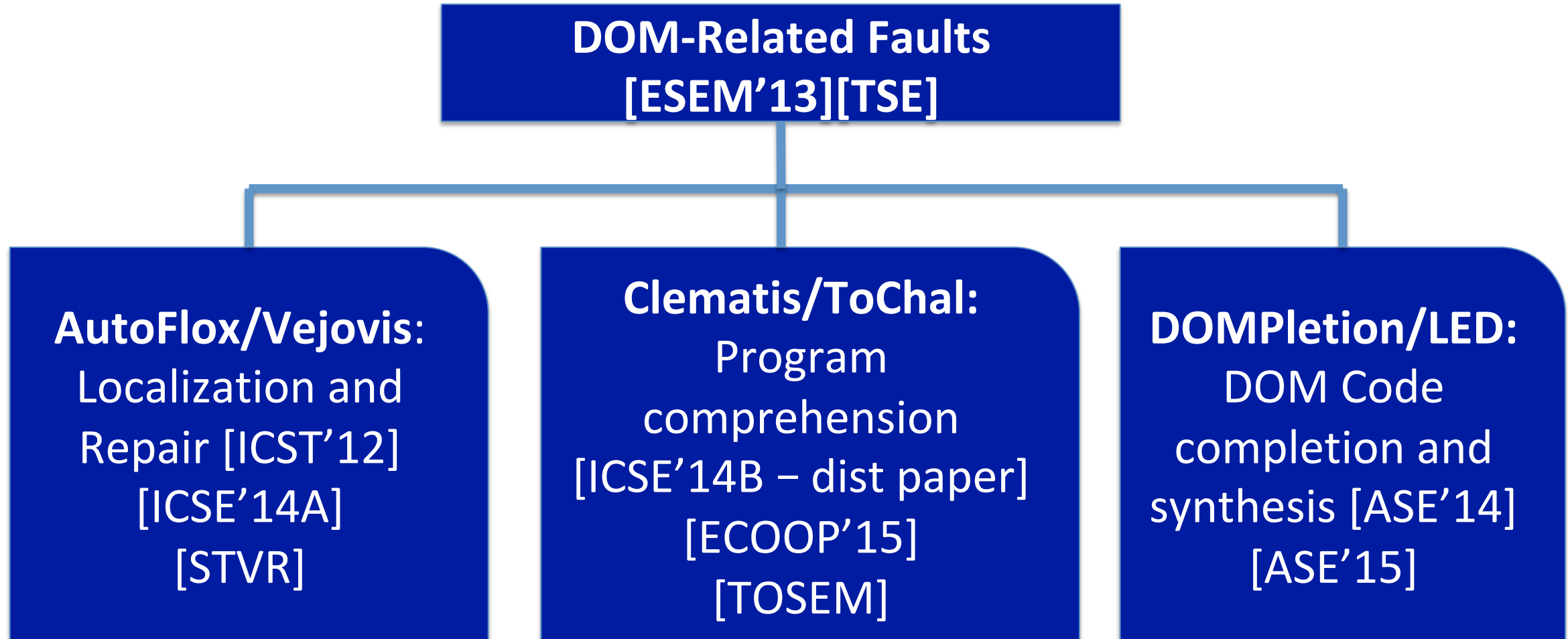
- **Add gradual typing to JavaScript (e.g., TypeScript from MS, DART from Google, Flow from Facebook)**
 - Typically ignore the DOM or provide only limited support
- **Use higher-level programming idioms in JavaScript**
 - MVC Frameworks (e.g., AngularJS)
 - Functional Reactive Programming (e.g., RxJS)
- **Detecting errors in web applications: Ignore DOM**
 - Race conditions [Vechev-OOPSLA'13][Livshits-FSE'15]
 - Type Coercion Errors [Pradel – ICSE'15][Moeller – OOPSLA'14]

Web Applications: Challenge

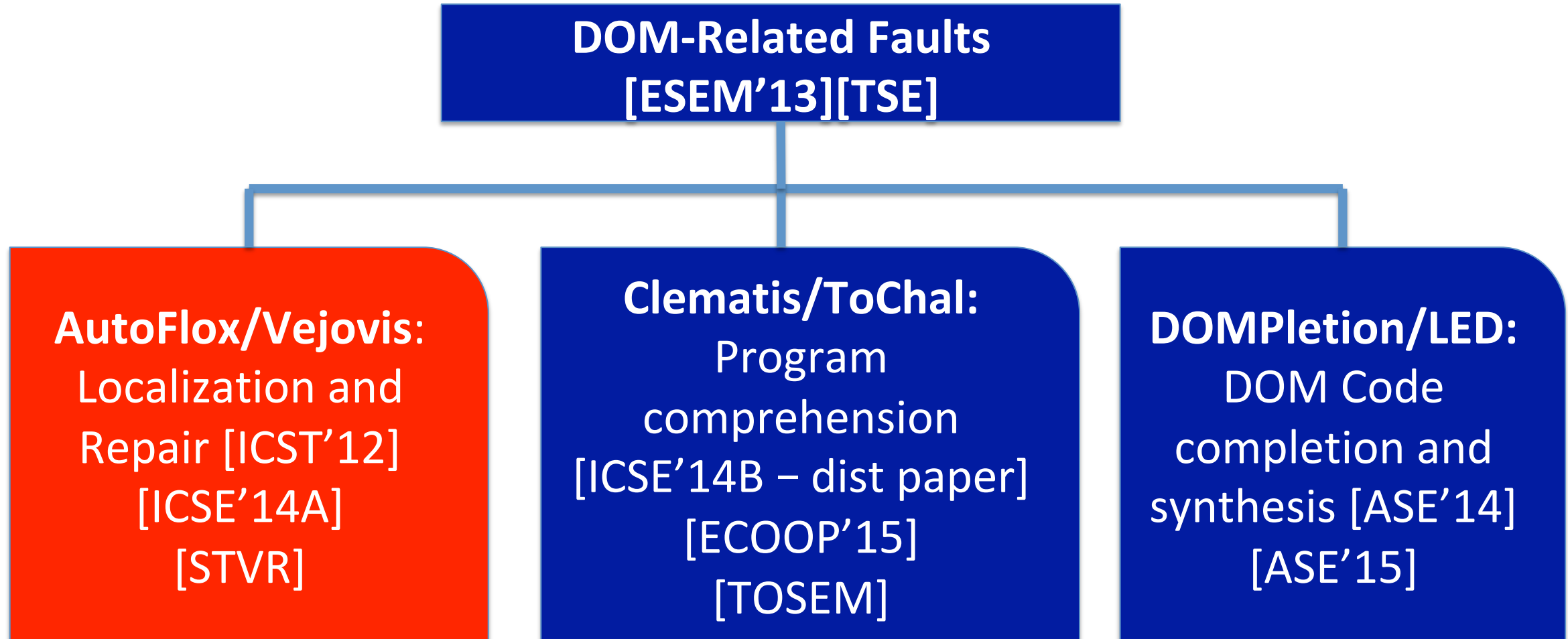
DOM is highly dynamic !



Web Applications: Our Approach



Web Applications: Our Approach



Web Applications: Program Repair (VejoVis)

WRONG

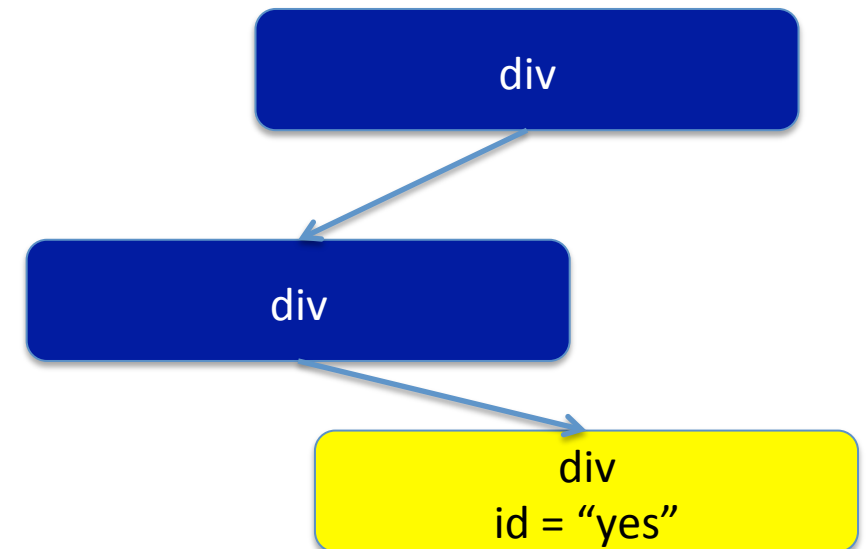
```
getElementById( "no" )
```

RIGHT

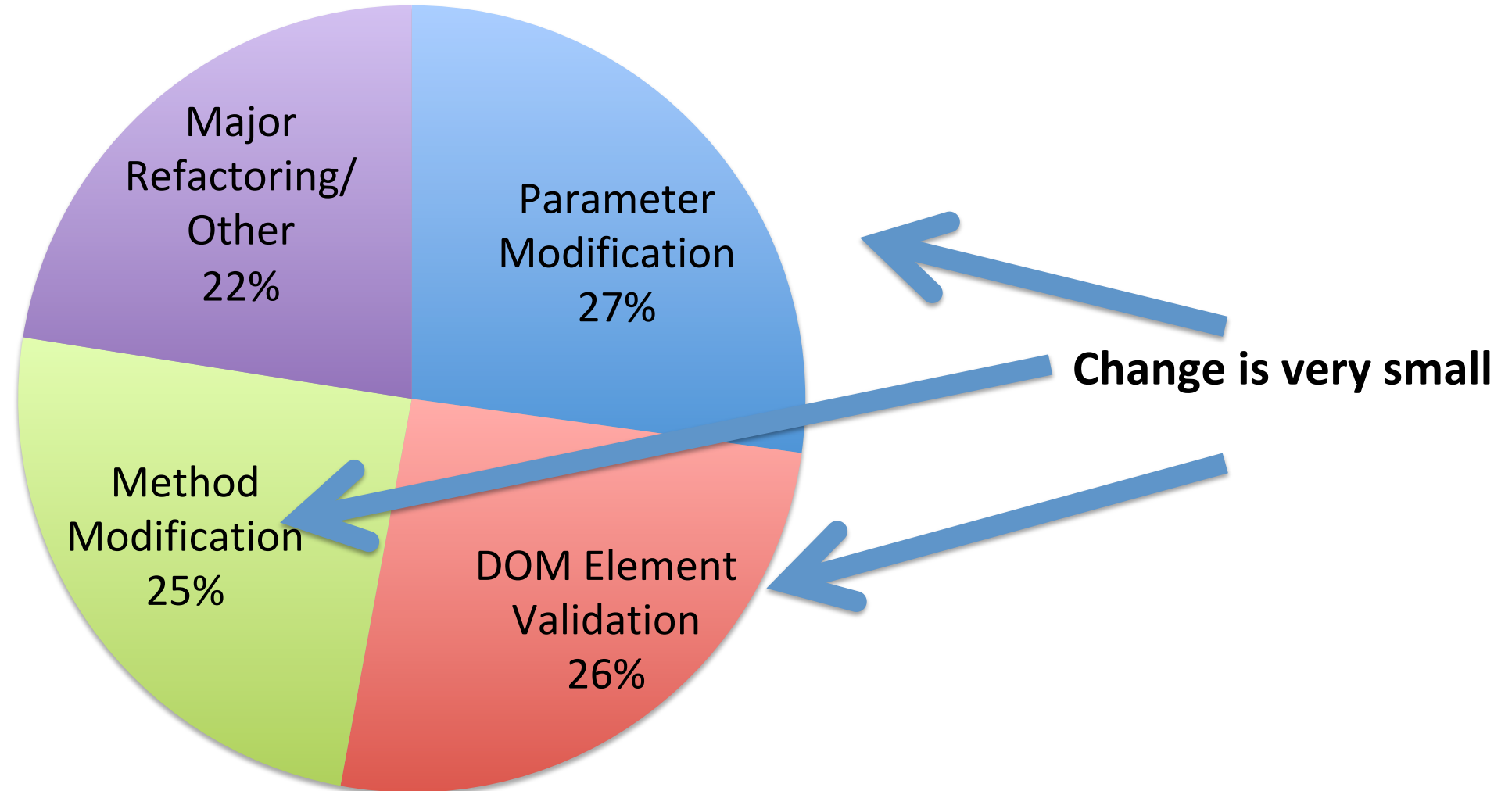
```
getElementById( "yes" )
```

Question: How do we know that we should replace "no" with "yes"

Answer: We use the DOM structure, and the JavaScript code structure



Web Applications: Bug Fix Patterns

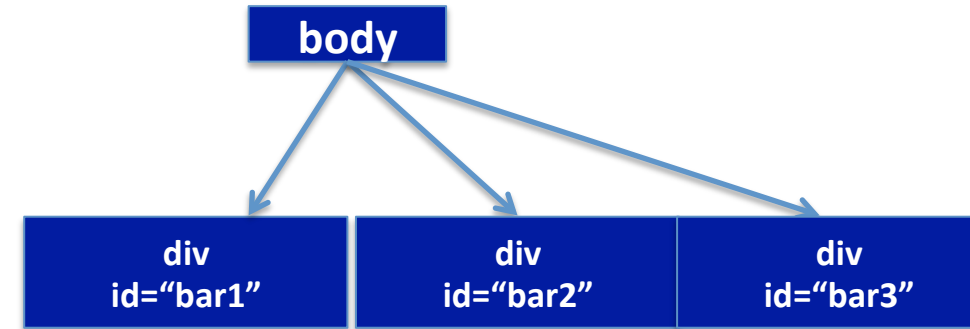


Web Applications: Running Example

```
1 function generateId(index) {  
2     var prefix = "bar";  
3     var id = prefix + index;  
4     return id;  
5 }
```

```
7 function retrieveElement(index) {  
8     var id = generateId(index);  
9     var e = document.getElementById(id);  
10    return e;  
11 }
```

```
13 for (var i = 1; i <= 4; i++) {  
14     var elem = retrieveElement(i);  
15     elem.innerHTML = "Item #" + i;  
16 }
```



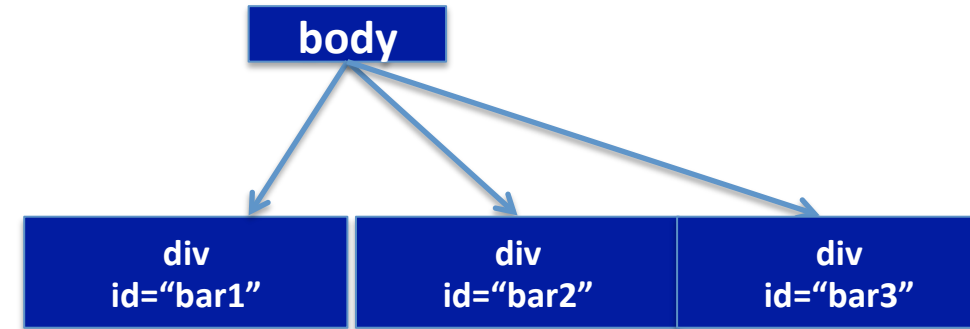
← Add the "bar" prefix to the ID

← Retrieve the element with index i

← Update retrieved element

Web Applications: DOM-Related Faults

```
1 function generateId(index) {
2     var prefix = "bar";
3     var id = prefix + index;
4     return id;
5 }
6
7 function retrieveElement(index) {
8     var id = generateId(index);
9     var e = document.getElementById(id);
10    return e;
11 }
12
13 for (var i = 1; i <= 4; i++) {
14     var elem = retrieveElement(i);
15     elem.innerHTML = "Item #" + i;
16 }
```



This should be "<",
not "<="

Evaluates to
"bar4" in 4th
iteration

NULL EXCEPTION!

AutoFlox: Fault Localization [ICST'12][STVR]

```
1 function generateId(index) {
2     var prefix = "bar";
3     var id = prefix + index;
4     return id;
5 }
6
7 function retrieveElement(index) {
8     var id = generateId(index);
9     var e = document.getElementById(id)
10    return e;
11 }
12
13 for (var i = 1; i <= 4; i++) {
14     var elem = retrieveElement(i);
15     elem.innerHTML = "Item #" + i;
16 }
```

ERROR POINT TO
REPAIR

Our Goal

FAILURE POINT

AutoFlox: Fault Localization [ICST'12][STVR]

```
1 function generateId(index) {  
2     var prefix = "bar";  
3     var id = prefix + index;  
4     return id;  
5 }  
6  
7 function retrieveElement(index) {  
8     var id = generateId(index);  
9     var e = document.getElementById(id);  
10    return e;  
11 }  
12  
13 for (var i = 1; i <= 4; i++) {  
14     var elem = retrieveElement(i);  
15     elem.innerHTML = "Item #" + i;  
16 }
```

ERROR POINT TO REPAIR

Vejovis
[ICSE'14]

DOM METHOD
CALL

AutoFlox
[ICST'12]
[STVR'16]

FAILURE POINT

Vejovis: Fault Repair [ICSE'14]

```
1 function generateId(index) {  
2     var prefix = "bar";  
3     // ...  
4     // ... and  
5 }  
6  
7 f  
8  
9  
10  
11 }  
12  
13 for (var i = 1; i <= 4; i++) {  
14     var elem = retrieveElement(i);  
15     elem.innerHTML = "Item #" + i;  
16 }
```

*“OFF-BY-ONE, SO REMOVE LAST
ITERATION OF FOR LOOP”*

Evaluates to “bar4”

“4”

“4” comes from iterator i

Web Applications: VejoVis Actions

- Determine **action** for programmer to match valid DOM element selectors (based on fix patterns)
 - Use String Solver (HAMPI) [Kiezun09] to find the action
 - Rank fixes based on their edit distances from original

MESSAGE TYPES
REPLACE STRING
REPLACE STRING AT ITERATION
OFF BY ONE AT BEGINNING
OFF BY ONE AT END
MODIFY UPPER BOUND
EXCLUDE ITERATION
ENSURE THAT

Web Applications: VejoVis Recall

Subject	Bug Report #1	Bug Report #2
Drupal	✓	✓
Ember.js	✓	✓
Joomla	✓	✓
jQuery	✓	✗
Moodle	✓	✓
MooTools	✓	✓
Prototype	✓	✓
Roundcube	✓	✗
TYPO3	✓	✓
WikiMedia	✓	✓
WordPress	✓	✓

We consider a fix to be correct if and only if it matches the programmers' fix for the bug.

Overall Recall: (20/22) = 91%

Web Applications: VejoVis Precision

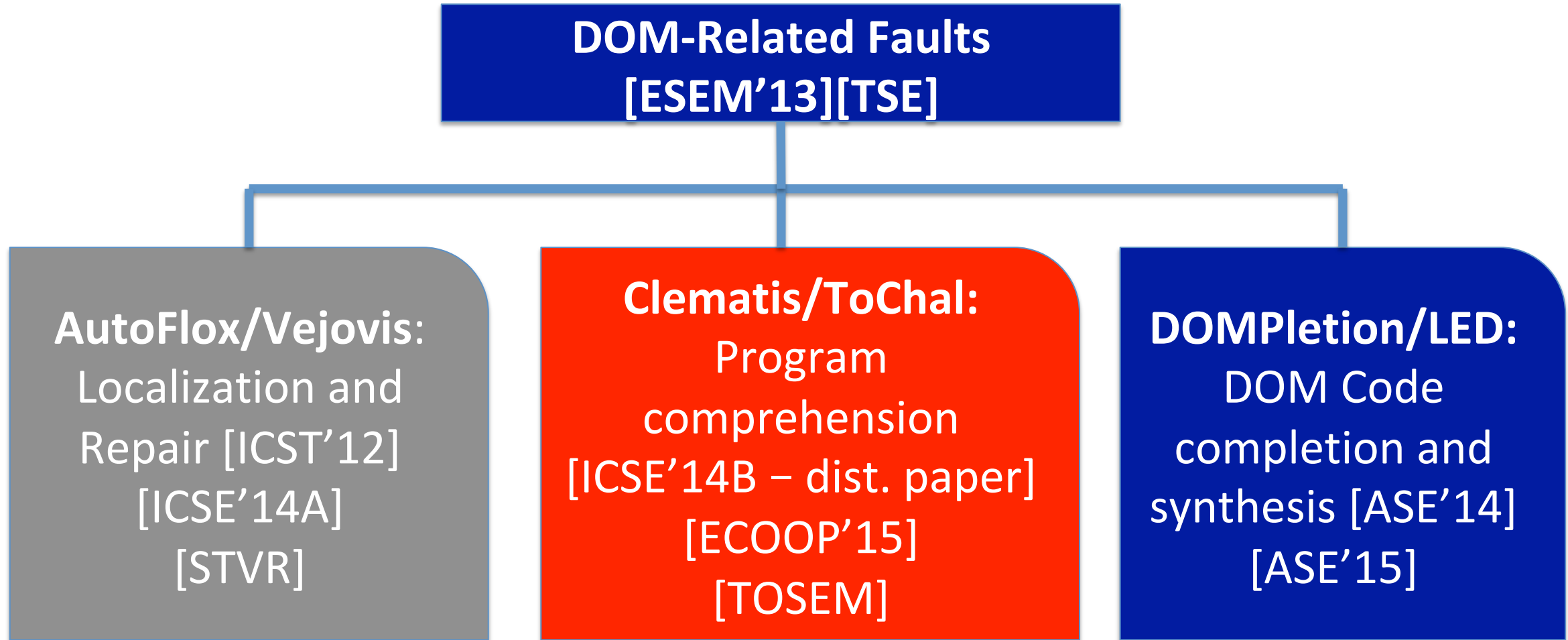
Subject	Bug Report #1	Bug Report #2
Drupal	31 / 40	1 / 4
Ember.js	1 / 2	1 / 3
Joomla	1 / 88	1 / 88
jQuery	2 / 108	-
Moodle	2 / 37	1 / 37
MooTools	2 / 2	1 / 2
Prototype	1 / 6	1 / 2
Roundcube	4 / 79	-
TYPO3	1 / 187	1 / 1
WikiMedia	6 / 24	1 / 71
WordPress	13 / 30	1 / 170

Many suggestions provided for some cases

Ranking function for fix

#1 Ranking in 13 out of 20 bugs
(#2 in 3)

Web Applications: Our Approach



Clematis: Motivation

- **Goal:** Understand and visualize dependencies between JavaScript events and the DOM
- **Challenge:** Difficult to understand the dynamic behavior and the control flow of events
 - Event propagation due to the DOM
 - Asynchronous events (e.g., timeouts)
 - DOM state changes due to events
- **Approach:** Dynamically capture execution of JavaScript applications through instrumentation and convert it to a model

Clematis: Visualization

Navigation bar with icons for home and refresh, and buttons for 'Event', 'XHR', and 'TO'. A search input field contains 'Search by Event' and a 'Search' button. A progress indicator with blue and red bars is on the right.

Source
"click"

Trace

Event type:click	onclick()	ss_next()
ss_update()	hideElem(x)	dg(x)
inlineElem(x)	Event type:load	updateNumOfLoads()
storeUserInformation()	sendStatsToServer()	ss_loaddone()
onload()		

Dom Mutations

"text" "removed" "text" "removed" "text" "added"
"text" "added"

Episode #3
Event

Source
TO:0

Trace

TID: 0	ss_slideshow()	ss_update()
hideElem(x)	dg(x)	inlineElem(x)
ss_run()	TID: 0	TID: 0
Event type:load	sts_data_collection()	updateNumOfLoads()
storeUserInformation()	sendStsToServer()	ss_loaddone()
onload()		

Episode #7
Event



Event

XHR

TO

Search by Event

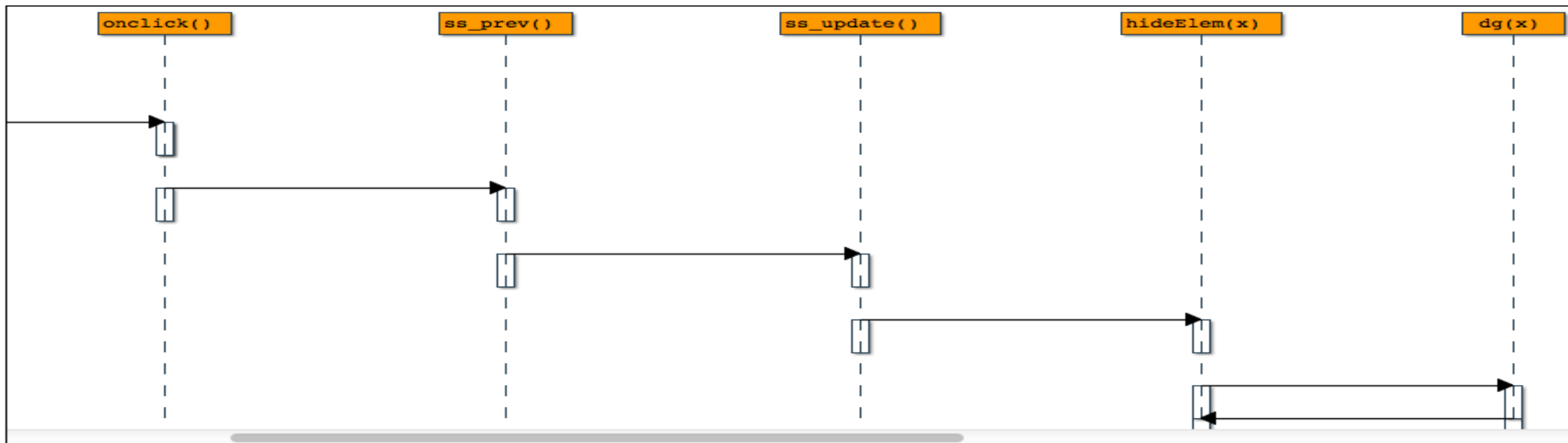
Search

Episode 5

Event Type

DOM mutations

Trace of Episode 5



phorm.js

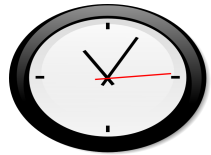
```
function ss_update() {
  ss_cur = Math.max(ss_cur, 0);

  if (ss_cur >= ss_date.length) {
    hideElem('ss_link2');
    showElem('ss_theend');
    ss_cur = ss_date.length;
    var a = dg('ss_n');
    a.innerHTML = "Final";
    if (ss_play)
      ss_playpause();
  }
}
```

Clematis: User Experiment

- Participants
 - 20 software developers from a large software company in Vancouver (they were all well versed in web development)
 - Experimental group: Clematis
 - Control group: Chrome, Firefox, Firebug (any tool of choice)
- Procedure
 - Tasks: control flow, feature location, DOM mutations, ...
- Data collection: Task completion duration & accuracy

Clematis: Results [ICSE'14 – Dist. Paper Award]



Duration



Accuracy

Task	Improvement	Task	Improvement
T1	(39%↑)	T1	(67%↑)
T2	(48%↑)	T2	(41%↑)
T3	(68%↑)	T3	(20%↑)
T4	(32%↑)	T4	(68%↑)

Task	Description
T1	Following control flow in presence of asynchronous events
T2	Finding DOM mutations caused by a DOM event
T3	Locating the implementation of a malfunctioning feature
T4	Detecting control flow in presence of event propagation

ToChal: Motivation

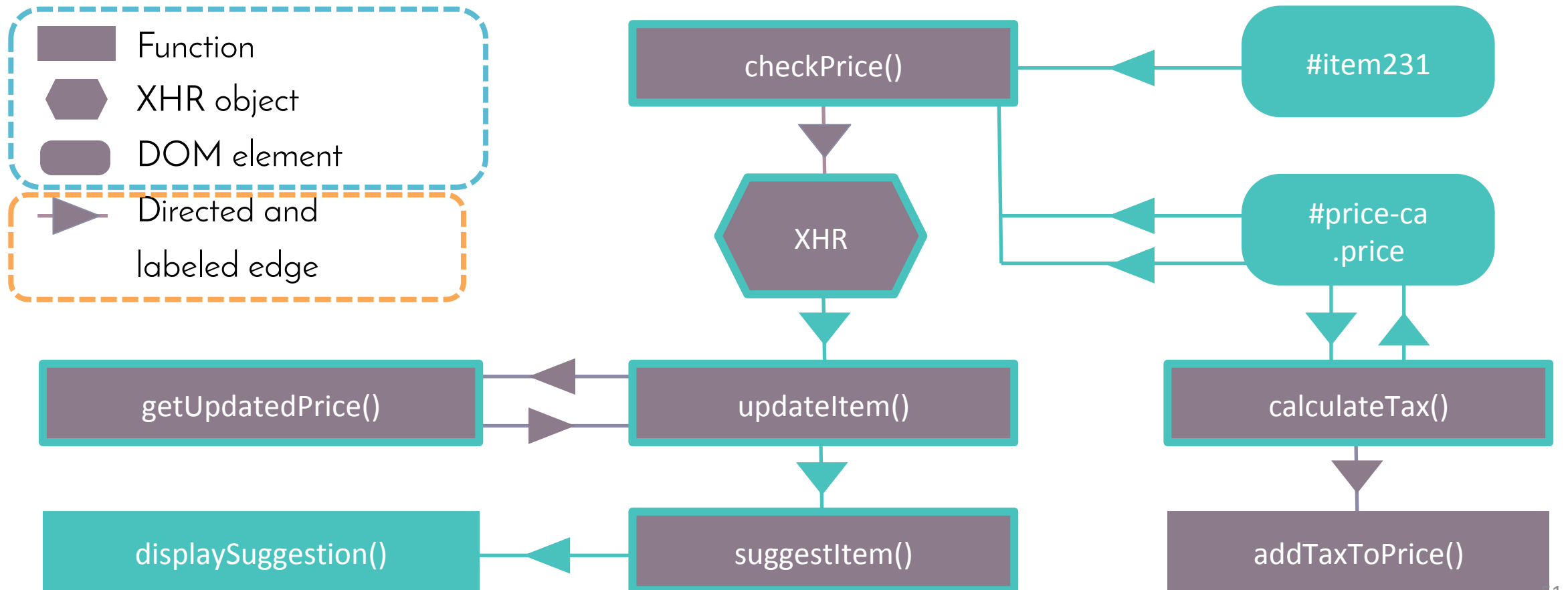
- Software must continually change to adapt to the changing environment.
- **Goal:** identifying parts of the program that are potentially affected by a change.



- Hybrid of static and dynamic analyses

ToChal: Approach

- Hybrid Flow Graph including the DOM



ToChal: User Experiment

- Question: Does Tochal help developers in practice to perform change impact analysis?
- Design:
 - 12 participants from industry
 - 4 tasks: detecting and analyzing change impact
 - Measured: task completion duration and accuracy

ToChal: User Experiment Results [ECOOP'15]



Accuracy

Task	Improvement
Total	223%↑↑



Duration

Task	Improvement
T1	78%↑↑

Task	Description
T1	Finding the potential impact of a DOM element
T2	Finding the potential impact of a JavaScript function
T3	Finding a conflict after making a new change (<u>no ranking</u>)
T4	Finding a bug in JavaScript code

Web Applications: Our Approach

DOM-Related Faults
[ESEM'13][TSE]

AutoFlox/Vejovis:
Localization and
Repair [ICST'12]
[ICSE'14A]
[STVR]

Clematis/ToChal:
Program
comprehension
[ICSE'14B]
[ECOOP'15]
[TOSEM]

DOMpletion/LED:
DOM Code
completion and
synthesis [ASE'14]
[ASE'15]

Dompletion [ASE'14]

- Perform code completion for DOM-JS interactions within the IDE
- Symbolic execution for finding DOM nodes along different paths
- Learn DOM patterns through machine learning techniques

```
1 a = document.getElementById('maincol').innerHTML;
2 -
3 - if(a == "header") {
4   ...elem = document.getElementById('headerBar');
5 - } else {
6   ...elem = document.getElementById('photoBoxes');
7 }
8 elem.getElementsByClassName('
```

Path: 0	VeryTitle	(span)	DOM Level: 1
Path: 1	photoBox	(div)	DOM Level: 1
Path: 0	topHeadAround	(a)	DOM Level: 2
Path: 1	titlePhotoBox	(span)	DOM Level: 2
Path: 1	darkdot	(span)	DOM Level: 2
Path: 1	spc	(span)	DOM Level: 2
Path: 1	rate	(select)	DOM Level: 2
Path: 1	dot	(span)	DOM Level: 3

LED [ASE'15]

- Synthesize DOM element selectors (i.e., CSS selectors) automatically through programmer-supplied examples using an SMT solver

The screenshot displays the LED tool interface, which is used for synthesizing DOM element selectors. The interface is divided into two main sections: "Step 1: Drag and Drop DOM elements" and "Step 2: Configure Options".

Step 1: Drag and Drop DOM elements

This section shows three examples of DOM elements being selected. Each example consists of a text input field and a set of control buttons (up/down arrows, minus, plus, and X). The examples are:

- Example 1:** A#nav-questions ... (with the minus button highlighted)
- Example 2:** A#nav-users ... (with the plus button highlighted)
- Example 3:** A#nav-unanswered ... (with the plus button highlighted)

Step 2: Configure Options

This section allows the user to configure the options for the generated selector. The options include:

- id (down)
- classes (up down)
- tag (up down)
- mix (up)

Additional options include Depth (4) and Max time (10). There is a checkbox for "Select only selected elements".

The "Must use (one per line):" section contains the following selectors:

```
April 2015 Community  
Hot Meta Posts  
43 Broaden the jsFiddle (et. al.)  
filter to disallow links as the
```

The "Ignore (one per line):" section contains the following selectors:

```
html  
body  
Editing post with same tag  
multiple times
```

A "Generate Selector" button is located at the bottom of the configuration section. Below this button, the "Total CSS Selectors: 1" is displayed, along with the generated selectors:

```
#nav-users  
#nav-unanswered
```

The "Jobs near you" section is visible at the bottom of the page.

Web Applications: Our Approach

DOM-Related Faults
[ESEM'13][TSE]

AutoFlox/Vejovis:
Localization and
Repair [ICST'12]
[ICSE'14A]
[STVR]

Clematis/ToChal:
Program
comprehension
[ICSE'14B]
[ECOOP'15]
[TOSEM]

DOMpletion/LED:
DOM Code
completion and
synthesis [ASE'14]
[ASE'15]

Our Recent Work

MVC Frameworks for JS

- Static analysis tools for JavaScript MVC Frameworks such as AngularJS [ICSE'15][ASE'17]



Server-side JavaScript

- Program Comprehension for server side JavaScript such as Node.js [ICSE'16][ICSE'18]



Talk Outline

- Motivation and Goals
- Empirical Study of reliability
- Reliability Improvements
- **Conclusions and Future Directions**

Conclusions and Future Work

- **DOM-related faults prevalent in JavaScript**
 - Responsible for 2/3rds of all real-world bugs, and 80% of the most critical ones
 - Techniques to fix the bugs, understand root causes, and write error-free code
 - Achieved considerable improvement in reliability and understandability
- **Future Work**
 - Extensions for JavaScript in the IoT context (ThingsJS project)
 - Understanding the Security Implications of JavaScript Faults

<http://blogs.ubc.ca/karthik/software>



Karthik Pattabiraman



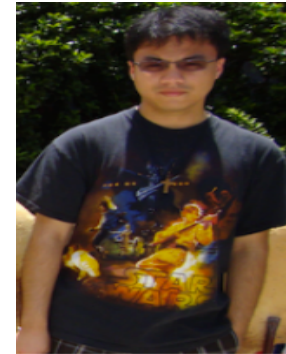
Ali Mesbah



Saba Alimadadi



Kartik Bajaj



Frolin Ocariza



Sheldon Sequira

