Integrated Impact Analysis for Managing Software Changes

Malcom Gethers, Bogdan Dit, Huzefa Kagdi, Denys Poshvyvanyk
Software change impact analysis aims at estimating the potentially impacted entities of a system due to a proposed change [Bohner and Arnold ‘96]
Impact Analysis (IA)
Impact Analysis (IA)
Impact Analysis (IA)

Impact Set

[Code snippets and images related to Impact Analysis and Bugzilla]
Related Work

Software change IA via Static and Dynamic Analysis
[Law and Rothermel ICSE'03] [Orso et al. ESEC/FSE'03] [Orso et al. ICSE'04] [Ren et al. OOPSLA'04]

Software change IA via Information Retrieval
[Canfora and Cerula MSR'06] [Hill et al. ASE'07] [Kagdi et al. WCRE'10] [Poshyvanyk et al. EMSE'09]

Software change IA via Mining Software Repositories
[Zimmermann et al. TSE'05] [Gall et al. ICSM'98] [Kagdi et al. WCRE'10] [Canfora and Cerula MSR'06]
Related Work

Software change IA via Static and Dynamic Analysis

[Law and Rothermel ICSE'03] [Orso et al. ESEC/FSE'03] [Orso et al. ICSE'04] [Ren et al. OOPSLA'04]
Related Work

Both relate to “corpus” processing

Software change IA via Information Retrieval

[Canfora and Cerula MSR’06] [Hill et al. ASE’07] [Kagdi et al. WCRE’10] [Poshyvanyk et al. EMSE’09]
Related Work

Implement functionality related to “request body handling”

Software change IA via Mining Software Repositories

[Zimmermann et al. TSE’05] [Gall et al. ICSM’98] [Kagdi et al. WCRE’10] [Canfora and Cerula MSR’06]
Research Goal

Develop a novel Impact Analysis (IA) approach that adapts to specific maintenance scenarios:

- Information Retrieval (IR), Latent Semantic Indexing (LSI)
- Mining Software Repositories (MSR), Itemset Mining
- Run time information, execution traces
Research Goal

Develop a novel Impact Analysis (IA) approach that adapts to specific maintenance scenarios.

How can we integrate these information sources for IA?

- Information Retrieval (IR), Latent Semantic Indexing (LSI)
- Mining Software Repositories (MSR), Itemset Mining
- Run time information, execution traces
Scenario-Driven Approach

Default: Given a textual change request, an Information Retrieval (IR) (e.g., Latent Semantic Indexing) is used to estimate the impact set.
Scenario-Driven Approach

**Default:** Given a textual change request, an Information Retrieval (IR) (e.g., Latent Semantic Indexing) is used to estimate the impact set.

**Scenario 1:**
User provides a verified starting entity
Scenario-Driven Approach

**Default:** Given a textual change request, an Information Retrieval (IR) (e.g., Latent Semantic Indexing) is used to estimate the impact set.

**Scenario 2:**

User provides an execution trace

```
readAndDispatch --> org.eclipse.swt.widgets.Display
checkDevice    --> org.eclipse.swt.widgets.Display
isDisposed     --> org.eclipse.swt.graphics.Device
drawMenuBars   --> org.eclipse.swt.widgets.Display
unPopups       --> org.eclipse.swt.widgets.Display
filterMessage  --> org.eclipse.swt.widgets.Display
windowProc     --> org.eclipse.swt.widgets.Display
windowProc     --> org.eclipse.swt.widgets.Control
WM_TIMER       --> org.eclipse.swt.widgets.Control
windowProc     --> org.eclipse.swt.widgets.Display
windowProc     --> org.eclipse.swt.widgets.Control
WM_TIMER       --> org.eclipse.swt.widgets.Control
windowProc     --> org.eclipse.swt.widgets.Display
windowProc     --> org.eclipse.swt.widgets.Control
```
Scenario-Driven Approach

Default: Given a textual change request, an Information Retrieval (IR) (e.g., Latent Semantic Indexing) is used to estimate the impact set.

Scenario 3:
Both verified starting entity and run-time information are available
Approach (IR)

IR (default): Analyzing textual information via IR
Approach (IR)

IR (default): Analyzing textual information via IR
Approach (IR+Dyn)

IR+Dyn: Analyzing textual information via IR and execution information via dynamic analysis
**Approach (IR+Dyn)**

**IR+Dyn:** Analyzing textual information via IR and execution information via dynamic analysis
Approach (IR+Dyn+Hist)

IR+Dyn+Hist: Analyzing textual information via IR, execution information via dynamic analysis, and history via data mining.
Approach (IR+Hist)

**IR+Hist**: Analyzing textual information via IR and history via data mining
Approach (IR+Hist)

IR+Hist: Analyzing textual information via IR and history via data mining
Approach (IR+Hist)

IR+Hist: Analyzing textual information via IR and history via data mining
ArgouML Bug #2472: “Wrong keyboard focus in settings dialog after close & reopen […]”

- 16 methods impacted by bug #2472

Impact Analysis - Motivating Example

• IR (default): 2, 16, 30, ...

• IR+Dyn 1, 3, 5, 7, 11, 12, 14, 29, ...
Impact Analysis - Case Study

Four open source software systems, namely ArgoUML, JabRef, jEdit and muCommander.

Commits in the version history used as ground truth, i.e., actual impact sets.

Widely used metrics precision (false positives), recall (false negatives), and f-measure (combination of precision and recall) used for accuracy measure.

<table>
<thead>
<tr>
<th>System</th>
<th>Version</th>
<th>LOC</th>
<th>Files</th>
<th>Methods</th>
<th>Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgoUML</td>
<td>0.22</td>
<td>148K</td>
<td>503</td>
<td>6K</td>
<td>4K</td>
</tr>
<tr>
<td>JabRef</td>
<td>2.6</td>
<td>74K</td>
<td>1,439</td>
<td>11K</td>
<td>5K</td>
</tr>
<tr>
<td>jEdit</td>
<td>4.3</td>
<td>103K</td>
<td>1,069</td>
<td>8K</td>
<td>4K</td>
</tr>
<tr>
<td>muCommander</td>
<td>0.8.5</td>
<td>76K</td>
<td>577</td>
<td>4K</td>
<td>5K</td>
</tr>
</tbody>
</table>
Impact Analysis - Research Question

Does providing more information improve the accuracy of IA when compared to the default approach?
Impact Analysis - Main Results

Is it possible to improve the accuracy?

Precision

IR    IR+Dyn   IR+Hist   IR+Hist+Dyn
CP5   CP10    CP20      CP30    CP40
Impact Analysis - Main Results

Is it possible to improve the accuracy?

- CP5
- CP10
- CP20
- CP30
- CP40

Precision

0% 13% 25% 38% 50%
Impact Analysis - Main Results

The best results are obtained in scenarios where both a verified starting entities and execution information are available.

Recall improvements of up to 41% over the default approach and up to 17% over the evolutionary technique in iBatis.
Impact Analysis - Main Results

The best results are obtained in scenarios where both a verified starting entities and execution information are available.

Recall improvements of up to 41% over the default approach and up to 17% over the evolutionary technique in iBatis.
Impact Analysis - Statistical Test

Does providing more information significantly improve results of impact analysis compared to using the default approach?
## Impact Analysis - Results of Wilcoxon Signed-Rank Test

<table>
<thead>
<tr>
<th>System</th>
<th>$\mathcal{H}_0, P_1$</th>
<th>$\mathcal{H}_0, R_1$</th>
<th>$\mathcal{H}_0, P_2$</th>
<th>$\mathcal{H}_0, R_2$</th>
<th>$\mathcal{H}_0, P_3$</th>
<th>$\mathcal{H}_0, R_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgouML</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
</tr>
<tr>
<td>JabRef</td>
<td>0.266</td>
<td>0.324</td>
<td>0.381</td>
<td>$&lt; 0.0001$</td>
<td>0.091</td>
<td>$&lt; 0.0001$</td>
</tr>
<tr>
<td>jEdit</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>0.068</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
</tr>
<tr>
<td>muCommander</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>0.425</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
<td>$&lt; 0.0001$</td>
</tr>
</tbody>
</table>

$\mathcal{H}_0\, P_1$: IR+Dyn does not significantly improve precision (compared to IR)

$\mathcal{H}_0\, R_1$: IR+Dyn does not significantly improve recall (compared to IR)

$\mathcal{H}_0\, P_2$: IR+Hist does not significantly improve precision (compared to IR)

$\mathcal{H}_0\, R_2$: IR+Hist does not significantly improve recall (compared to IR)

$\mathcal{H}_0\, P_3$: IR+Dyn+Hist does not significantly improve precision (compared to IR)

$\mathcal{H}_0\, R_3$: IR+Dyn+Hist does not significantly improve recall (compared to IR)
## Impact Analysis - Results of Wilcoxon Signed-Rank Test

<table>
<thead>
<tr>
<th>Software</th>
<th>$H_0 P_1$</th>
<th>$H_0 P_2$</th>
<th>$H_0 R_2$</th>
<th>$H_0 P_3$</th>
<th>$H_0 R_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgoUML</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>JabRef</td>
<td>0.266</td>
<td>0.324</td>
<td>0.381</td>
<td>&lt; 0.0001</td>
<td>0.091</td>
</tr>
<tr>
<td>jEdit</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>muCommander</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>&lt; 0.0001</td>
<td>1.01</td>
</tr>
</tbody>
</table>

**Results are statistically significant!**

$H_0 P_1$: IR+Dyn does not significantly improve precision (compared to IR)

$H_0 R_1$: IR+Dyn does not significantly improve recall (compared to IR)

$H_0 P_2$: IR+Hist does not significantly improve precision (compared to IR)

$H_0 R_2$: IR+Hist does not significantly improve recall (compared to IR)

$H_0 P_3$: IR+Dyn+Hist does not significantly improve precision (compared to IR)

$H_0 R_3$: IR+Dyn+Hist does not significantly improve recall (compared to IR)
Threats to Validity

Commits used as gold standard for accuracy computation

- Not all the entities in a commit maybe related to a single change request
- All the entities related to a single change request maybe present in a single commit
- Developer established actual change-sets

Quality of dynamic traces obtained for change request

Statistically significant results for the four open source Java systems may not generalize
Scenario-driven combination of IR, dynamic analysis and MSR techniques for IA

Recall improvements of up to 41% over the default approach

Precision improvement up to 17% over the default approach
Thank You

SEMERU @ William and Mary
http://www.cs.wm.edu/semeru