

Why do Web Applications Fail and What can be done about it ?



Karthik Pattabiraman¹

Frolin Ocariza Jr.¹

Ali Mesbah¹

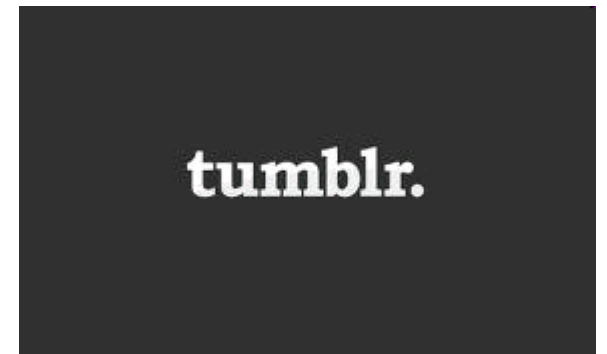
Benjamin Zorn²

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

My Research

- **How to build Reliable and Secure Software**
- **Software Error Resilience to Hardware Errors**
 - Error Detection in Multicore Systems [SELSE'12]
 - Fault diagnosis using software methods [SELSE'12]
- **Web 2.0 Applications' Reliability – This Talk**

Web 2.0 Applications



Web 2.0 Application: Amazon.com



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

Web 2.0 Application: JavaScript



The screenshot shows a web browser window with the address bar displaying a URL from Amazon's image service. The browser's developer tools are open, showing a large block of JavaScript code in the console. The code is part of a function named `S9MultiPackLayout.prototype.makeVisible`, which appears to be responsible for dynamically adjusting the visibility and width of items in a layout. The code includes several nested loops and conditional statements, indicating a complex logic for rendering a grid of items. The browser's status bar at the bottom shows buttons for 'Format code', 'Execute JS', and 'Error console', suggesting that the code is being executed in the browser's console.

```
109. }  
110. }  
111. }  
112. }  
113. S9MultiPackLayout.prototype.makeVisible = function() {  
114.     var numProposedVisibleItems = this.numProposedVisibleItems();  
115.     var lastVisibleCol = this.firstVisibleCol + numProposedVisibleItems - 1;  
116.     var width = ((100 / numProposedVisibleItems)-1);  
117.   
118.     if (this.seedItem) {  
119.         this.seedItem.style.width = width + "%";  
120.         this.itemChildren[0].style.display = "";  
121.         this.itemChildren[0].style.width = "100%";  
122.         this.otherItems.style.width = (98 - width) + "%";  
123.         var widthWithoutSeed = ((100 / (numProposedVisibleItems-1))-1);  
124.         for (var i = 1; i < this.itemChildren.length; i++) {  
125.             if ((i >= this.firstVisibleCol) && (i <= lastVisibleCol)) {  
126.                 this.itemChildren[i].style.display = "";  
127.                 this.itemChildren[i].style.width = widthWithoutSeed + "%";  
128.                 if (this.itemImages[i].getAttribute("url")) {  
129.                     this.itemImages[i].src = this.itemImages[i].getAttribute("url");  
130.                     this.itemImages[i].setAttribute("url", "");  
131.                 }  
132.             } else {  
133.                 this.itemChildren[i].style.display = "none";  
134.             }  
135.         }  
136.     } else {  
137.         for (var i = 0; i < this.itemChildren.length; i++) {  
138.             if ((i >= this.firstVisibleCol) && (i <= lastVisibleCol)) {  
139.                 this.itemChildren[i].style.display = "";  
140.                 this.itemChildren[i].style.width = width + "%";  
141.                 if (this.itemImages[i].getAttribute("url")) {  
142.                     this.itemImages[i].src = this.itemImages[i].getAttribute("url");  
143.                     this.itemImages[i].setAttribute("url", "");  
144.                 }  
145.             } else {  
146.                 this.itemChildren[i].style.display = "none";  
147.             }  
148.         }  
149.     }  
150. }
```

Significant amount of JavaScript code executing in the browser

Web 2.0 Application: Amazon.com



Web Apps experience errors, yet they continue to execute !

Web 2.0 Applications: Problems



Multiple Parties



Loose semantics



JavaScript

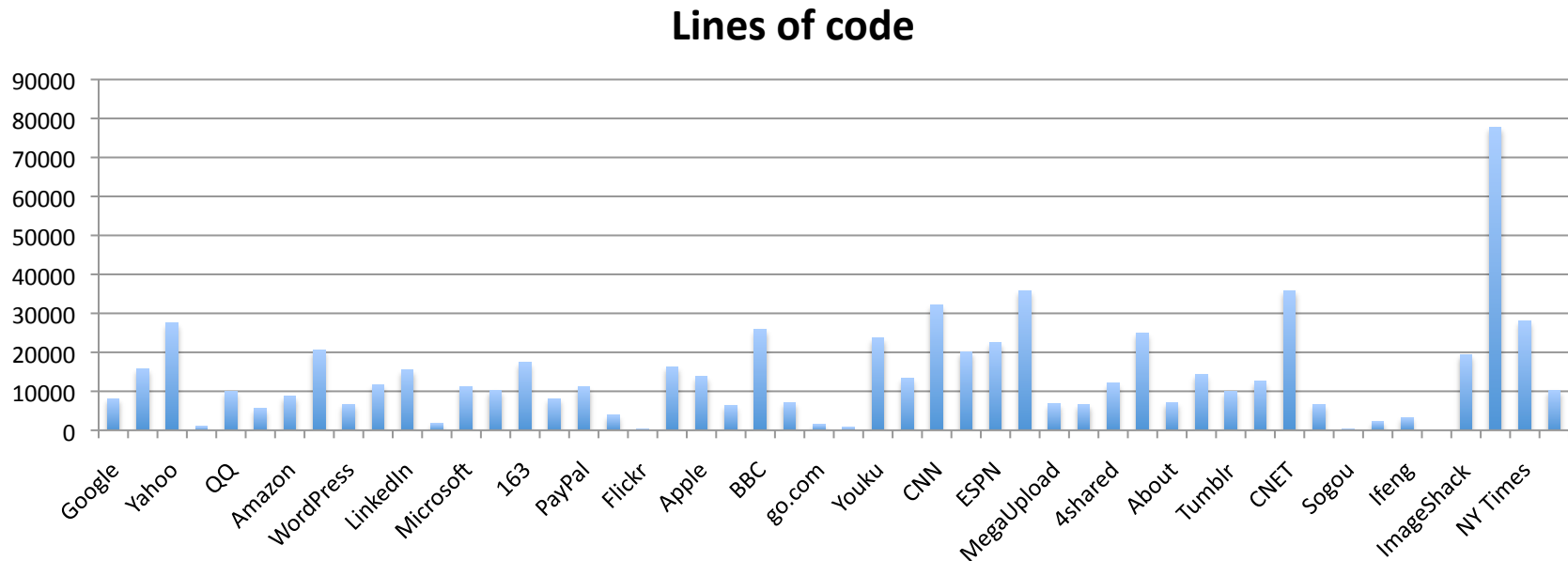


JavaScript (JS) had to “look like Java” only less so, be Java’s dumb kid brother or boy-hostage sidekick. Plus, I had to be done **in ten days** or something worse than JS would have happened

– Brendan Eich (Inventor of JavaScript)

Prevalence of JavaScript

- 97 of Alexa top 100 websites use JavaScript
- Many of them use “Evil” features [Richards-PLDI10]



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



王治郅闭幕式上挥舞国旗入场

2 of 3

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

Web Apps' Reliability: Challenges

- **Lack of specifications**
 - Automatically find reliability violations
- ▶ **Lack of tool support**
 - ▶ Automatically localize application errors

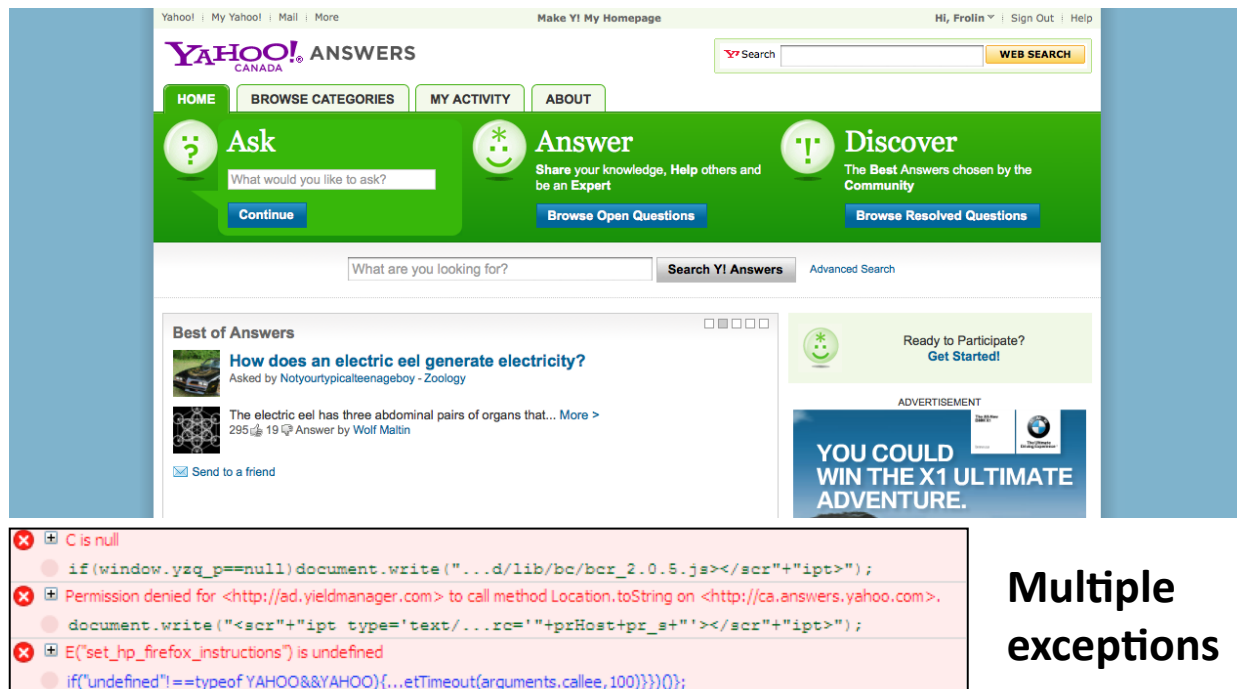


This Talk

- Motivation and Approach
- Two approaches for JS Reliability
 - JSER [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - AutoFlox [ICST 2012] – With F. Ocariza, A. Mesbah
- Future Directions and Conclusions

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
- Challenging to analyze JavaScript statically
- Capture interactions with the DOM



Vs.



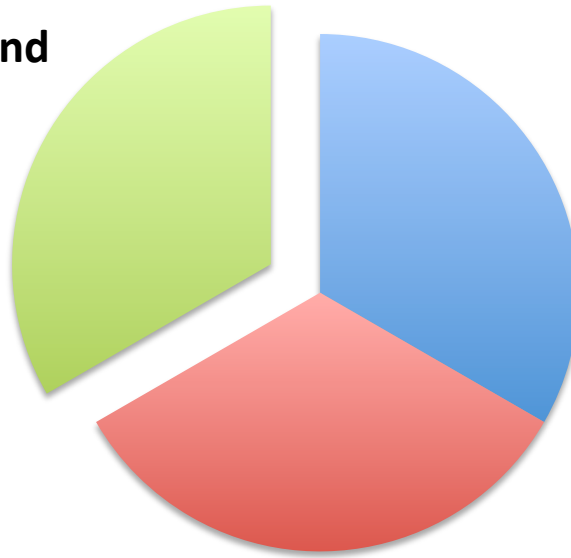
JSER: Tools

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



JSER: Research Questions

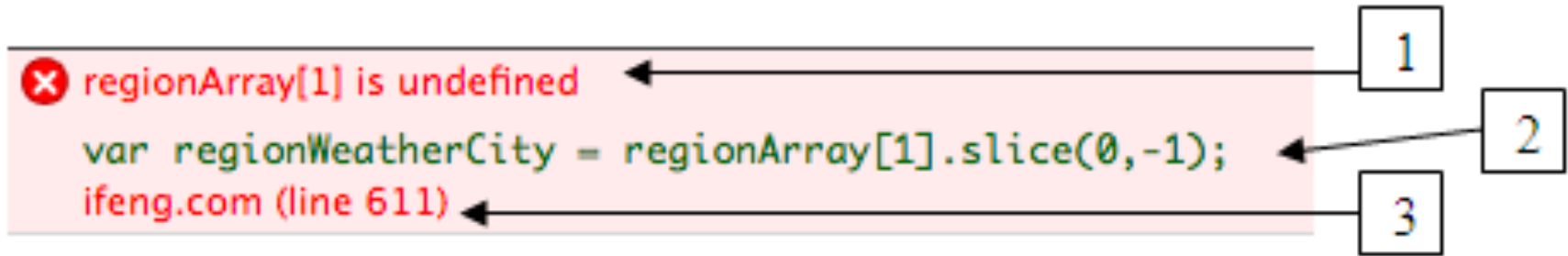
Do errors occur in web apps and if so, what categories do they fall in ?



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

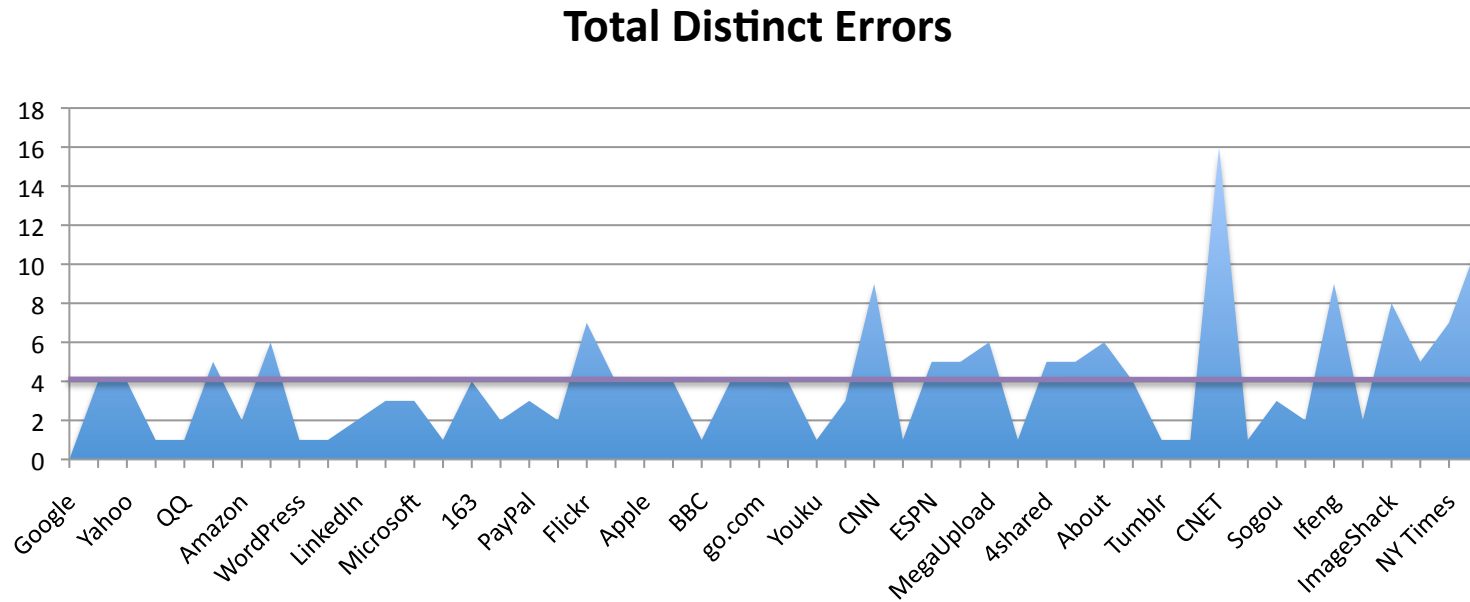


1. Description of error message
2. Line of code corresponding to error
3. Domain number and line number

Two errors are different if any attribute is different

JSER: Error Frequencies Results

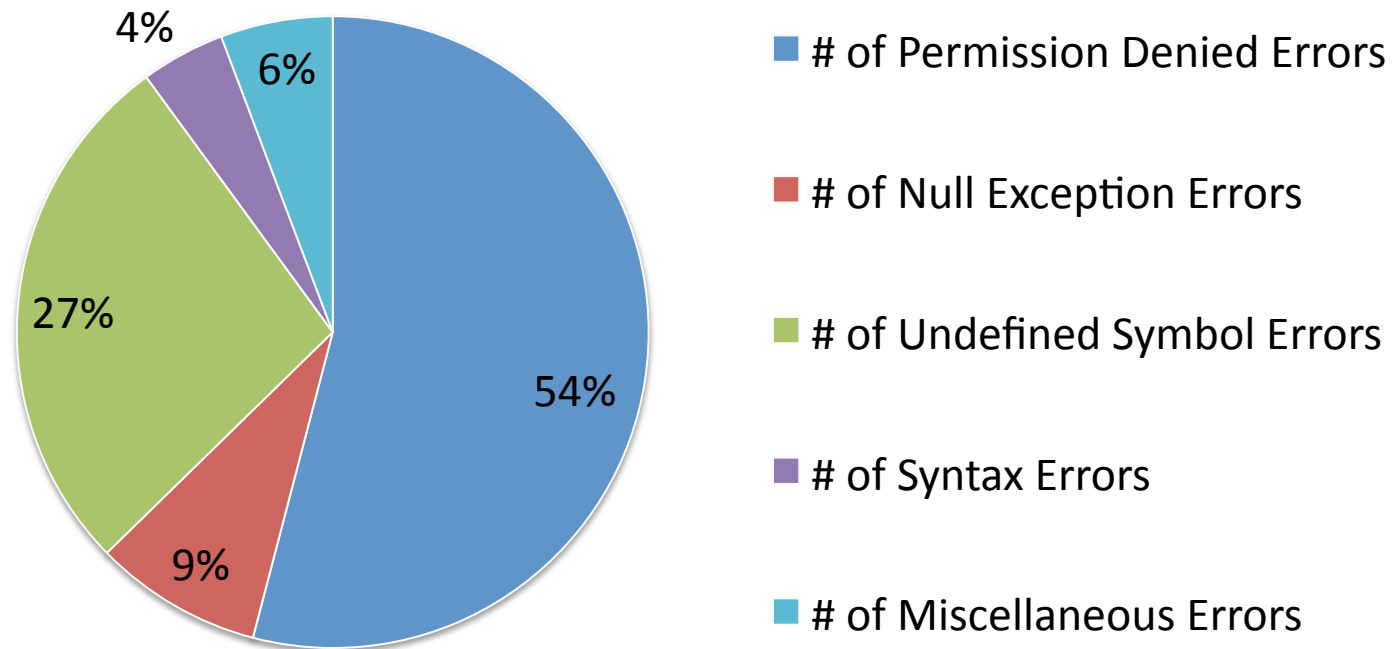
- **Average of 4 distinct error messages for each app**
 - **Standard dev: 3**
 - **Max: 16 (Cnet)**
 - **Min: 0 (Google)**



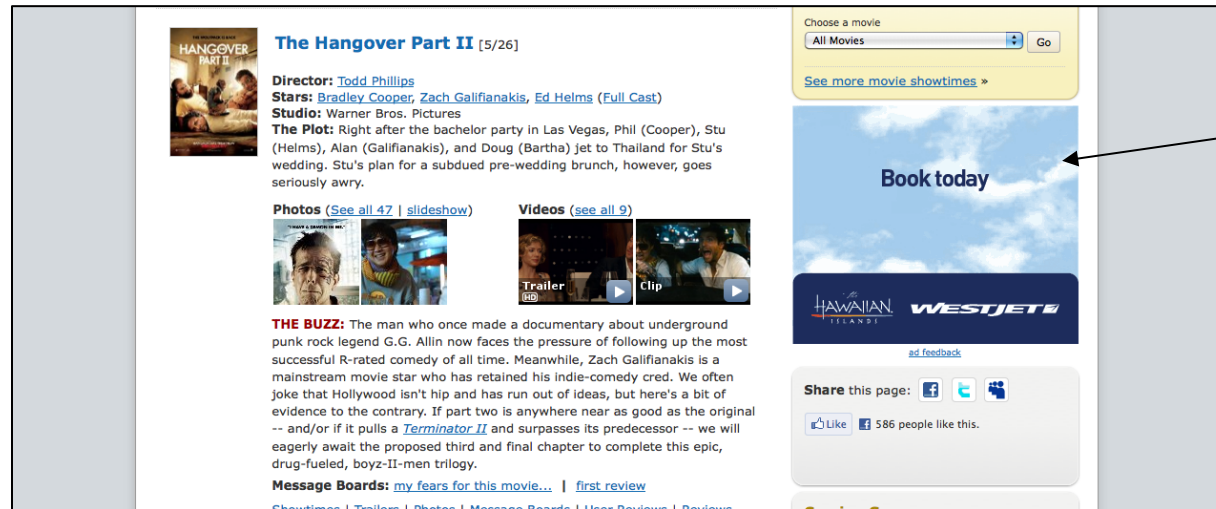
JSER: Error Classification Results

- 94 % of errors fall into four predominant categories

Distribution of Error Messages

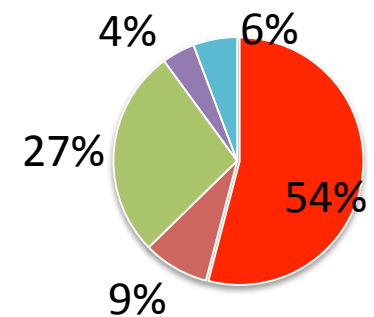


JSER: Permission Denied Example



Taken from
imdb.com

advertisement



- Error Message:** Permission denied for <http://view.atdmt.com> to call method `Location.toString` on <http://www.imdb.com>

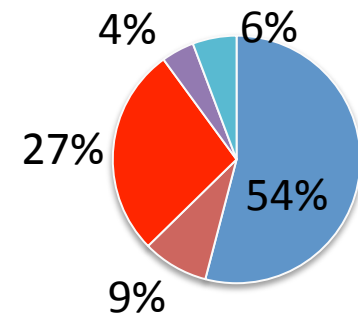
- Explanation:** Triggered by appearance of advertisement. Leads to SOP violation.

Bottom Line: JS errors may appear as a result of code written by others

JSER: Undefined Symbol Example



Taken from
cnn.com



- **Error Message:** cnn_onMemFBInit() is undefined

// this probably isn't needed anymore

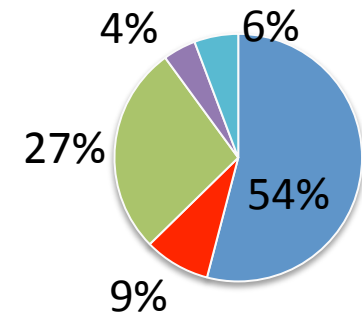
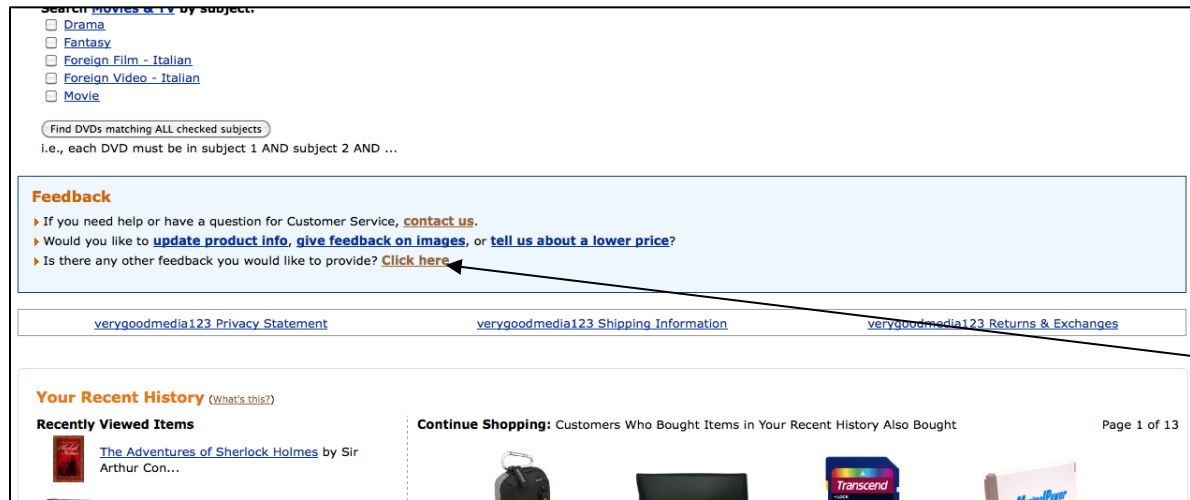
```
if (CNN_ISMemInit && CNN_IsFBInit) cnn_onMemFBInit();
```

- **Explanation:** Both CNN_IsMemInit and CNN_IsFBInit set to true

- **Bottom Line:** JS code is difficult to maintain

JSER: Null Exception Example

Taken from
amazon.com



Causes error
on click

- **Error Message:** `document.getElementById("inappDiv")` is null

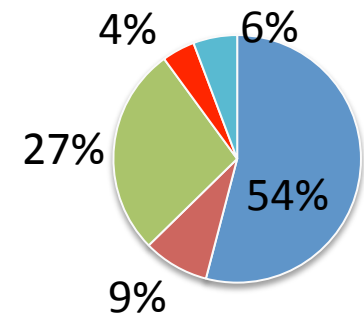
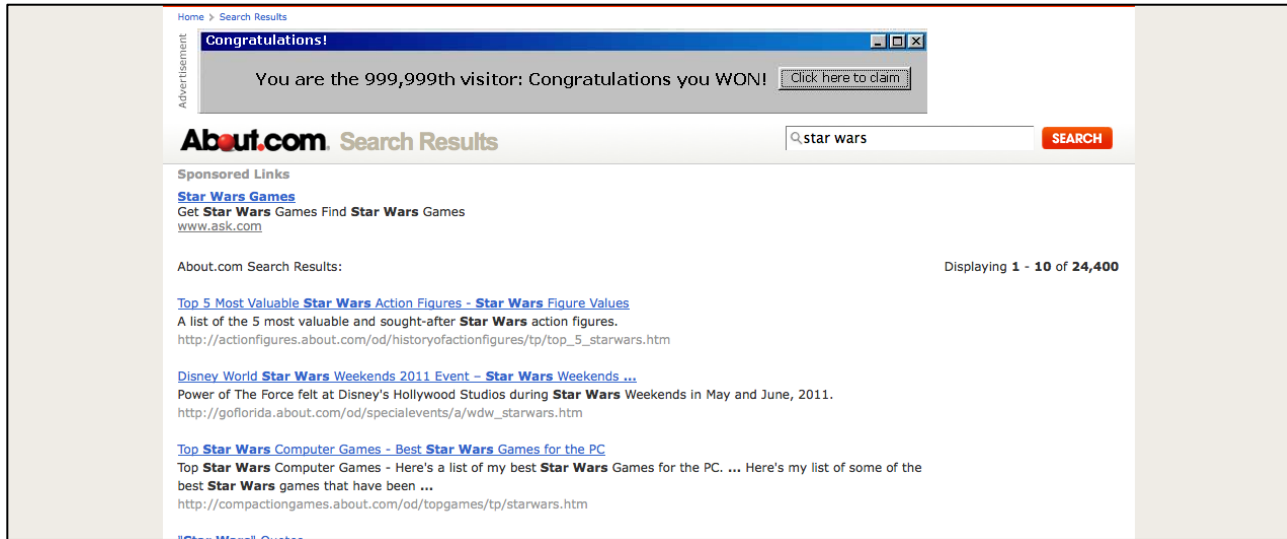
```
document.getElementById("inappDiv").style.display = 'none';
```

- **Explanation:** `inappDiv` was only defined for users who are logged in

- **Bottom Line:** JS code may depend on the DOM

JSER: Syntax Error Example

Taken from
about.com



- Error Message:** unterminated string literal

```
zGPU = 'http://movies.about.com/od/onlinemovies  
Movies_Available_on_the_Internet.html' "
```

- Bottom Line:** JS code is sometimes not well-tested

JSER: Research Questions

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

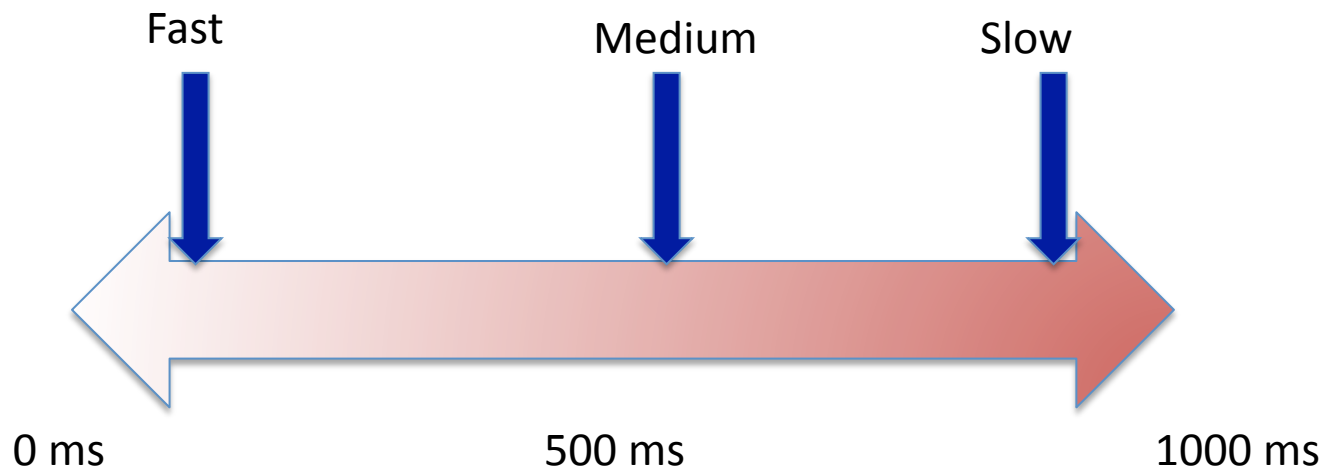


How do errors correlate with static and dynamic characteristics of the app?

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

JSER: Effect of Testing Speed

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

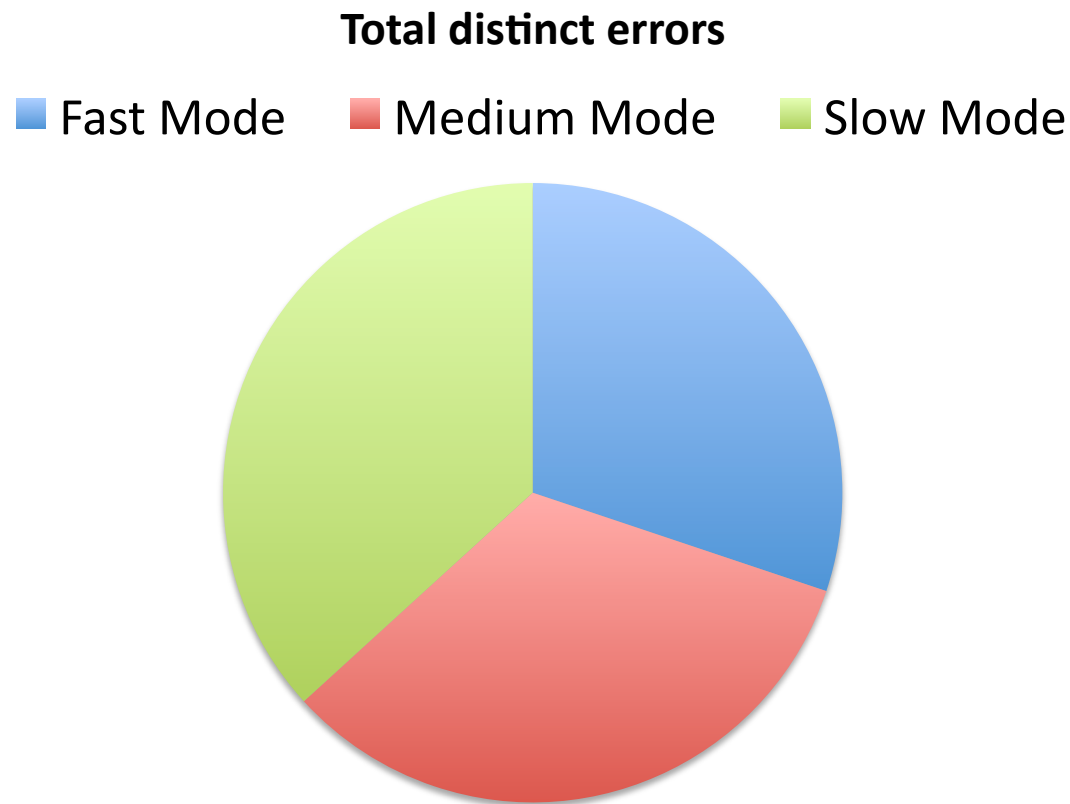


JSER: Testing Speed Results (CNN)

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

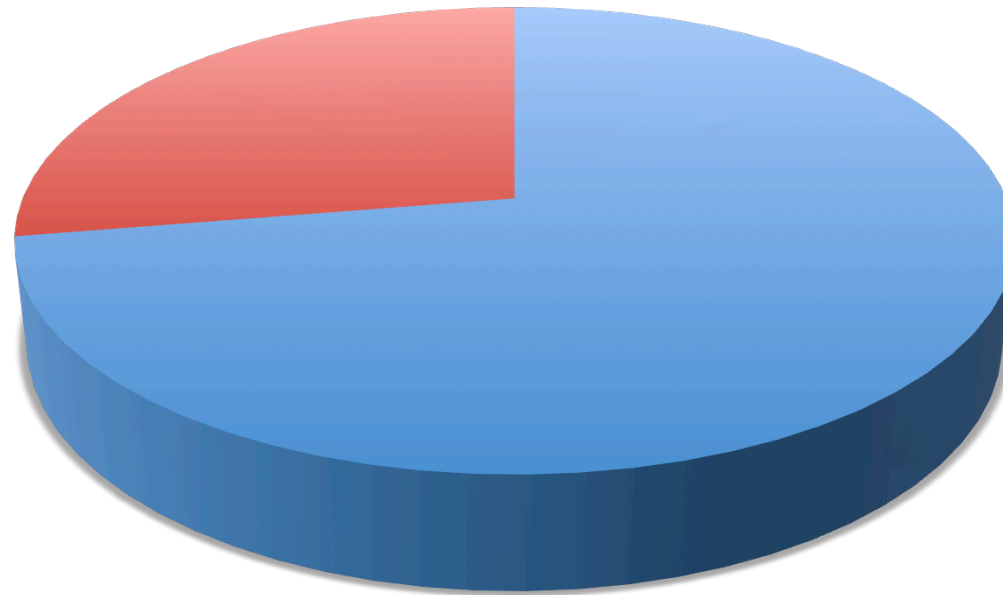
JSER: Effect of Testing Speed

- All three testing modes expose different errors



JSER: Non-Determinism

- More than 70% of errors are non-deterministic



■ Total non-deterministic errors

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: Static/Dynamic Characteristics

Static Characteristics

Measured using Phoenix & Firebug plugins

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

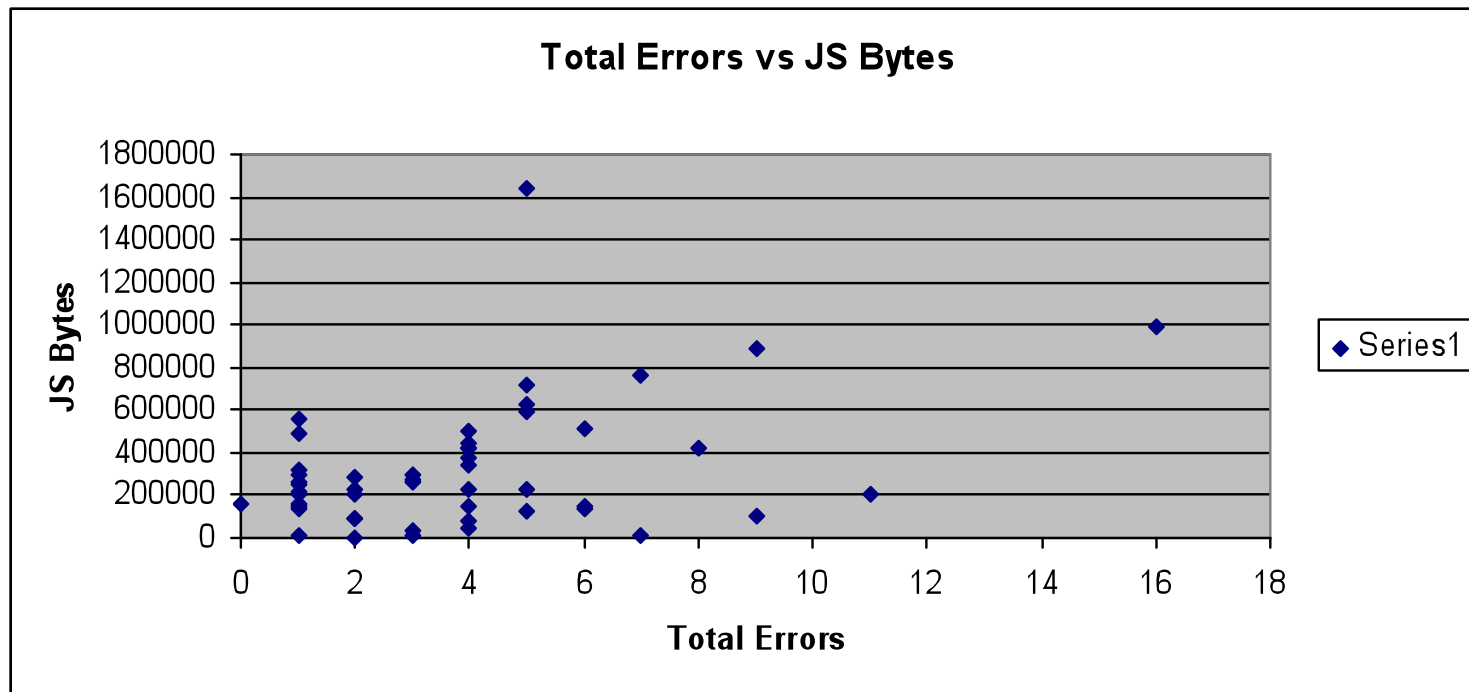
Dynamic Characteristics

From Richards et al. [PLDI – 2010]

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

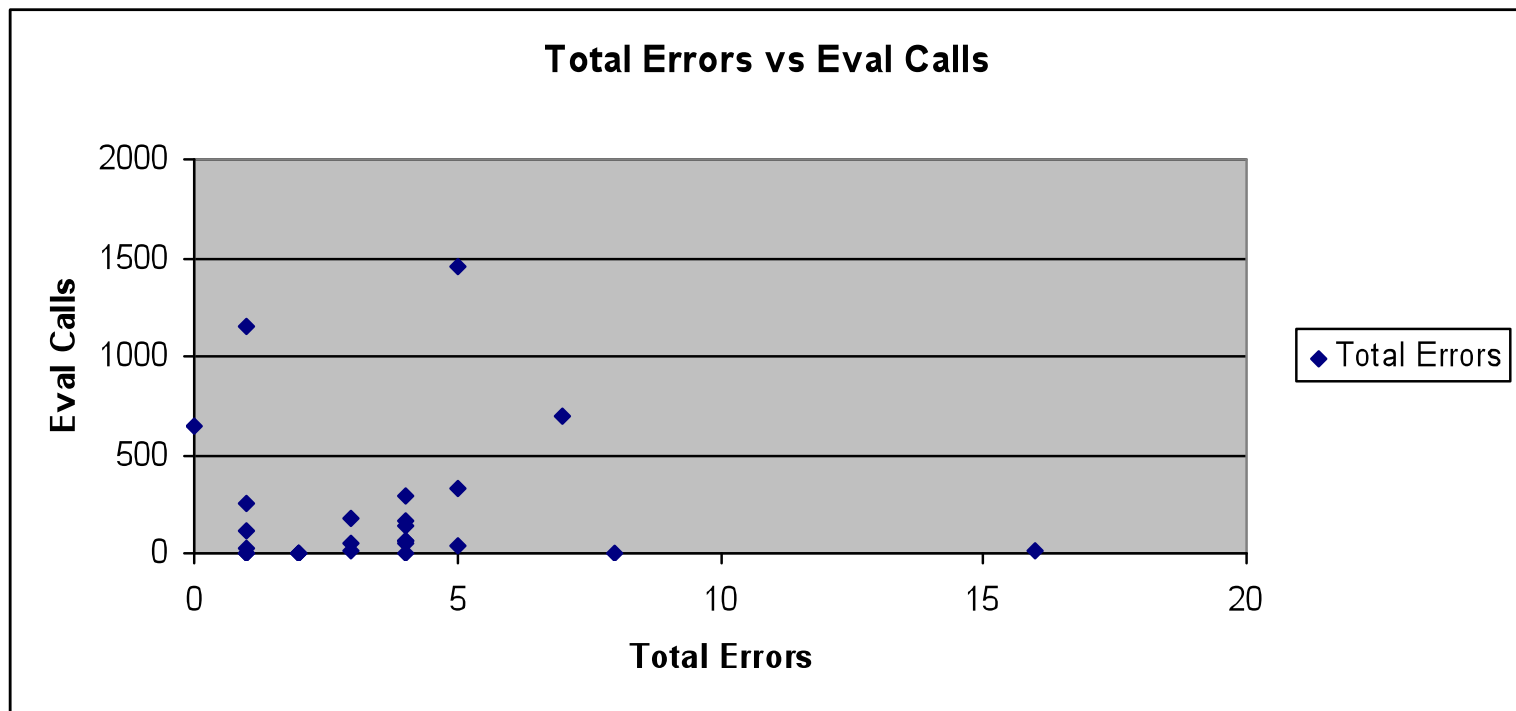
JSER: JS Code Size Correlations

- Low correlation
 - JS reliability not correlated with code size



JSER: Eval Calls Classification

- Low correlation
 - Eval calls do not seem to influence reliability



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

Average of four errors in each app. Errors fall into four well-defined categories



Correlated with no of late domains with JS, Alexa rank the app?

Errors vary by speed of testing. Majority 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
- Two approaches for JS Reliability
 - JSER [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - AutoFlox [ICST 2012] – With F. Ocariza, A. Mesbah
- Future Directions and Conclusions

AutoFlox: Motivation

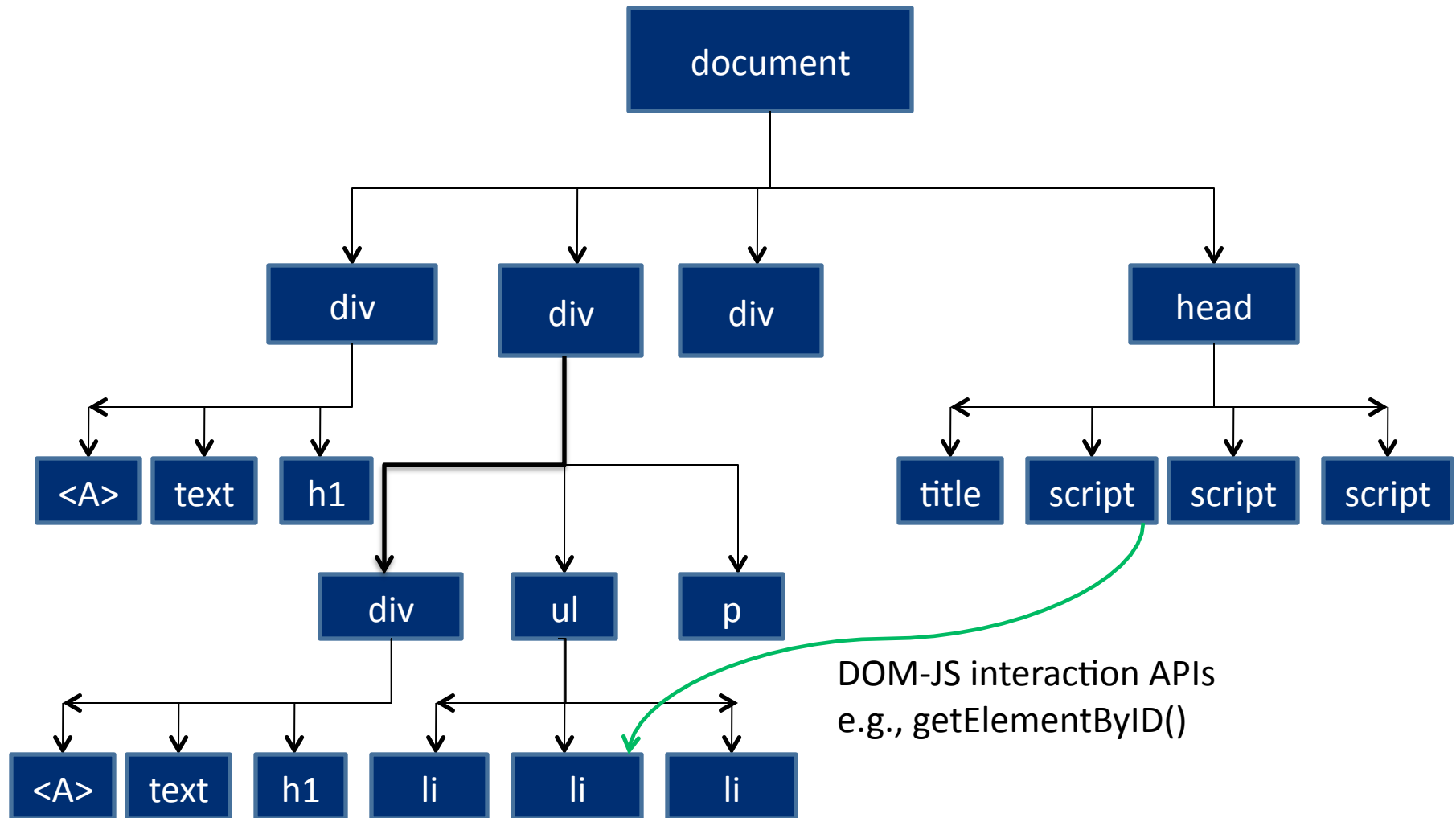
- **Goal: Find JavaScript line corresponding to error**



Why is this challenging ?

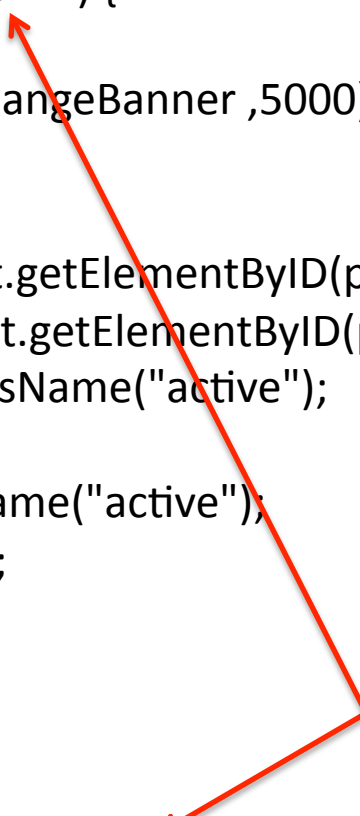
- Many errors occur due to DOM-JavaScript Interactions
- Lack of strict error checking leads to error propagation

AutoFlox: What is the DOM ?



AutoFlox: Example (Tumblr.com)

```
1 function changeBanner( bannerID ) {  
2   clearTimeout(changeTimer);  
3   changeTimer=setTimeout(changeBanner ,5000);  
4   ...  
5   prefix = "banner_";  
6   currBannerElem = document.getElementById(prefix + currentBannerID);  
7   bannerToChange = document.getElementById(prefix + bannerID);  
8   currBannerElem.removeClassName("active");  
  
9   bannerToChange.addClassName("active");  
10  currentBannerID = bannerID;  
}  
...  
currentBannerID = 1;  
  
changeTimer=setTimeout(changeBanner , 5000);
```



Missing
argument

AutoFlox: Example (Tumblr.com)

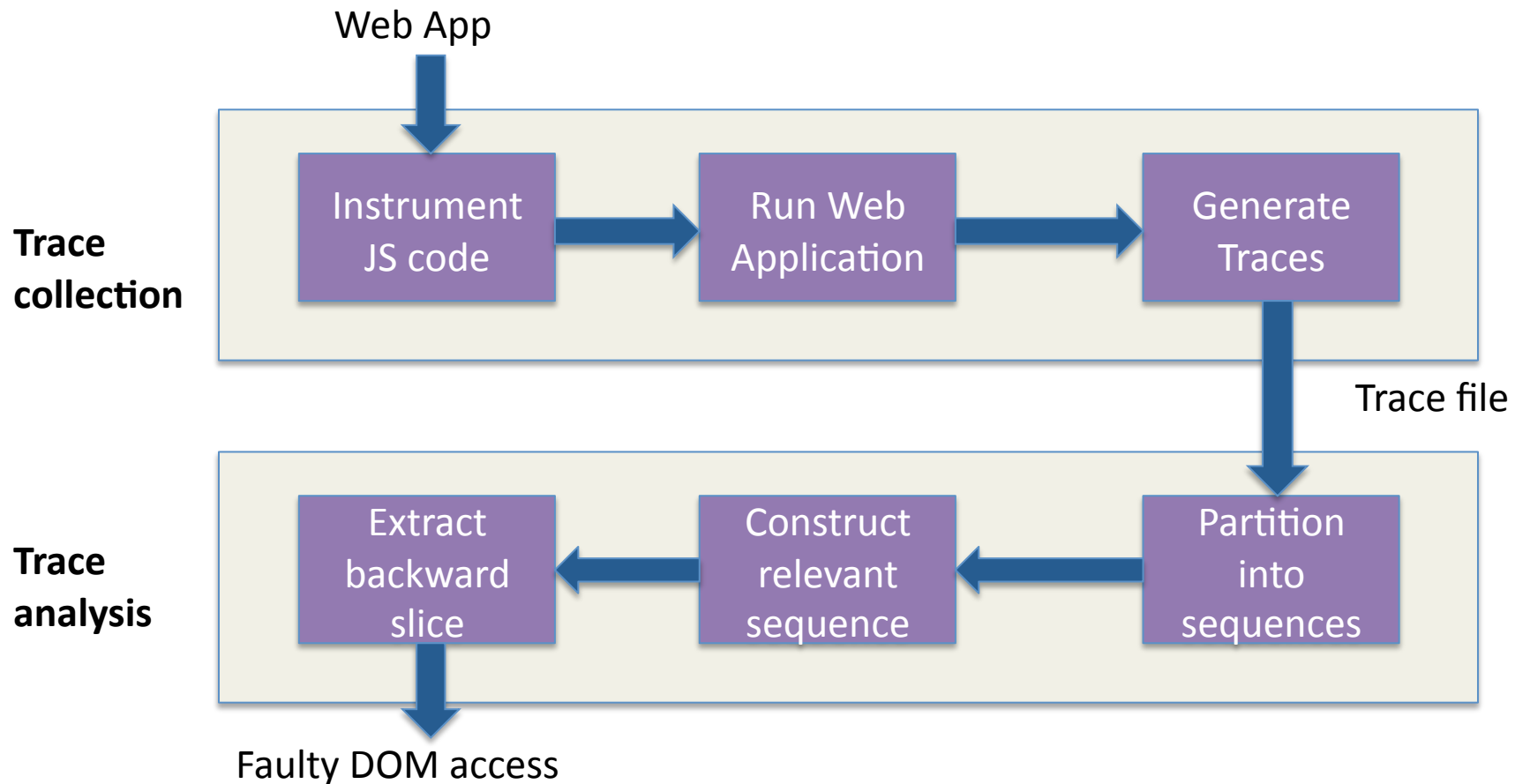
```
1 function changeBanner( bannerID ) {  
2   clearTimeout(changeTimer);  
3   changeTimer=setTimeout(changeBanner ,5000);  
4   ...  
5   prefix = "banner_";  
6   currBannerElem = document.getElementById(prefix + currentBannerID);  
7   bannerToChange = document.getElementById(prefix + bannerID);  
8   currBannerElem.removeClassName("active");  
  
9   bannerToChange.addClassName("active");  
10  currentBannerID = bannerID;  
}  
...  
currentBannerID = 1;  
  
changeTimer=setTimeout(changeBanner , 5000);
```

Faulty DOM access

Null Exception

AutoFloX: Approach

- Identify faulty DOM access through dynamic tracing
- Focus on errors due to **DOM-JavaScript** interactions



AutoFlox: Research Questions

- **RQ1: Are DOM-related JS errors a real problem ? How frequently are they reported in practice ?**
- **RQ2: Is AutoFlox effective at finding DOM-related JS errors in open-source applications ?**
- **RQ3: What is the performance overhead of AutoFlox ?**

AutoFlox: Experiment 1

- **RQ1: Are DOM-related JS errors a real problem ? How frequently are they reported in practice ?**

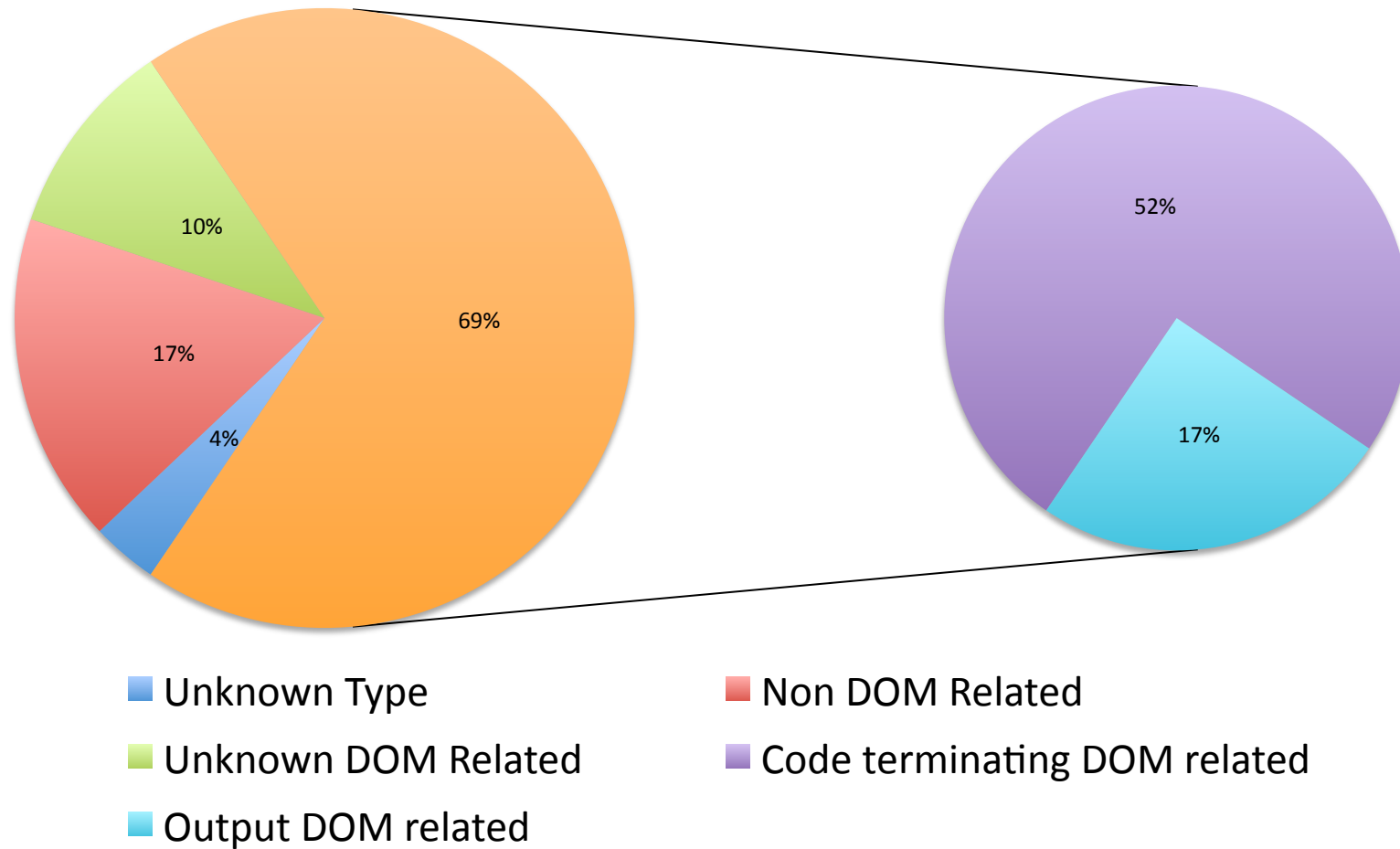
Studied bug reports of three open-source JavaScript apps and one closed-source JavaScript app (Google Docs)

- Filtered based on keywords such as “JavaScript”, “JS”, “Console”
- Only considered those that had been fixed and fix involved JS

Total of 29 bug reports satisfied this criteria from 135 reports

AutoFlox: Results RQ1

Breakdown of JavaScript Errors



AutoFlox: Experiment 2

- **RQ2: Is AutoFlox effective at finding DOM-related JS errors ?**

Benchmarks: Tudu, TaskFreak and WordPress

Experiment: Injected faults by mutating DOM accessor methods. Compared AutoFlox output with the injected line – match equals success

AutoFlox: Results RQ2

Name of Appln	No of lines of code	No of mutations	Detected no	Percentage of successes
TaskFreak	3044	29	29	100%
Tudu	11653	24	24	100%
Wordpress	8366	13	7	54%
Total		66	60	91%

AutoFlox diagnosed **all errors** in 2 of 3 applications

WordPress was the exception – couldn't diagnose errors in anonymous functions (future work)

Overall accuracy was about 91%

AutoFlox: Summary

- Fault localization for JavaScript is challenging
- About 80% of JavaScript bugs occur due to DOM interactions – 52% code-terminating
- AutoFlox uses dynamic backward slicing to successfully isolate > 90% of injected faults
 - Real error from tumblr.com localized

This Talk

- Motivation and Approach
- Two approaches for JS Reliability
 - JSER [ISSRE 2011] – With F. Ocariza and B.G. Zorn
 - AutoFlox [ICST 2012] – With F. Ocariza, A. Mesbah
- Future Directions and Conclusions

Future Directions

- **Static analysis to augment dynamic analysis**
 - Seed with errors found by dynamic analysis
 - Prioritize errors based on past experience
- **Analyze the impact of an error message**
 - Does it impact application's functionality ?
 - Does it affect other users of the application
- **Fault Injection**
 - Inject realistic faults in the application
 - Study its robustness under faults



Conclusions

- **Web 2.0 applications' reliability is challenging**
- **Measured reliability of web apps in the wild**
 - JSER: Based on error messages
 - AutoFlox: Fault localization
- **Need novel solutions to improve web apps reliability – the world is your playground**