Hybrid DOM-Sensitive Change Impact Analysis for JavaScript

Saba Alimadadi, Ali Mesbah and Karthik Pattabiraman
ECOOP 2015
saba@ece.ubc.ca
Change Impact Analysis (CIA)

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

JavaScript

DOM

Server
Challenge 1: Impact through the DOM

```javascript
function calculateTax() {
  $('.price').each(function(index) {
    $(this).text(addTaxToPrice(
      $(this).text()));
  });
}

function checkPrice() {
  . . .
  var cad_price = $('#price_ca').innerText();
  . . .
}
```
Challenge 2: Impact through Event Propagation

```javascript
function checkPrice() {
    ...
}

function calculateTax() {
    ...
}

$('#price_ca').bind('click', checkPrice);
$('prices').bind('click', calculateTax);
```
Challenge 3: Impact through XHRs

```javascript
function checkPrice() {
    var itemName = extractName($('item231').0);
    $.ajax({
        url: 'prices/latest.php',
        type: 'POST',
        data: itemName,
        success: eval(getAction() + 'item')
    });
}

function updateItem(xhr) {
    var updatedInfo = getUpdatedPrice(xhr.responseText);
    suggestItem.apply(this, updatedInfo);
}
```
Challenges of CIA for Client-Side JavaScript

1. JavaScript and Document Object Model (DOM)

2. Events and event propagation

3. JavaScript and XMLHttpRequests (XHRs)

+ High dynamism of JavaScript
Exploratory Study: DOM-related and Event-based Impacts

• Subject: 10 web apps (online contests, GitHub trending, etc.)

• Gathered data:
  – JavaScript-DOM interactions (write-read pairs)
  – Event propagation

• Further analysis of the structure of graphs.
  – Measured metrics: fan-in and fan-out of functions and DOM elements, and average path lengths
Exploratory Study: Results

- **W-R DOM elements:** 42%
- **Propagated handlers:** 14%

**DOM Elements**

- 58%
- 42% W-R Elements

**Event Handlers**

- 14% Propagated
- 86%

foo() → elem → bar()
Hybrid Analysis

- Static Analysis
- Dynamic Analysis
- Hybrid Model
- Ranked Impact Sets
Static Analysis

- Static Analysis
- Dynamic Analysis
- Hybrid Model
- Ranked Impact Sets

Control (and data) dependencies
Partial data-flow analysis
Example: Static Model

- Function
- XHR object
- DOM element
- Directed and labeled edge
Dynamic Analysis

Impact through DOM

Impact through events

Dynamic call graph

XHR relations (open, send, response)

JavaScript dynamism (eval(), function variadicity)
Example: Dynamic Model

- **Function**
- **XHR object**
- **DOM element**
- **Directed and labeled edge**

```
checkPrice()
```

```
updateItem()
```

```
suggestItem()
```

```
getUpdatedPrice()
```

```
calculateTax()
```

```
XHR
```

```
#item231
```

```
#price-ca .price
```
Static Analysis

Dynamic Analysis

Hybrid Model

Ranked Impact Sets
Example: Hybrid Analysis

Vertices
- Function
- XHR object
- DOM element
- Directed and labeled edge

Edges
- checkPrice()
- #item231
- XHR
- #price-ca .price
- getUpdatedPrice()
- updateItem()
- calculateTax()
- displaySuggestion()
- suggestItem()
- addTaxToPrice()
Static Analysis

Dynamic Analysis

Hybrid Model

Ranked Impact Sets
Impact Metrics and Impact Set Ranking

• Problem: size of impact sets
• Solutions: impact ranks, based on impact metrics
  – $f_{in}(d)$: Number of functions $f$ such that $f W d$
  – $f_{in}(f)$: Number of elements $d$ such that $f R d$
  – $f_{out}(f)$: Number of elements $d$ such that $f W d$
  – $L_{avg}(P)$: Average length of impact paths in the app
  – $D_m(e)$: Minimum distance of $e$ from the change set
  – $IR_{pr}(e)$: Impact of previous entity
Tool Implementation: Tochal

• Tochal: open source
  – https://github.com/saltlab/tochal

• Proxy (Java, JavaScript)
  – Esprima, Estraverse, Escodegen, Mutation Summary, WALA

• Client-side (Google Chrome extension)
  – Chrome DevTools
Research Question 1

Does Tochal outperform static and dynamic analysis methods in terms of the completeness of the results obtained?
Study: Static vs. Dynamic vs. Tochal

- 10 web applications
- 3 random functions as change sets
- Comparing:
  - Size of impact sets
  - Number of functions in dependency graphs
Results: Impact Sets

• Comparing size of impact sets

<table>
<thead>
<tr>
<th></th>
<th>Static Hybrid</th>
<th>Dynamic Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>26%</td>
<td>80%</td>
</tr>
</tbody>
</table>

![Bar chart showing comparison between Static, Dynamic, and Tochal for Static and Dynamic Hybrid systems.](chart.png)
Results: Graphs

• Comparing size of model graphs

<table>
<thead>
<tr>
<th>Static Hybrid</th>
<th>59%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic Hybrid</td>
<td>84%</td>
</tr>
</tbody>
</table>
Results: Graphs

• Comparing size of model graphs

Pure Static Hybrid: 15%

Pure Dynamic Hybrid: 42%
Research Question 2

Does Tochal help developers in practice to perform change impact analysis?
Experiment: Design

• 12 participants from industry

• Performed 4 tasks

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

• We measured task completion duration and accuracy
**User Experiment: Results**

**Duration**

- **Tochal**
  - T1: 300
  - T2: 40
  - T3: 60
  - T4: 80
  - Total: 480

- **Other**
  - T1: 30
  - T2: 40
  - T3: 60
  - T4: 80
  - Total: 210

**Accuracy**

- **Tochal**
  - T1: 100
  - T2: 80
  - T3: 60
  - T4: 80
  - Total: 320

- **Other**
  - T1: 20
  - T2: 40
  - T3: 60
  - T4: 80
  - Total: 180

**80% faster**

**2 times more accurate**
Results: Ranking

Duration

Accuracy

2~3 times faster

25% more accurate
Challenges of CIA for Client-Side JavaScript

1. JavaScript and Document Object Model (DOM)
2. Events and event propagation
3. JavaScript and XMLHttpRequests (XHRs)
   + High dynamism of JavaScript

Exploratory Study: Results

- W-R DOM elements: 42%
- Propagated handlers: 14%

Example: Hybrid Analysis

User Experiment: Results

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>2 times more accurate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration</td>
<td>80% faster</td>
</tr>
</tbody>
</table>

DOM Elements

Event Handlers

foo() → elem → bar()