

Empirical Studies of JavaScript-based Web Application Reliability



Karthik Pattabiraman¹

Frolin Ocariza¹

Kartik Bajaj¹

Ali Mesbah¹

Benjamin Zorn²

¹ University of British Columbia (UBC), ²Microsoft Research (MSR)

My Research

- **Building reliable and secure software applications**
- **Compiler & runtime techniques for error resilience**
 - Partitioning data for differential resilience [ASPLOS'11]
 - Error detection in different programs [DSN'12][DSN'13]
 - Fault Injection techniques and tools [DSN'14][ISPASS'14]
- **This tutorial**
 - Reliability of modern web applications (Part 1)
 - Tools for building robust web applications (Part 2)

Web 2.0 Applications



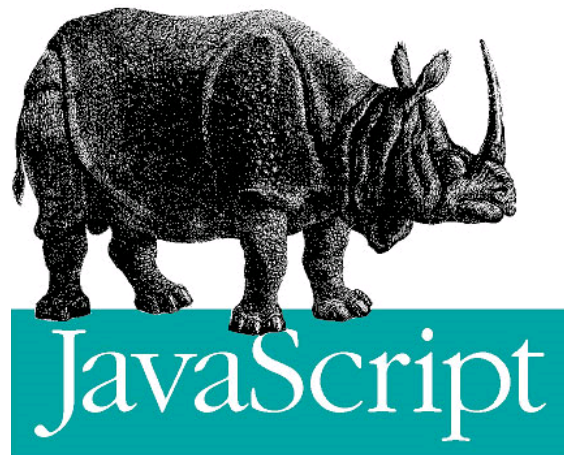
Web 2.0 Application: Amazon.com



Web 2.0 applications allow rich UI functionality within a single web page

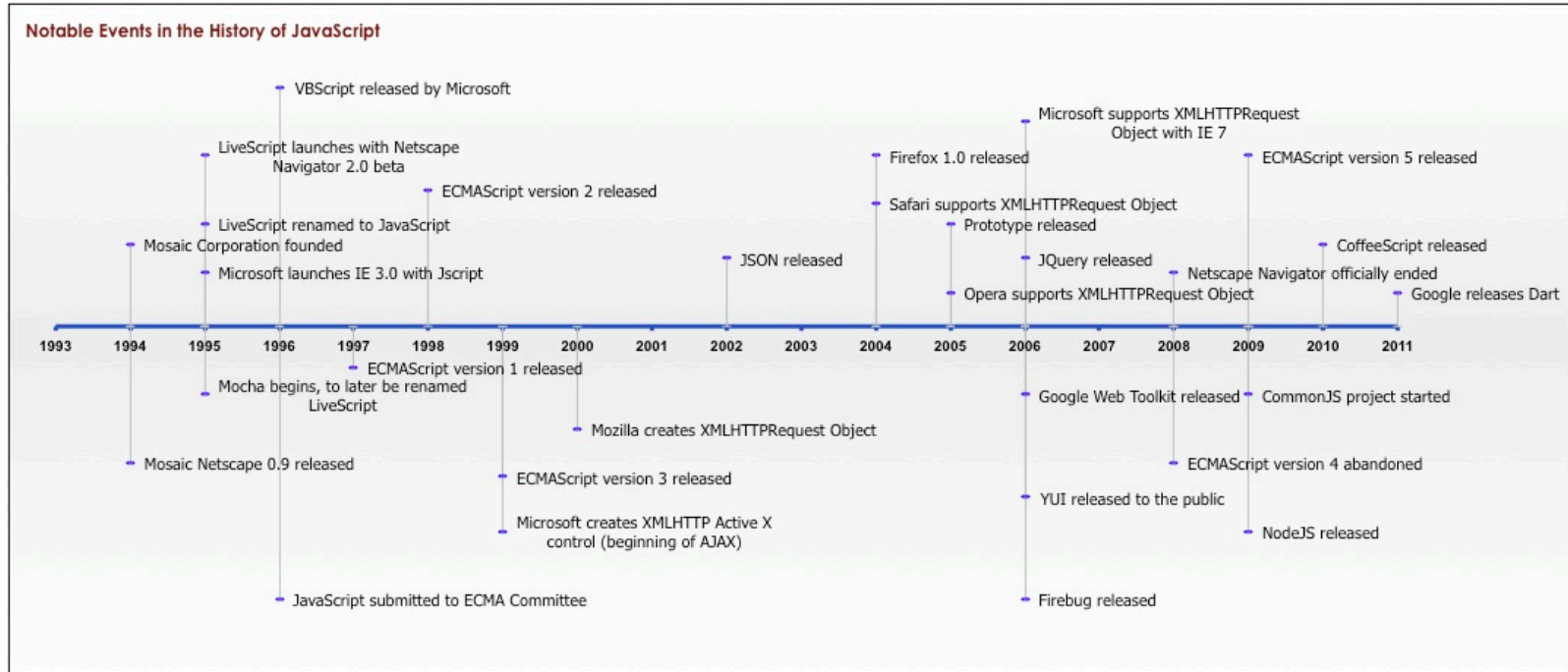
Modern Web Applications: JavaScript

- 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)

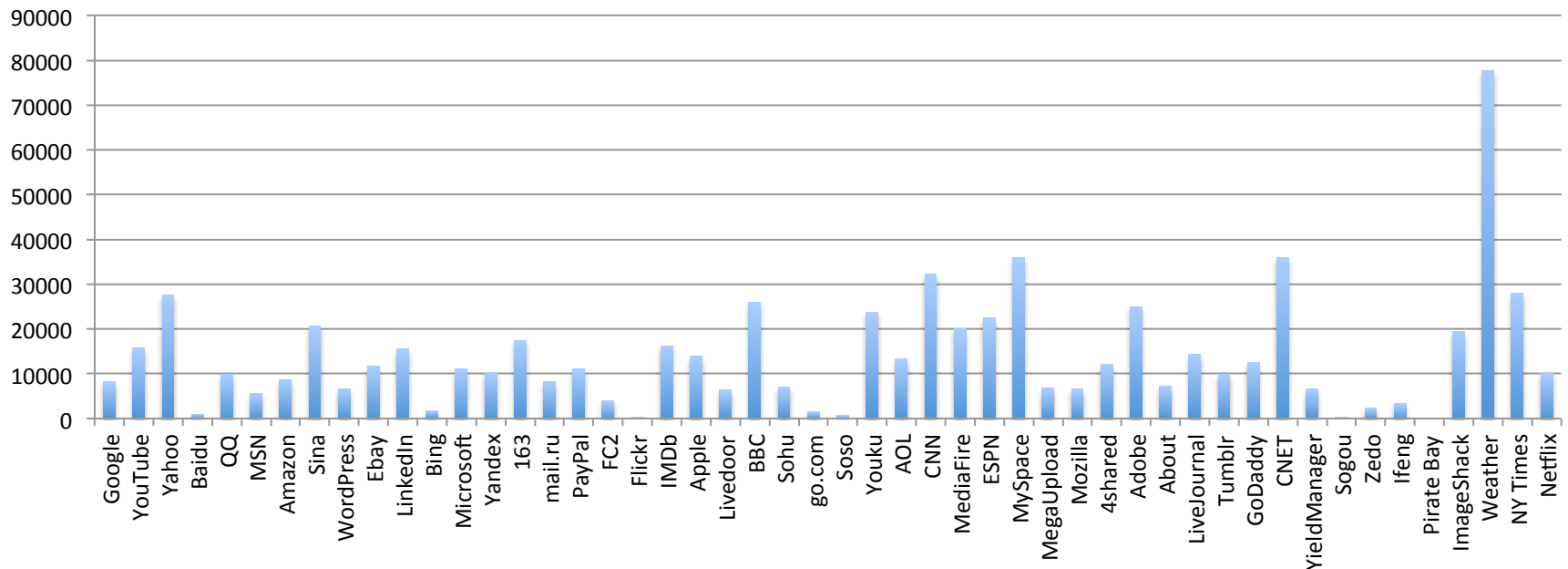


JavaScript (JS) had to “look like Java” only less so, be Java’s dumb kid brother or boy-hostage sidekick. Plus, I had to be done **in ten days** or something worse than JS would have happened – Brendan Eich (Inventor of JavaScript)

JavaScript: Prevalence

- 97 of the Alexa top 100 websites use JavaScript
- Thousands of lines of code, often $> 10,000$

Lines of code



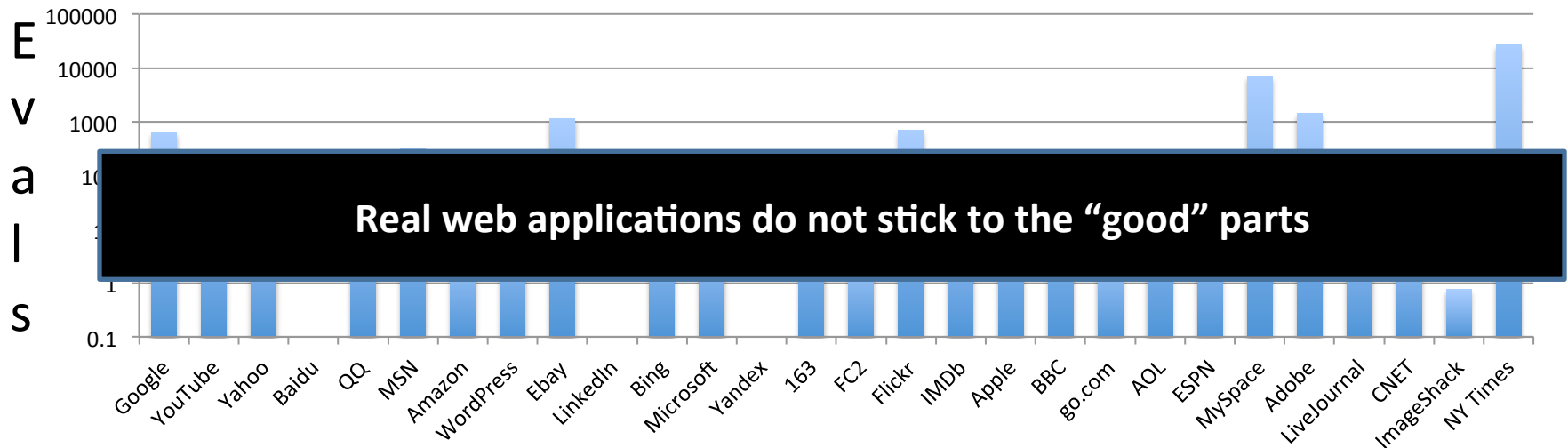
JavaScript: “Good” or “Evil” ?



Vs



Eval Calls (Source: Richards et al. [PLDI-2010])



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

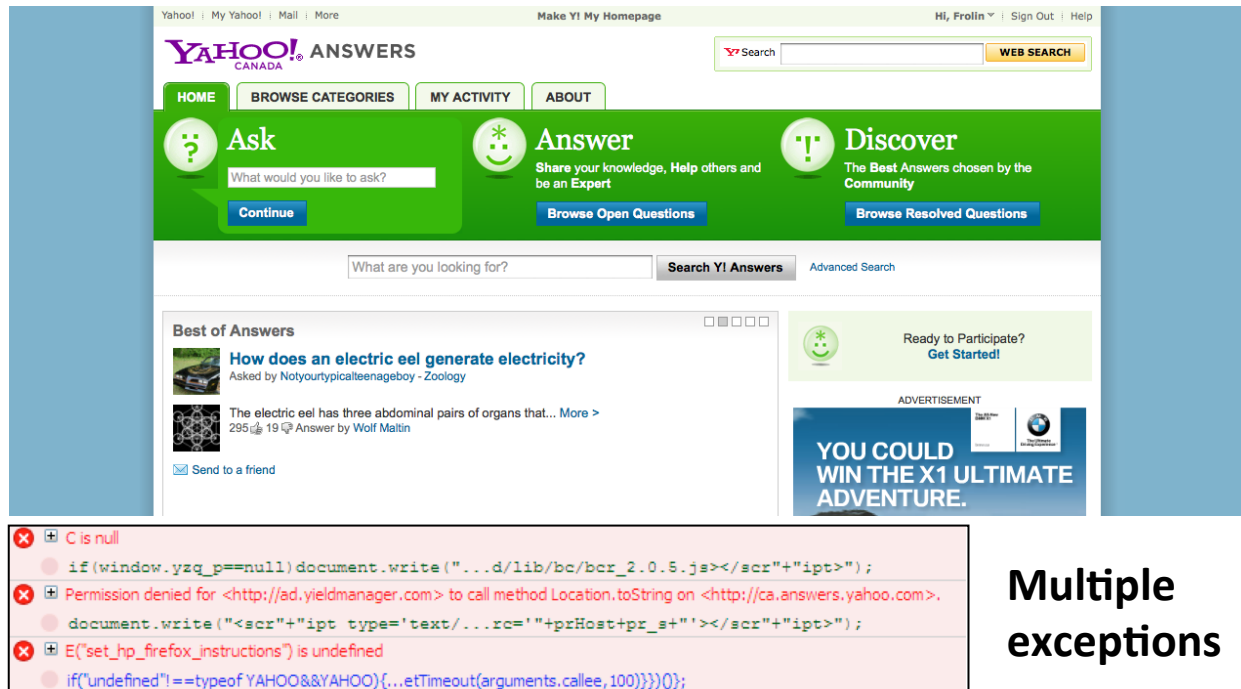


This Talk

- Motivation and Approach
- Three approaches for measuring JS Reliability
 - Error Messages [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - Bug Reports [ESEM 2013] – With F. Ocariza, K. Bajaj, A. Mesbah
 - Stack Overflow Reports [MSR 2014] – With F. Ocariza, A. Mesbah
- Conclusion and Next Steps

JSER: JavaScript Error Messages

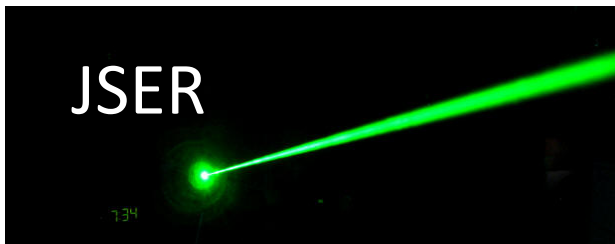
- All exceptions thrown are logged to JS console



Multiple exceptions

JSER: Error Messages Vs. Static Analysis

- No false positives unlike static analysis
- Capture interactions with third-party code (advertisements)
- Capture interactions with the DOM



Vs.



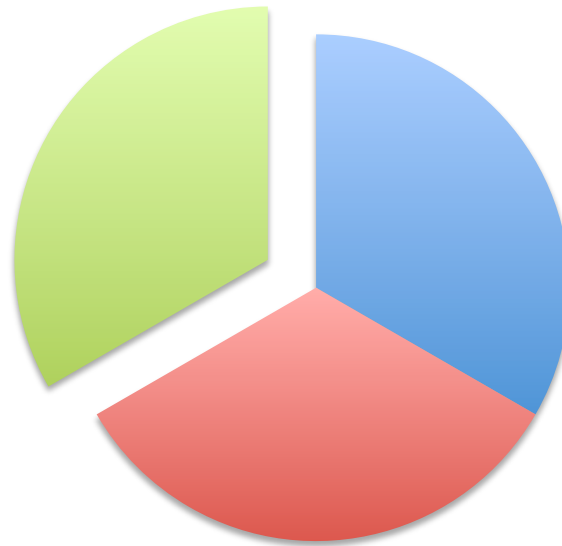
JSER: Tools

- Chose 50 web applications from Alexa top 100
- Created Selenium tests for normal interactions
- Capture JavaScript Errors printed to Firebug



JSER: Research Questions

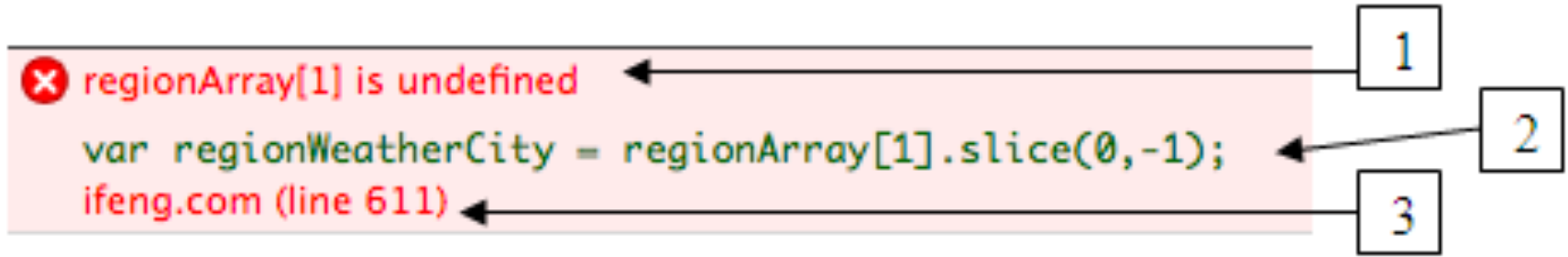
**Do errors occur in web apps
and if so, what categories do
they fall in ?**



How do errors correlate
with static and dynamic
characteristics of the app?

How do errors vary by
speed of testing ? Are
they all deterministic ?

JSER: Method

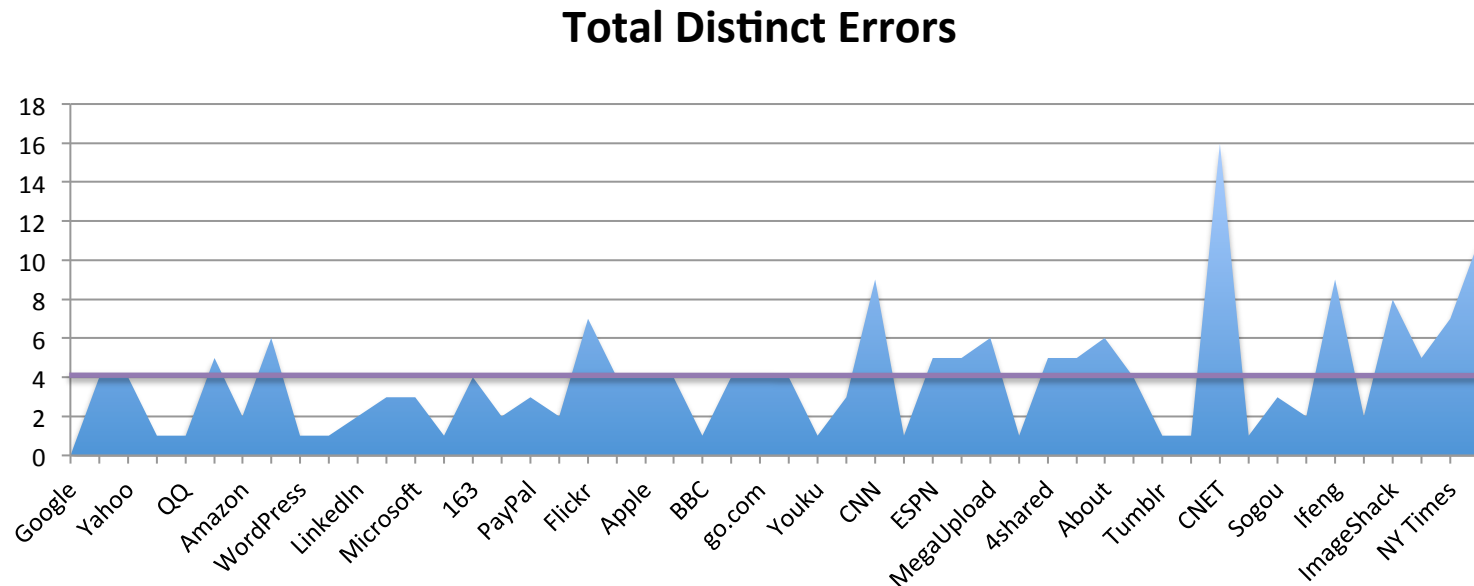


1. Description of error message
2. Line of code corresponding to error
3. Domain number and line number

Two errors are different if any attribute is different

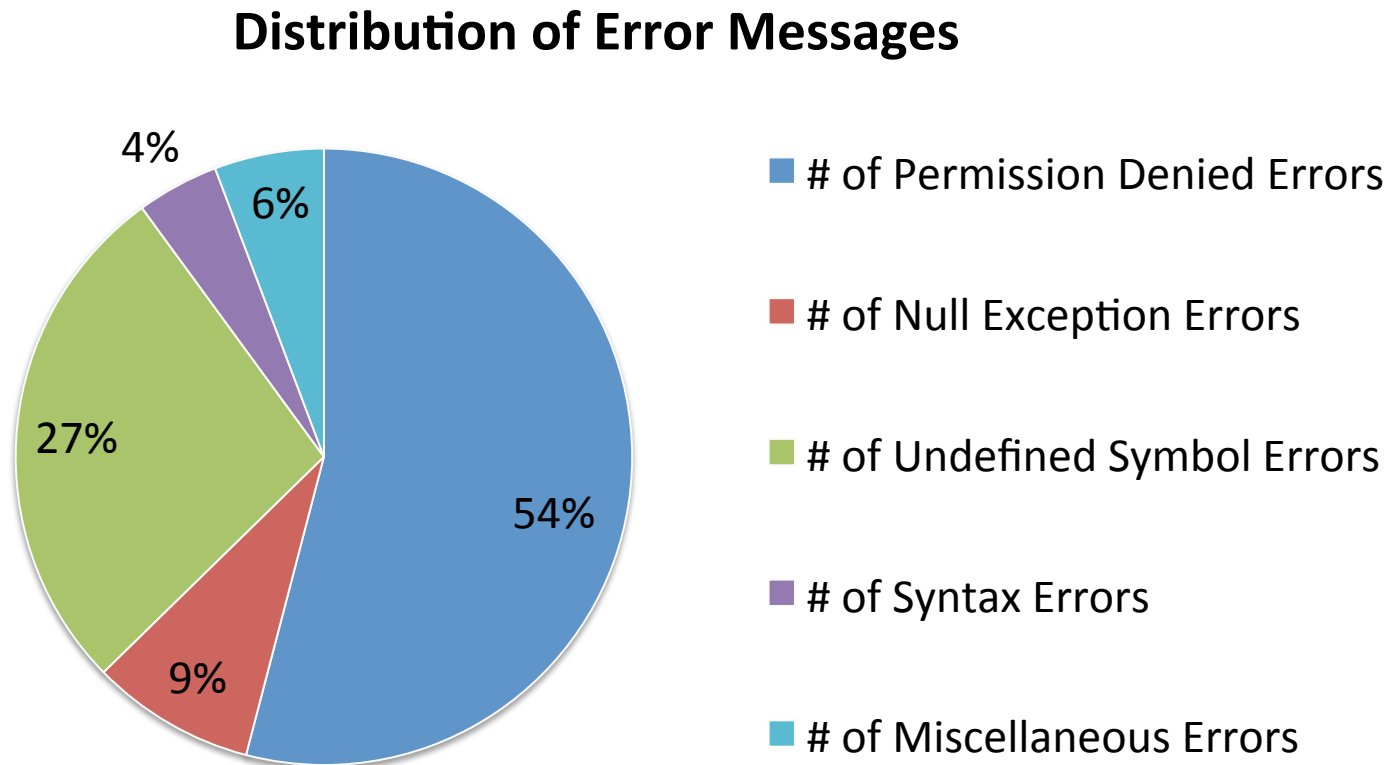
JSER: Error Frequencies Results

- **Average of 4 distinct error messages for each app**
 - **Standard dev: 3**
 - **Max: 16 (Cnet)**
 - **Min: 0 (Google)**



JSER: Error Classification Results

- 94 % of errors fall into four predominant categories



JSER: Research Questions

Errors occur in web applications
(4 per application on average)
and fall into four categories

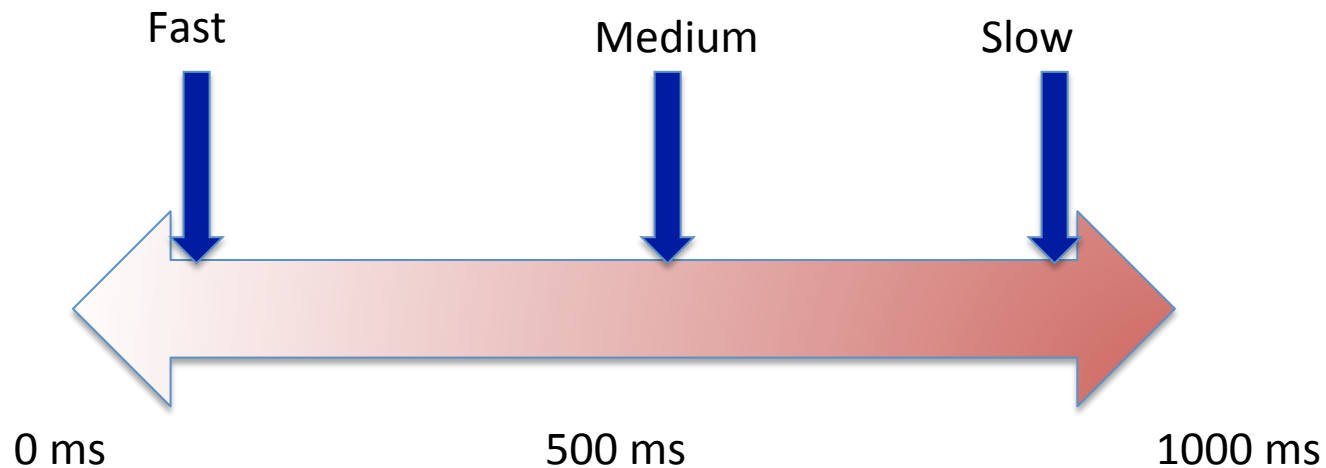


How do errors correlate
with static and dynamic
characteristics of the app?

**How do errors vary by
speed of testing ? Are
they all deterministic ?**

JSER: Effect of Testing Speed

- Varied testing speed for replaying events in Selenium
- Performed three executions in each testing speed

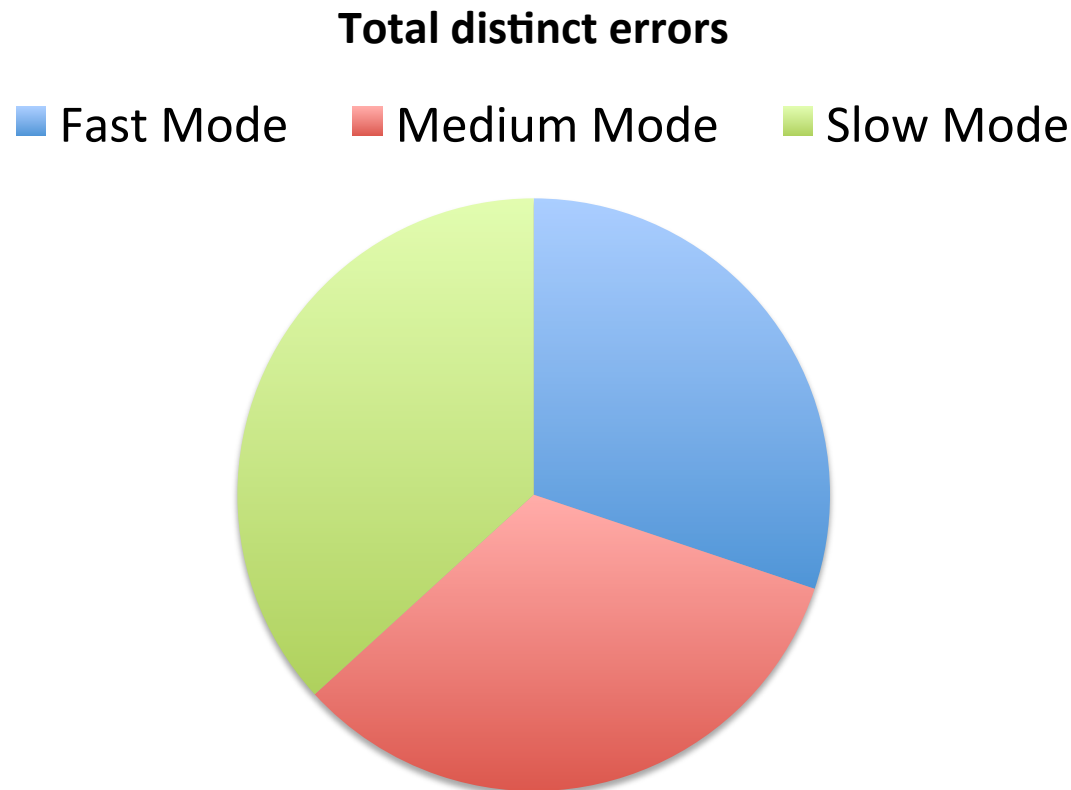


JSER: Testing Speed Results (CNN)

Error Message (shortened)	F 1	F 2	F 3	M 1	M 2	M 3	S 1	S 2	S 3
Permission Denied for view.atdmt.com to call <fname> on marquee.blogs.cnn.com	4	4	4	1	3	3	2	2	3
targetWindow.cnnad showAd is not a function	0	2	5	0	0	0	0	0	0
window.parent.CSIManager is un- defined	0	0	0	0	0	0	1	1	0

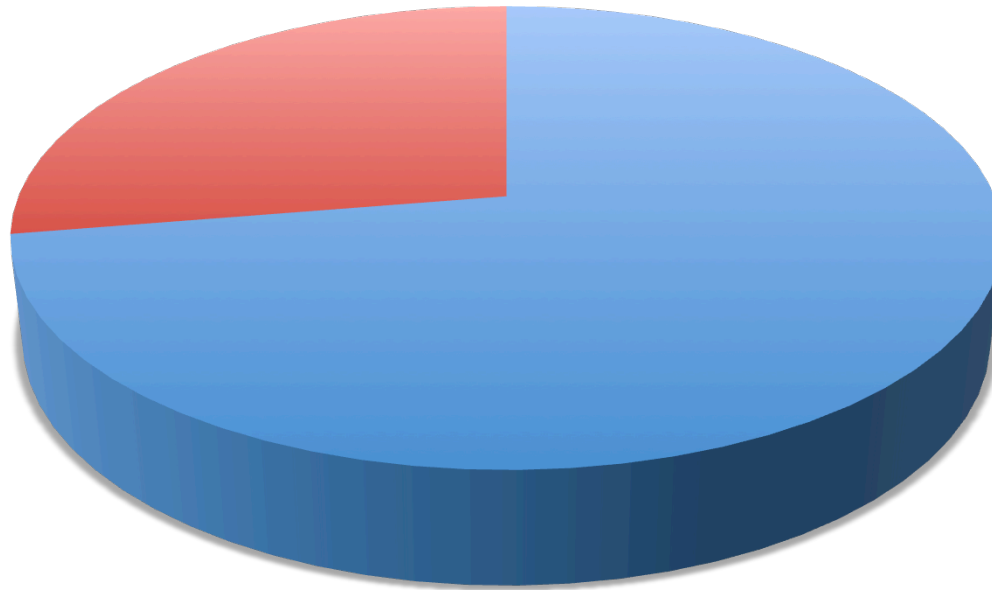
JSER: Effect of Testing Speed

- All three testing modes expose different errors



JSER: Non-Determinism

- More than 70% of errors: non-deterministic



■ Total non-deterministic errors

JSER: Research Questions

Errors occur in web applications
(4 per application on average)
and fall into four categories



**How do errors correlate
with static and dynamic
characteristics of the app?**

Error occurrences vary
with speed of testing.
About 70% of errors
are non-deterministic.

JSER: Static/Dynamic Characteristics

Static Characteristics

Measured using Phoenix & Firebug plugins

- Alexa Rank
- Bytes of JavaScript code
- Number of domains
- Domains containing JS

Dynamic Characteristics

From Richards et al. [PLDI – 2010]

- Number of called functions
- Number of eval calls
- Properties deleted
- Object inheritance overridings

JSER: Correlations Summary

Static Characteristics

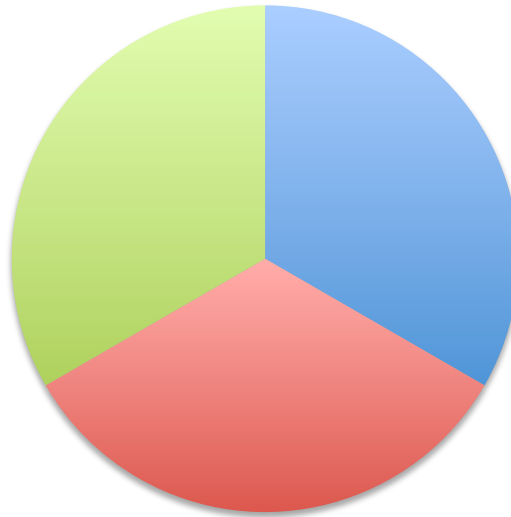
- **Alexa Rank**
- Bytes of JavaScript code
- **Number of domains**
- **Domains containing JS**

Dynamic Characteristics

- Number of called functions
- Number of eval calls
- Properties deleted
- Object inheritance overridings

JSER: Research Questions

Errors occur in web applications
(4 per application on average)
and fall into four categories



Errors correlate with Alexa
rank, no of domains but
not with loc or eval calls

Error occurrences vary
with speed of testing.
About 70% of errors
are non-deterministic.

JSER: Implications of the Results

- **Programmers**
 - Need to make code robust against other code/scripts
 - Make sure interactions with DOM are checked
- **Testers**
 - Perform integration testing to see effects of ads
 - Need to test at multiple testing speeds, multiple times
- **Static analysis tool developers**
 - Target most common classes of errors
 - Need to model the DOM in the analysis



This Talk

- Motivation and Approach
- Three approaches for measuring JS Reliability
 - Error Messages [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - Bug Reports [ESEM 2013] – With F. Ocariza, K. Bajaj, A. Mesbah
 - Stack Overflow Reports [MSR 2014] – With F. Ocariza, A. Mesbah
- Conclusion and Next Steps

Bug Report Study: Goals

- What errors/mistakes ***cause*** JavaScript faults?



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



Bug Report Study of twelve popular, Open Source JavaScript Applications



Bug Report Study: Objects

Eight JavaScript Web Applications

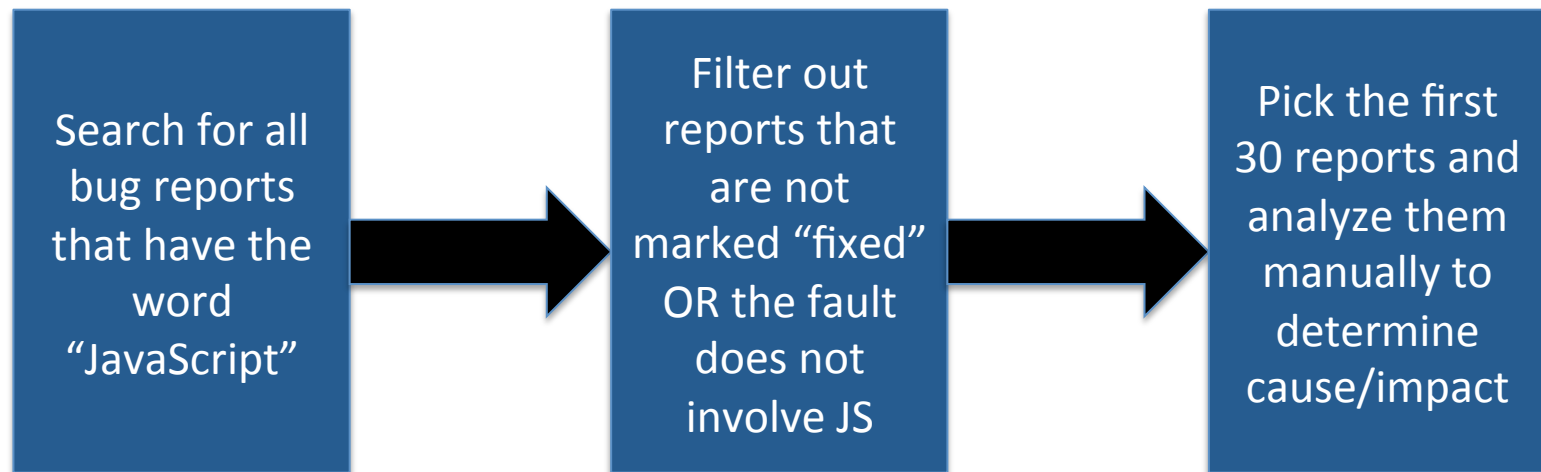


Four JavaScript Libraries



Bug Report Study : Method

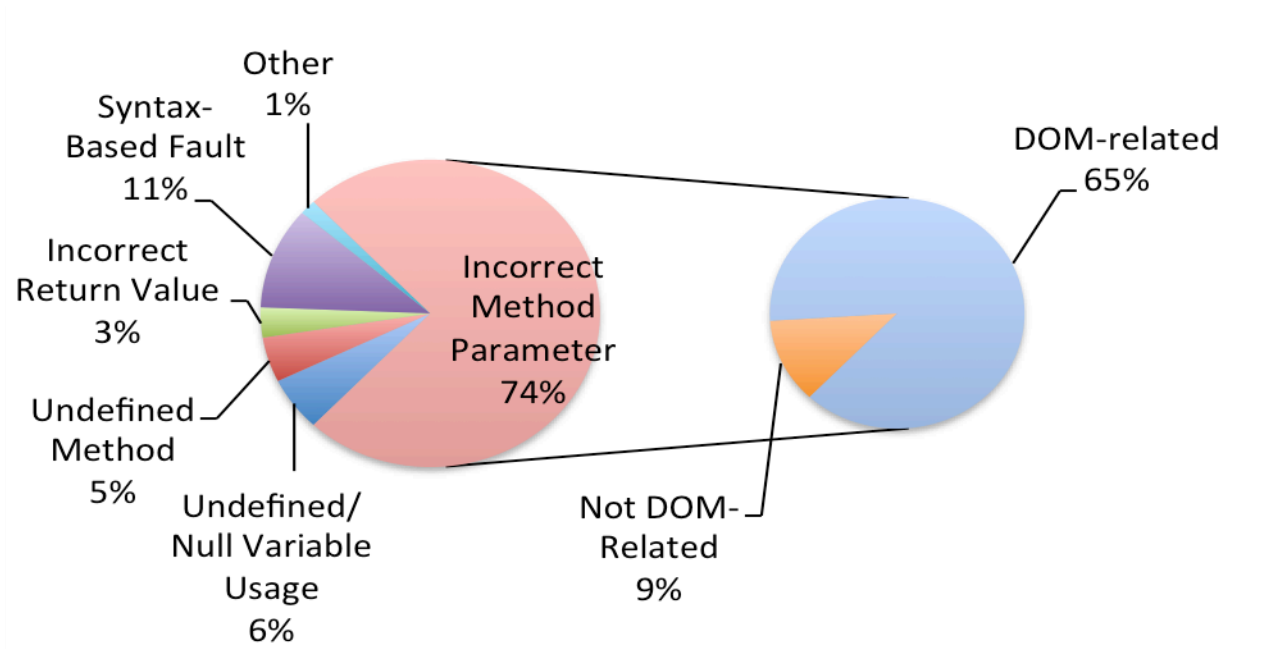
- Collect bug reports from bug repositories
 - Focus on bugs that are marked fixed to avoid spurious bugs
 - Organized into a uniform format (XML file)



Bug Report Study: Questions

- **RQ1:** What types of JavaScript *faults* occur in web apps?
- **RQ2:** What is the impact of JavaScript faults ?
- **RQ3:** How long does it take to fix a JavaScript fault?
- **RQ4:** Are JavaScript faults browser-specific ?

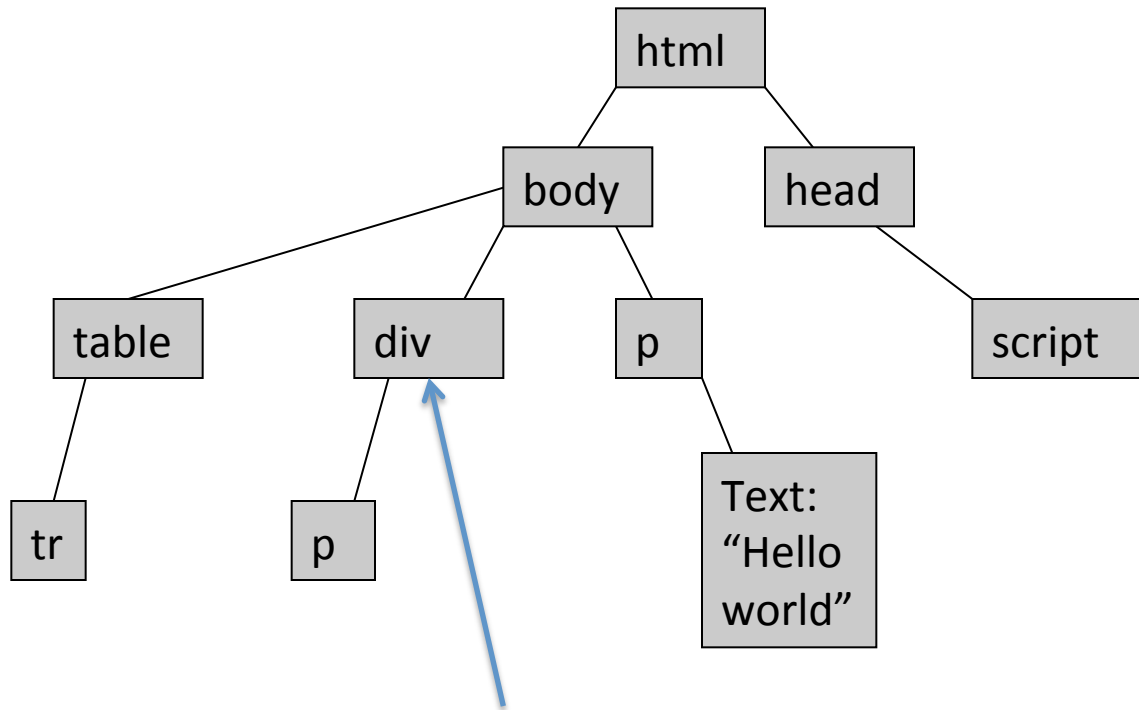
Bug Report Study: Categories



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 around **two-thirds of JavaScript Faults**

Bug Report Study: DOM



Want to retrieve this
element

Bug Report Study: DOM-Related Faults

JavaScript code: `var x = document.getElementById("elem");`

↑ ↑
Will return null Inexistent ID

div

DOM:

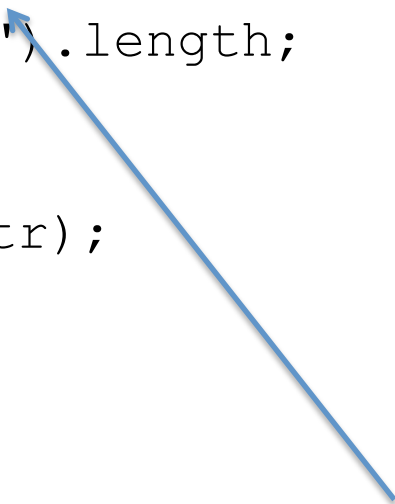
id: elem

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

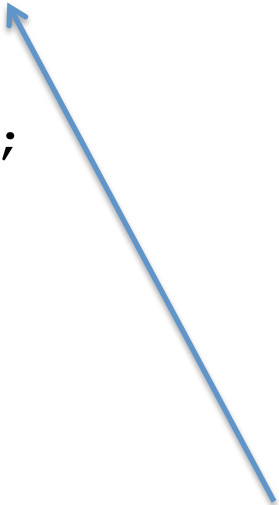
```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("##" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```



Retrieved string
via XHR

DOM-Related Fault: Example

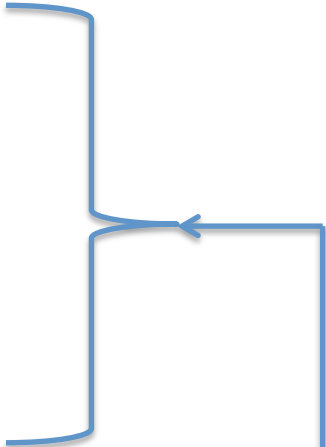
```
var elem, retrievedStr = [Retrieved via XHR];  
var dotsInStr = retrievedStr.split(".").length;  
if (dotsInStr == 0) {  
    var prefix = "id_";  
    elem = $("#" + prefix + retrievedStr);  
}  
else {  
    elem = $(retrievedStr);  
}  
elem[0].focus();
```




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

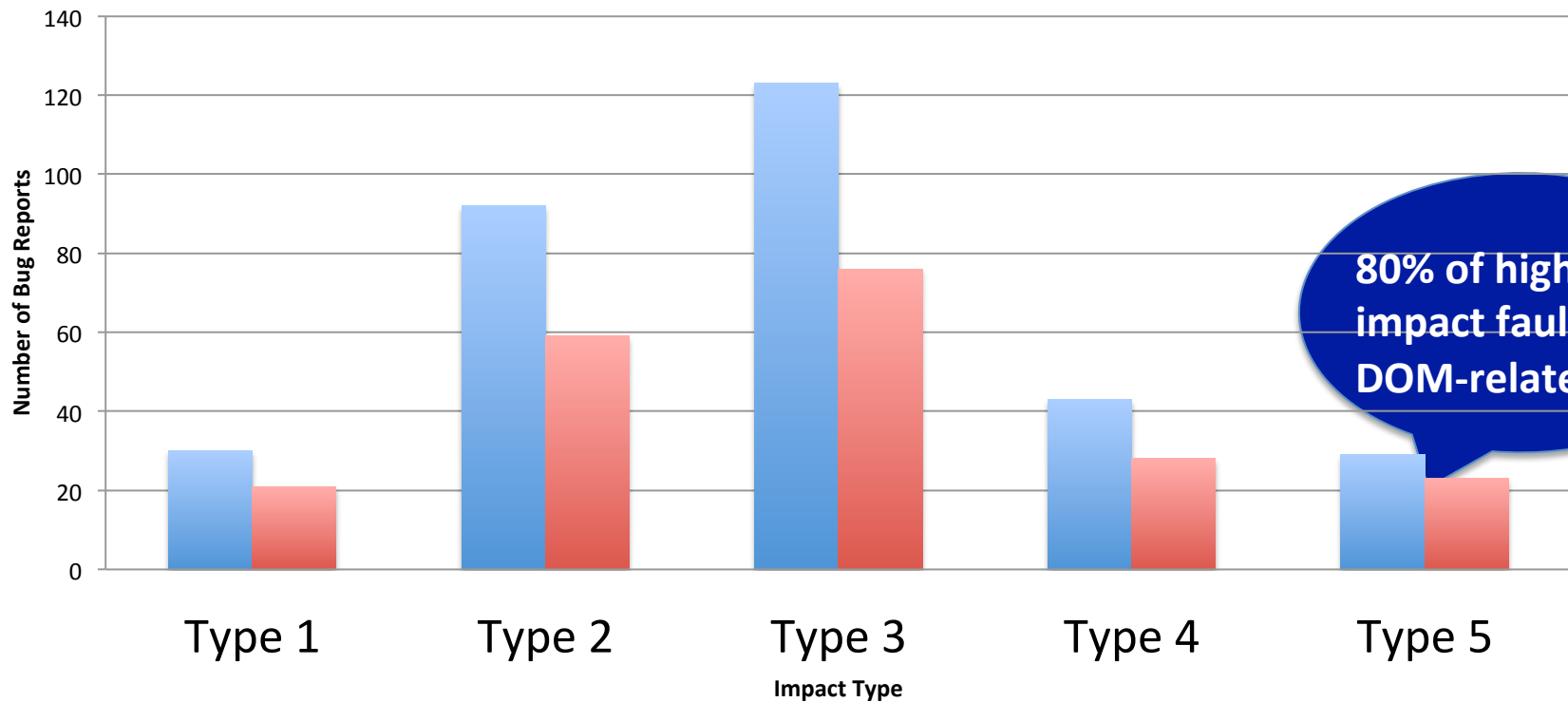
```
var elem, retrievedStr = [Retrieved via XHR];
var dotsInStr = retrievedStr.split(".").length;
if (dotsInStr == 0) {
    var prefix = "id_";
    elem = $("#" + prefix + retrievedStr);
}
else {
    elem = $(retrievedStr);
}
elem[0].focus();
```



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

Bug Report Study: Impact

- Impact Types – Based on Bugzilla [ICSE'11]
 - Type 1 (lowest impact), Type 5 (highest impact)



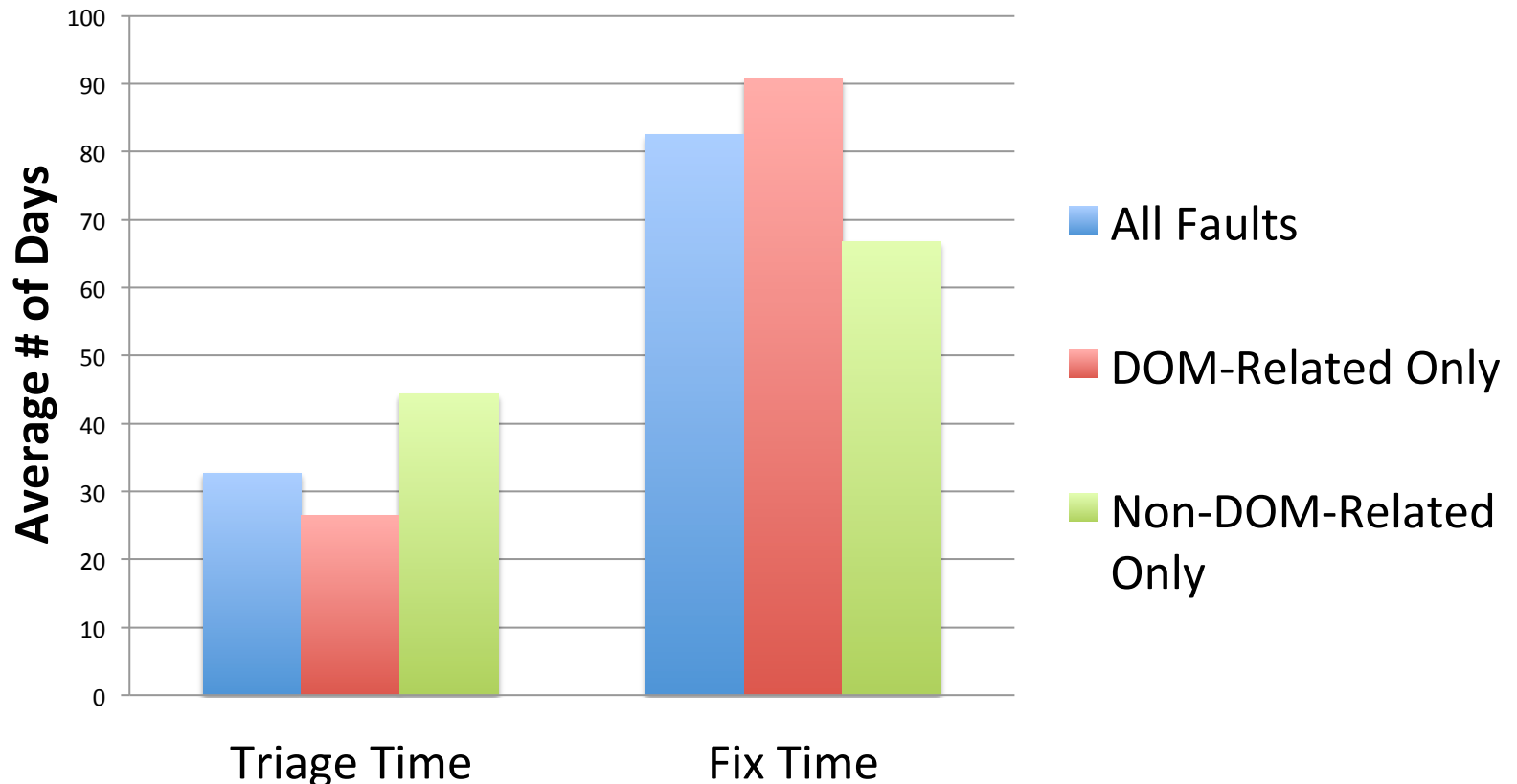
80% of highest impact faults are DOM-related

■ All Faults

■ DOM-Related Faults Only

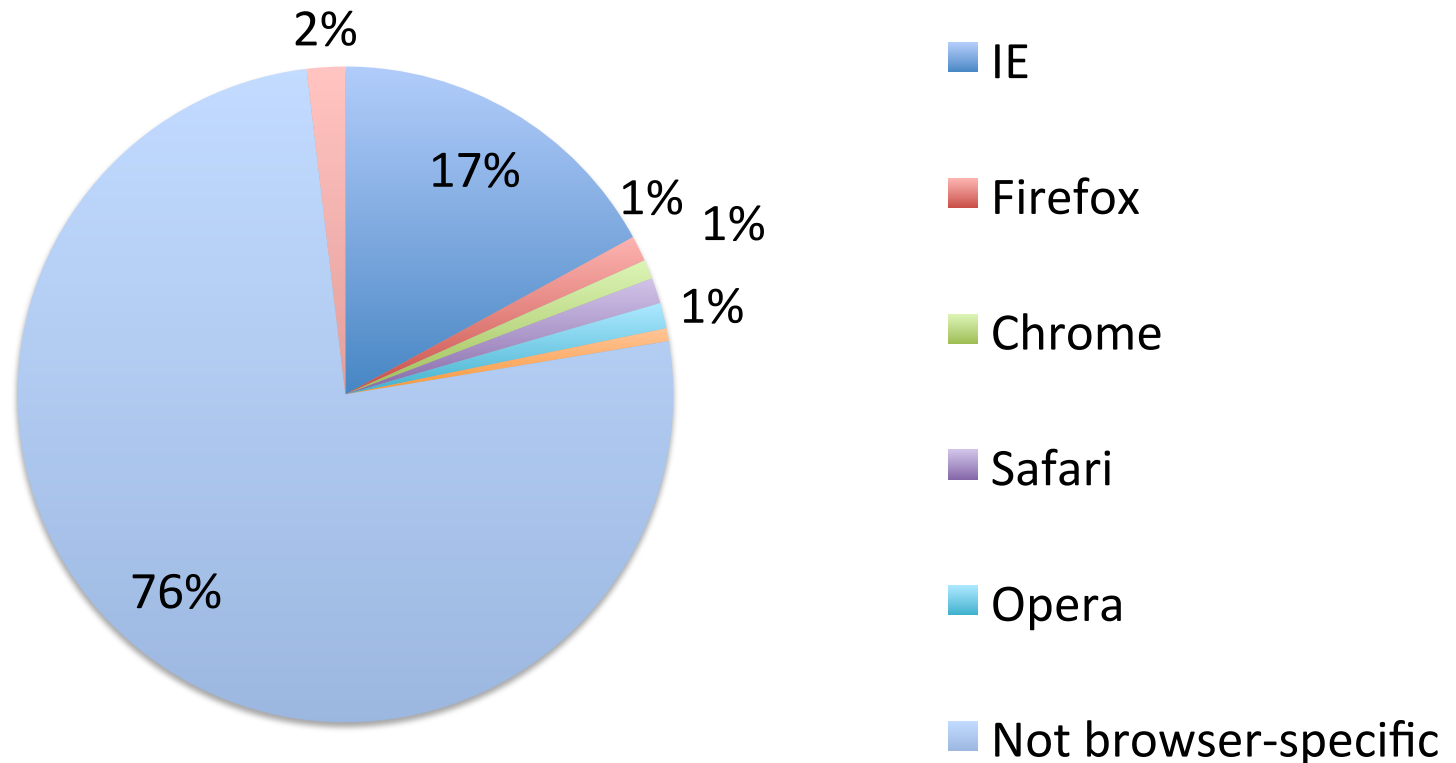
Bug Report Study: Fix Times

- **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: Summary

- **Bug report study of 12 applications: JS faults**
 - Over 300 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
- **Need robust solutions for DOM-related faults**
 - Fixing, Understanding and writing correct code

This Talk

- Motivation and Approach
- Three approaches for measuring JS Reliability
 - Error Messages [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - Bug Reports [ESEM 2013] – With F. Ocariza, K. Bajaj, A. Mesbah
 - Stack Overflow Reports [MSR 2014] – With F. Ocariza, A. Mesbah
- Conclusions and Next Steps

StackOverflow: Background

- Stack Overflow
 - QA website for programmers
 - Started in 2008
 - 4,125,638 questions asked from Jan'09 to Dec'12
 - 500,000+ questions related to web development
- Questions directly asked/answered by developers
 - Followed by discussion in comments

StackOverflow: Example

UIWebView intermittently denied access to html5 database storage



5



1

I have an iphone app that enables users to login via a native `UIView` on ios, that then fires up a `UIWebView` to display the main content. The webapp uses database storage to retain some of the content locally.

However, every now and then, the webapp will fail to load when attempting to access the database with the following message appearing in the logs.

```
..... sandboxd[3203] : APPNAME(3201) deny file-write-create /Databases.db
```

iphone

ios

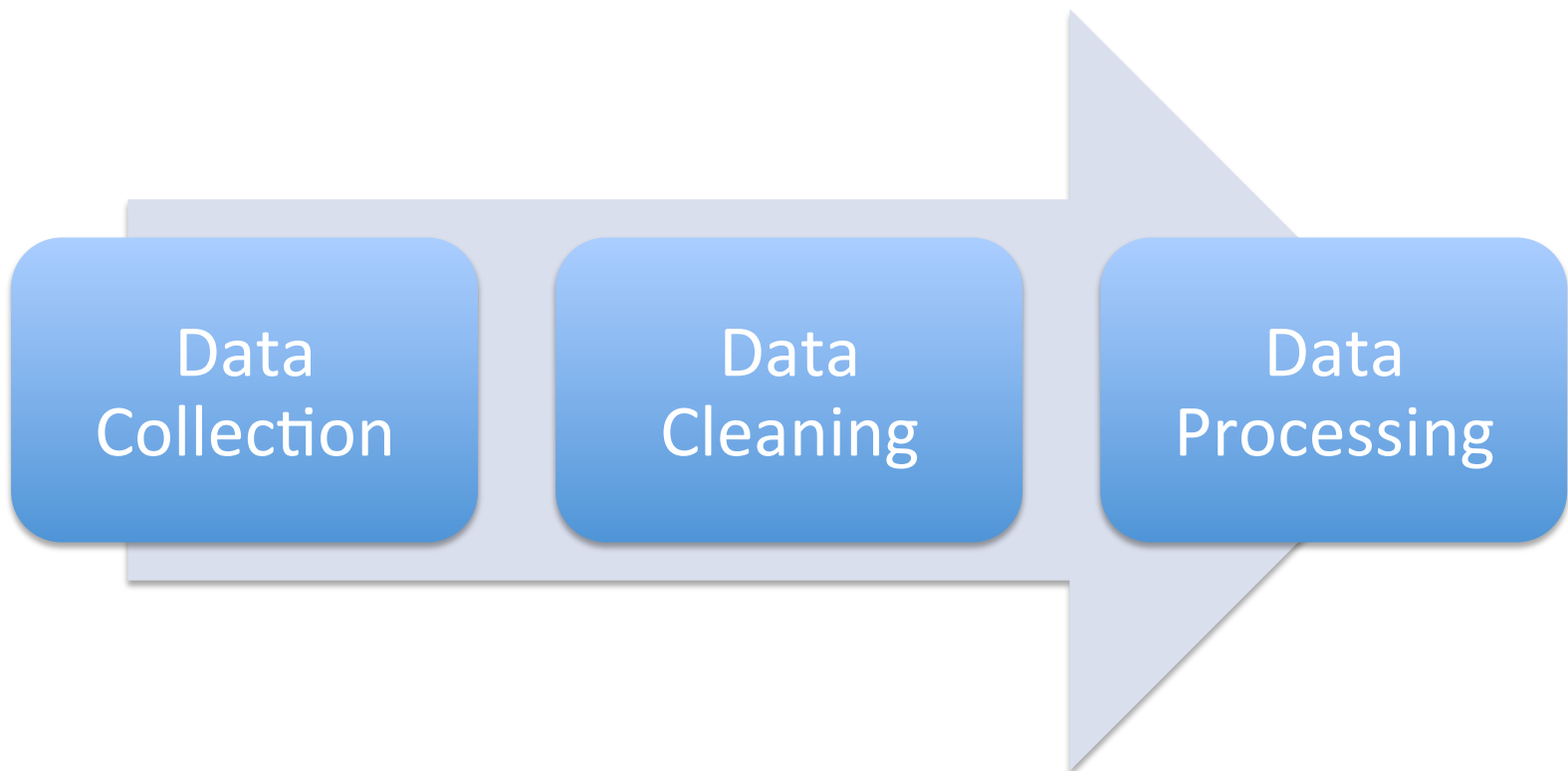
html5

uiwebview

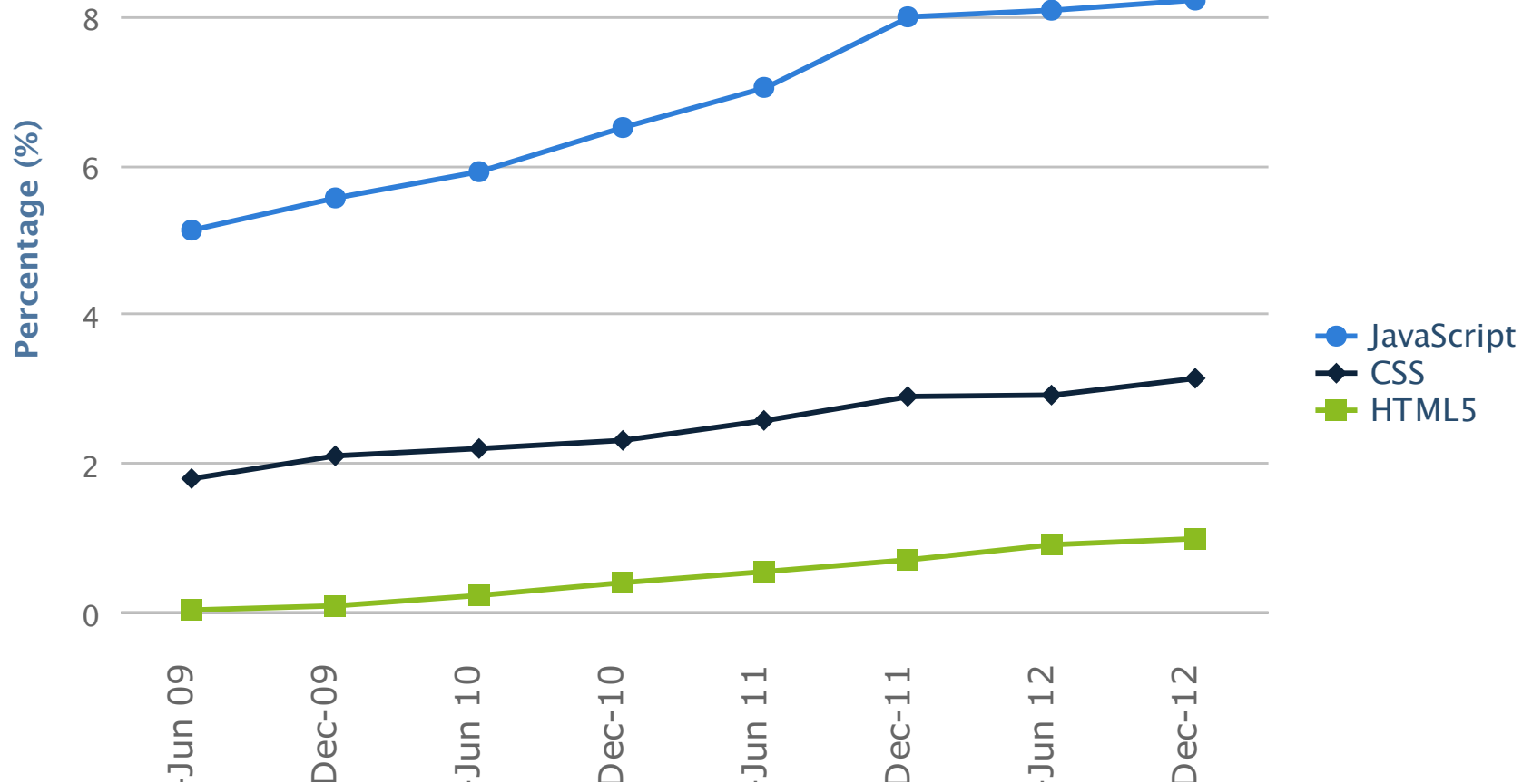
local-storage

StackOverflow: NLP Analysis

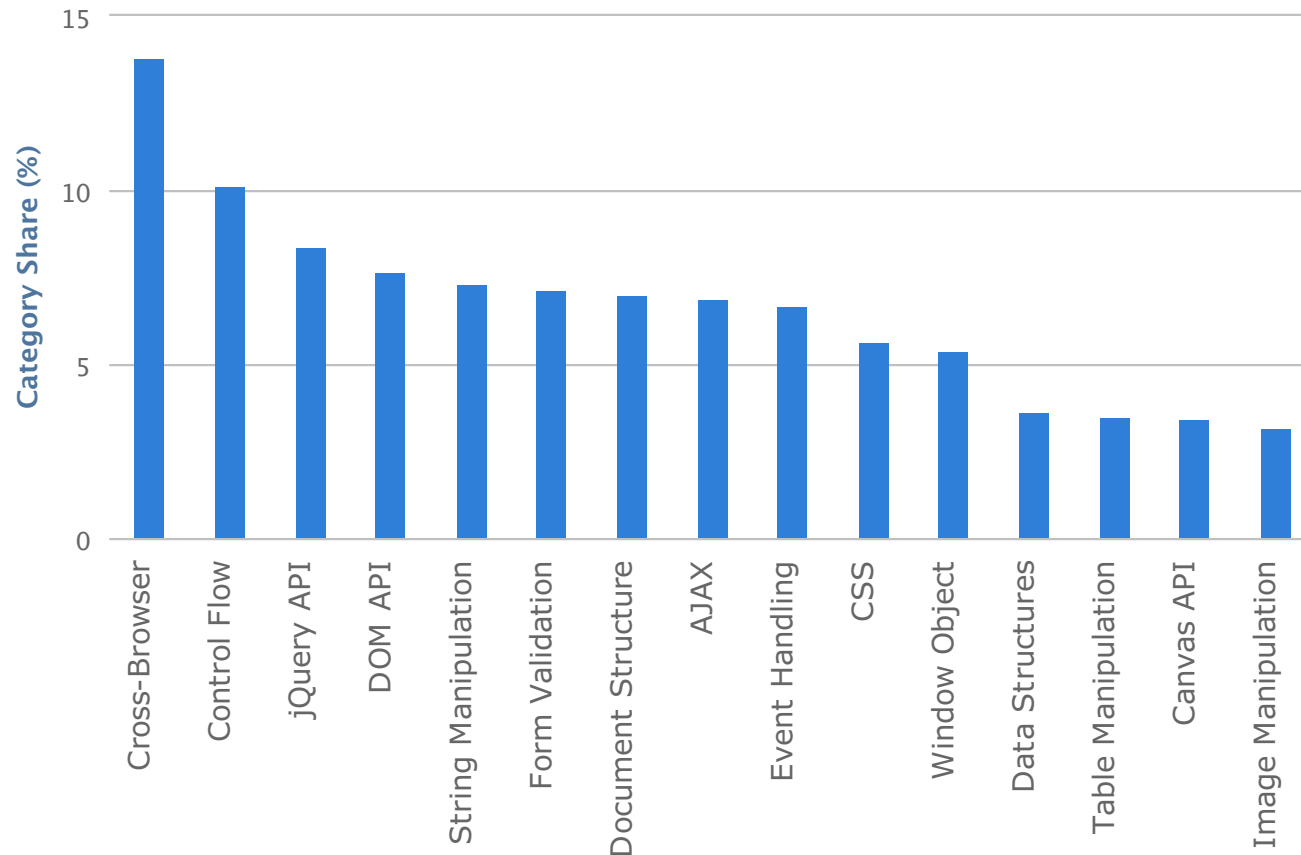
- Filter web-related questions based on tags provided
- Analyzing the text provided in the questions and answers (Latent Dirichlet Allocation)



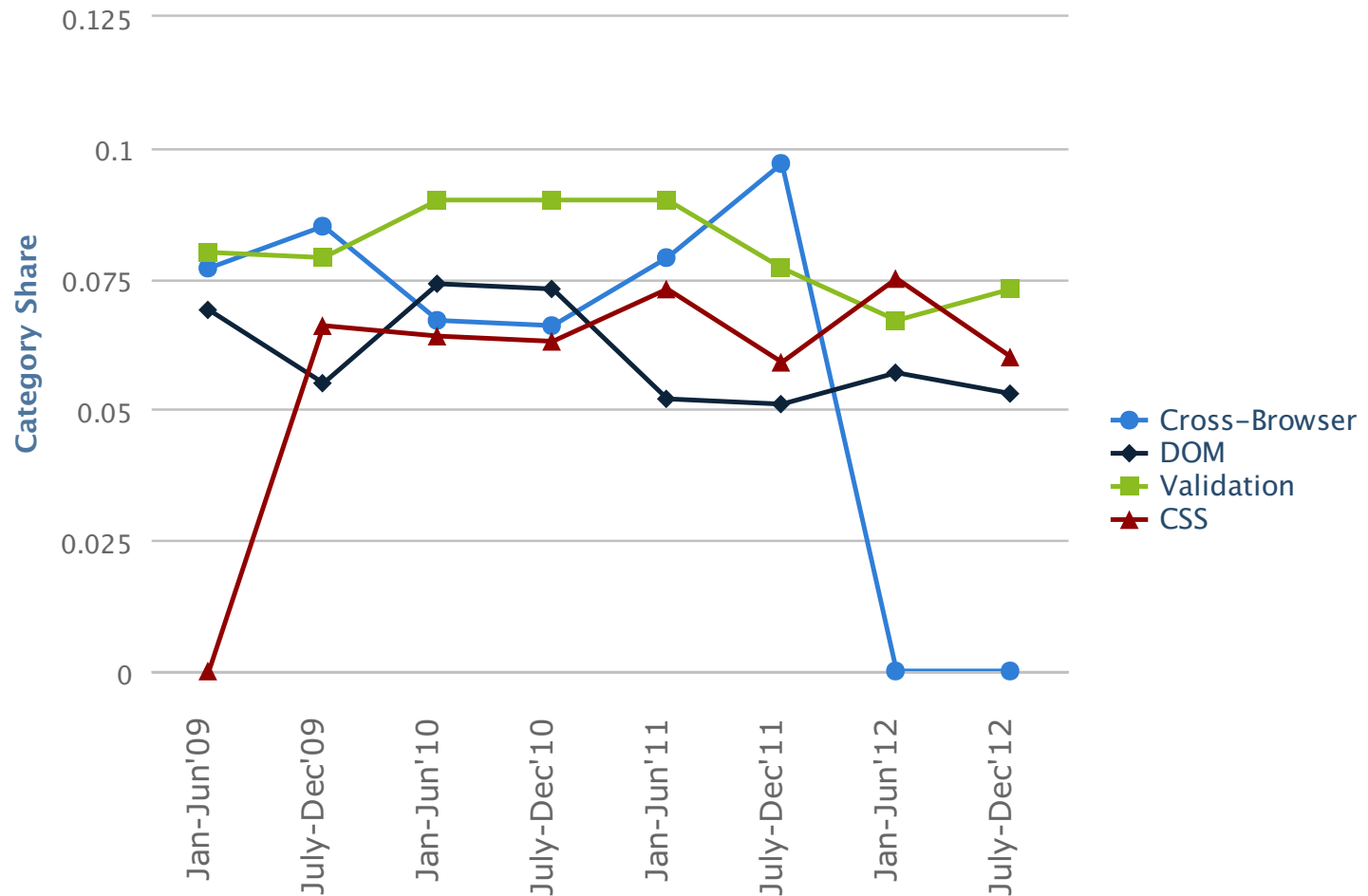
StackOverflow Datasets



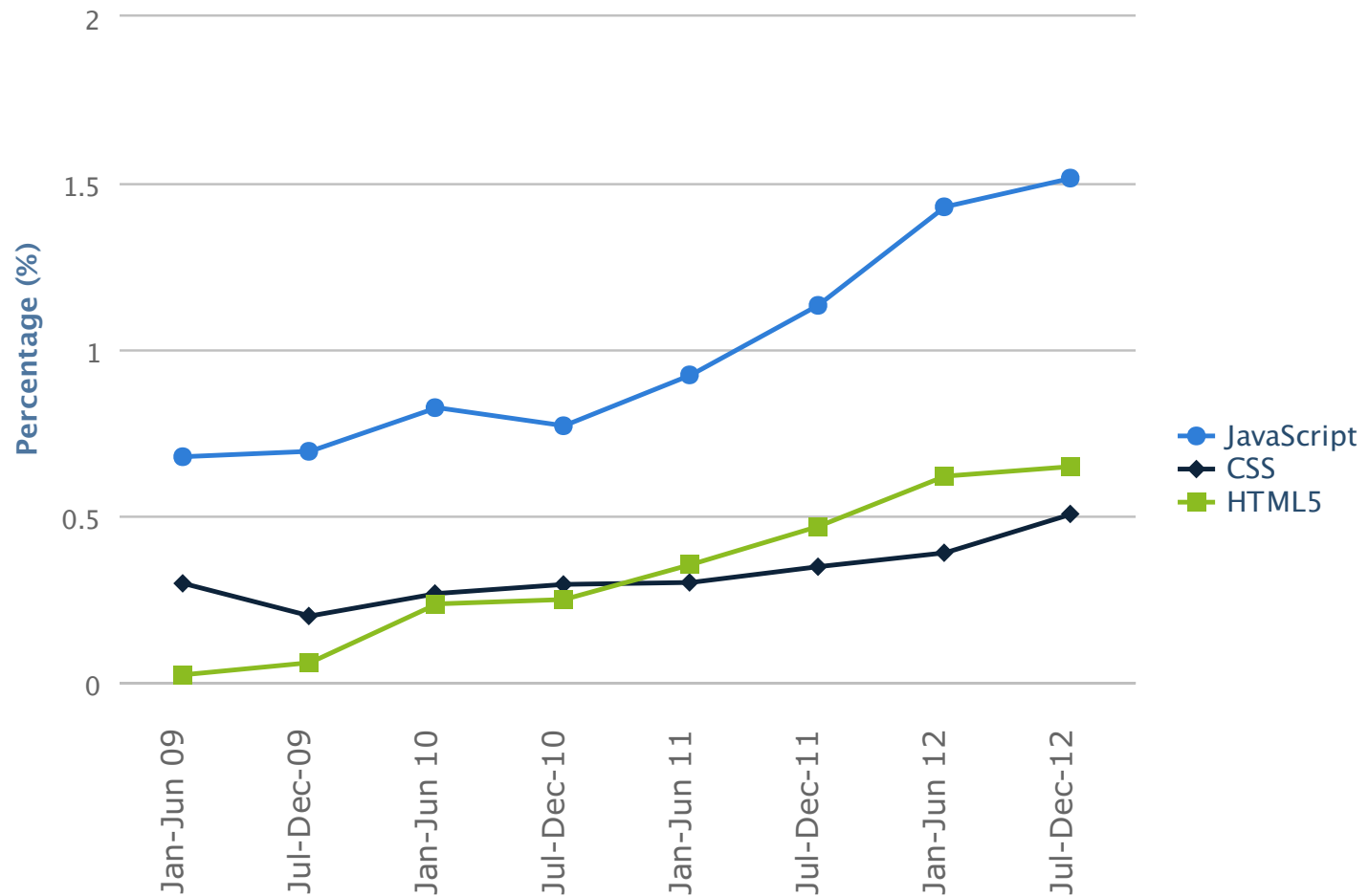
RQ1: Categorization of topics of discussion



RQ2: Temporal trends over time



RQ3: Prevalence of web in mobile development



RQ4: Technical challenges

I'd like to store a JavaScript object in HTML5 `localStorage`, but my object is apparently being converted to a string.

I can store and retrieve primitive JavaScript types and arrays using `localStorage`, but objects don't seem to work. Should they?

I have noticed I am getting a "CSS Explosion". It is becoming difficult for me to decide how to best organize and abstract data within the CSS file.

StackOverflow: Summary of Findings

- **Finding 1:** Though cross-browser issues dominated in the past, they have declined sharply since 2012.
- **Finding 2:** DOM and Canvas interactions consistently dominate
- **Finding 3:** Mobile web application development is on the rise, compared to traditional web appln. development
- **Finding 4:** Even expert programmers are confused by APIs and documentation

StackOverflow: Implications

- **Finding 1, 2 (Categorization and temporal trends)**
 - *Researchers* can shift their focus away from cross browser issues to DOM and Canvas related ones.
- **Finding 4 (Prevalence of mobile applications)**
 - Need better tools for mobile web development
- **Finding 5 (Technical Challenges)**
 - Can guide *standardization communities* to focus on areas that need improvement.

This Talk

- Motivation and Approach
- Three approaches for measuring JS Reliability
 - Error Messages [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - Bug Reports [ESEM 2013] – With F. Ocariza, K. Bajaj, A. Mesbah
 - Stack Overflow Reports [MSR 2014] – With K. Bajaj and A. Mesbah
- **Conclusion and Next Steps**

Conclusions

- **Web 2.0 applications' reliability is challenging**
- **Measure the reliability of web applications**
 - [ISSRE'11]: Based on error messages in real web apps
 - [ESEM'13]: Based on bug reports in web apps
 - [MSR'14]: Based on StackOverflow questions
- **Need to improve web applications' reliability –**
Use of empirical data to drive improvements

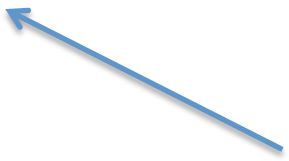
Next Steps (Part 2 of tutorial)

- AutoFlox (with Frolin Ocariza): ICST 2012
 - Fault localization
- Vejovis (with Frolin Ocariza): ICSE 2014
 - Fault Repair
- Clematis (with Saba Alimadi, Sheldon Sequeira): ICSE 2014
 - Program Understanding
- Dompletion (with Kartik Bajaj): ASE'14
 - Code completion

AutoFlox [Ocariza – ICST 2012]

- **AutoFlox**: Automatic fault localization tool for JS
 - Find origin of the null value
 - i.e., find the *direct DOM access*

```
1  var toggle = 1;
2  var x = "hlelo_";
3  var y = "world";
4  var elem = document.getElementById(x + y);
5  var dis = "";
6  if (toggle == 1) {
7      dis = "block";
8  }
9  else {
10     dis = "inline";
11 }
12 elem.style.display = dis;
```



Direct DOM Access
(This is where the NULL
value came from)

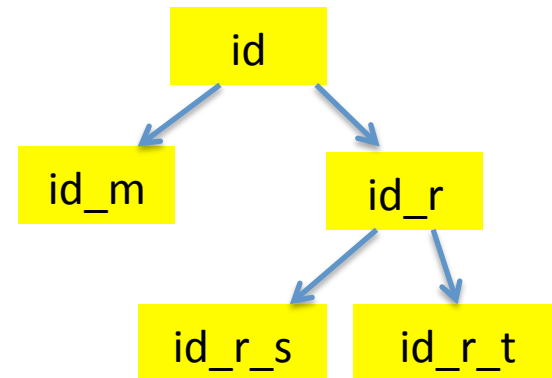
Vejovis [Ocariza – ICSE'14]

- **Vejovis**: automatic repair of DOM-related errors
 - Starts at direct DOM access found by AutoFlox
 - Provide fix suggestions based on common fix patterns

```
1  var toggle = 1;
2  var x = "hlelo_";
3  var y = "world";
4  var elem = document.getElementById(x + y);
5  var dis = "";
6  if (toggle == 1) {
7      dis = "block";
8  }
9  else {
10     dis = "inline";
11 }
12 elem.style.display = dis;
```

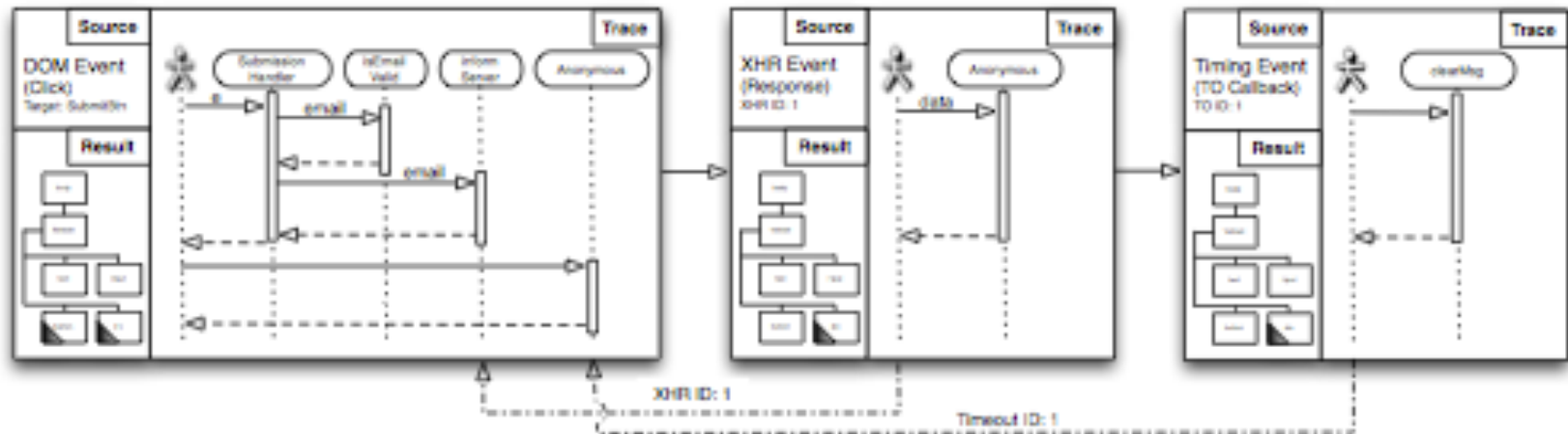
Parameter

Find “potential replacements” in DOM



Clematis [Alimadadi – ICSE'14]

- Challenge: Web applications are complex, and consist of DOM interactions, AJAX messages and timeouts
- Difficult to trace the links between events and JS code
- Clematis allows users to visualize causal dependencies between events and code, and the DOM



Dompletion [Bajaj - ASE'14]

- Provide code-completion suggestions for programmers for DOM-JavaScript interaction
 - Based on analysis of JavaScript code and DOM

```
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



Karthik Pattabiraman



Ali Mesbah



Kartik Bajaj



Frolin Ocariza



Saba Alimadadi



Sheldon Sequira

