Automatically Documenting Program Changes

Ray Buse
Wes Weimer

13 May 2011
13th CREST Open Workshop
Automated Software Engineering '10
Automatically Documenting Program Changes

So Much Change
Understanding Change Is Important

Developers
Managers
Evaluation
Understanding change Remains Difficult

Peter Hallam. *What Do Programmers Really Do Anyway?*  
State of the Art: **Diff**

```plaintext
19c19, 22
<   else return "";
---
>   else return pageParts[0];
>   // else return "";
```

JabRef Revision 3066
Automatically Documenting Program Changes

State of the Art:
Side-by-side
State of the Art: Commit Messages

• Free-form text which may describe
  – **What** the change was.
  – **Why** the change was made.

**jfreechart rev 3405**

(start): Changed from Date to long,
(end): Likewise,
(getStartMillis): New method,
(getEndMillis): Likewise,
(getStart): Returns new date instance,
(getEnd): Likewise.

**Jabref rev 2917**

Fixed NullPointerException when downloading external file and file directory is undefined.

**Phex 3542**

Minor change
Subject: An appeal for more descriptive commit messages

I know there is a lot going on but please can we be a bit more descriptive when committing changes. Recent log messages have included:
"some cleanup"
"more external service work"
"Fixed a bug in wiring" which are a lot less informative than others...


Toby, Going forward, could I ask you to be more descriptive in your commit messages? Ideally you should state what you've changed and also why (unless it's obvious)... I know you're busy and this takes more time, but it will help anyone who looks through the log ...


Sorry to be a pain in the neck about this, but could we please use more descriptive commit messages? I do try to read the commit emails, but since the vast majority of comments are "CAY-XYZ", I can't really tell what's going on unless I then look it up.

We Propose: **DeltaDoc**

Describes the observable EFFECT of a change

- Conditions that trigger the changed code
- How the change impacts functional behavior and program state

**Symbolic Execution + Summarization Transformations**
When calling LastPage format(String s)
If s is not null and s.split ("[-]+").length != 2
return s.split ("[-]+")[0] instead of ""

19c19, 22
< else return "";
---
> else return pageParts[0];
Temporarily removed the trade routes from the game menu.

When calling `FreeColMenuBar buildOrdersMenu`:
No longer call `JMenu.add(getMenuItem("assignTradeRouteAction"))`

When calling `FreeColMenuBar buildViewMenu`:
No longer call `JMenu.add(getMenuItem("tradeRouteAction"))`
The rest of this talk

• **Approach:** How *DeltaDoc* works.

• **Evaluation:** Comparing *DeltaDoc* to *Commit messages*.
DeltaDoc Architecture
DeltaDoc Architecture

Compute symbolic path predicates for each statement.
DeltaDoc Architecture

Identify statements that have been added, removed, or have a different predicate.
DeltaDoc Architecture

Apply summarization transformations until result is sufficiently concise.
Example Change

```
String sayHello(String name)
{
    String ret = "";
    if(name != null)
    {
        if(System.Lang == ENG)
            ret = ret + "Hello ";
        else
            ret = ret + "Bonjour ";
    }
    ret = ret + name;
    return ret;
}
```
Example
Predicate Generation

<table>
<thead>
<tr>
<th>Stmt</th>
<th>Symbolic</th>
<th>Path Predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>String ret = &quot;&quot;;</td>
<td>ret = &quot;&quot;</td>
<td>true</td>
</tr>
<tr>
<td>name != null</td>
<td>name != null</td>
<td>true</td>
</tr>
<tr>
<td>System.Lang == ENG</td>
<td>System.Lang = ENG</td>
<td>name != null</td>
</tr>
<tr>
<td>ret = ret + &quot;Hi &quot;;</td>
<td>ret = &quot;&quot; + &quot;Hi &quot;</td>
<td>name != null &amp;&amp; System.Lang = ENG</td>
</tr>
<tr>
<td>ret = ret + name;</td>
<td>ret = &quot;&quot; + &quot;Hi &quot; + name</td>
<td>name != null &amp;&amp; System.Lang = ENG</td>
</tr>
<tr>
<td>return ret;</td>
<td>return &quot;Hi &quot; + name</td>
<td>name != null &amp;&amp; System.Lang = ENG</td>
</tr>
</tbody>
</table>

Automatically Documenting Program Changes
### Example Change Enumeration

<table>
<thead>
<tr>
<th>Stmt</th>
<th>Path Predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>return &quot;&quot;</td>
<td>name == null</td>
</tr>
<tr>
<td>return &quot;Hello &quot; + name</td>
<td>name != null &amp; System.Lang = ENG</td>
</tr>
<tr>
<td>return &quot;Bonjour &quot; + name</td>
<td>name != null &amp; System.Lang != ENG</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Stmt</th>
<th>Path Predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>return &quot;&quot;</td>
<td>name == null</td>
</tr>
<tr>
<td>return &quot;Hi &quot; + name</td>
<td>name != null &amp; System.Lang = ENG</td>
</tr>
<tr>
<td>return &quot;Bonjour &quot; + name</td>
<td>name != null &amp; System.Lang != ENG</td>
</tr>
</tbody>
</table>
Example
Generated Documentation

When calling `sayHello(String name)`

If `name != null AND System.Lang == ENG`
return “Hi ” + name Instead of “Hello ” + name
Summarization: Lossless

• Re-arrange terms (code hoisting)
  - If $P$ and $Q$, Do $X$
  - If $P$ and $Q$ and $R$, Do $Y$

• Simplification (common subexpression)
  - If $\text{iter.hasNext}$
    - call $\text{iter.hasNext}$
    - call $\text{otherFun}$

  - If $\text{iter.hasNext}$
    - call $\text{otherFun}$
Summarization: Lossy

Remove irrelevant path predicates.

If \( s \neq \text{null} \) and \( a \) is true and \( b \) is true and \( c \) is true

\[
\text{return } s
\]
Summarization: Effects

Random search, greedy prioritization of mutation operators.
Evaluation

Size

Content

Quality

(Human study: 16 graduate students)
## Benchmarks

<table>
<thead>
<tr>
<th>Name</th>
<th>Revisions</th>
<th>Domain</th>
<th>kLOC</th>
<th>Developers</th>
</tr>
</thead>
<tbody>
<tr>
<td>FreeCol</td>
<td>2000-2200</td>
<td>Game</td>
<td>91</td>
<td>33</td>
</tr>
<tr>
<td>iText</td>
<td>3800-4000</td>
<td>PDF utility</td>
<td>200</td>
<td>14</td>
</tr>
<tr>
<td>jFreeChart</td>
<td>1700-1900</td>
<td>Presentation</td>
<td>305</td>
<td>3</td>
</tr>
<tr>
<td>JabRef</td>
<td>2500-2700</td>
<td>Reference Management</td>
<td>107</td>
<td>28</td>
</tr>
<tr>
<td>Phex</td>
<td>3500-3700</td>
<td>File Sharing</td>
<td>177</td>
<td>13</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1000</strong></td>
<td></td>
<td><strong>880</strong></td>
<td><strong>91</strong></td>
</tr>
</tbody>
</table>
Size Comparison

Diffs are about 35 lines

DeltaDocs are about 9 lines

Commit messages are always less than 10 lines
Content Comparison

DeltaDoc

Relational Form

Commit Messages

Relational Form
Score Metric

• Conservatively assume only relations from commit messages are important.
• Reward precision.
• Used 16 human annotators to validate.
• Score of 0.5 indicates that the DeltaDoc contained all the information in the commit message.
Results

*DeltaDoc* implies all relations in log message

![Average Score Metric for different tools]

- freecol
- itext
- jfreechart
- jabref
- phex
- average
Results

Score Metric Distribution

- freecol
- itext
- jfreechart
- jabref
- phex
- average

Legend:
- <0.5
- 0.5
- >0.5
Results

About 89% coverage.
Qualitative Evaluation

“very useful” “highly useful” “would be a great supplement” “definitely a useful supplement” “can help make the logic clear” “often easier to understand“ “more accurate” “easy to read" “provides more information"
DeltaDoc

• Cheap
  – Can be computed in about a second on average.

• Suitable for quick adoption
  – Can supplement or replace many existing commit messages.

• Structured
  – Suitable for search.

• Does not address “why”

• Search-based simplification?
Questions?
# Relational Form

<table>
<thead>
<tr>
<th>Type</th>
<th>Arity</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>unary</td>
<td>is true</td>
<td>false</td>
</tr>
<tr>
<td>logical</td>
<td>or</td>
<td>and</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>equal to</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>not equal to</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>less-than</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>greater-than</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>less-than or equal to</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td>greater-than or equal to</td>
</tr>
<tr>
<td>binary</td>
<td>is empty</td>
<td>call / invoke</td>
</tr>
<tr>
<td></td>
<td>unary</td>
<td>return</td>
</tr>
<tr>
<td></td>
<td>unary</td>
<td>throw</td>
</tr>
<tr>
<td>programmatic</td>
<td>assign to</td>
<td>element of</td>
</tr>
<tr>
<td></td>
<td>binary</td>
<td>inserted in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>implies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>instance of</td>
</tr>
<tr>
<td></td>
<td>edit</td>
<td>added</td>
</tr>
<tr>
<td></td>
<td>unary</td>
<td>changed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>removed</td>
</tr>
</tbody>
</table>
Relational Form Example

getPriceForBuilding() > getOwner().getGold()  
has an insufficient amount of gold

getPriceForBuilding() > getOwner().getGold()  
? > gold
Score Metric

• Conservatively assume only relations from commit messages are important.
• Reward precision.
• Used 16 human annotators to validate.
• Score of 0.5 indicates that the DeltaDoc contained all the information in the commit message.
Example Score = 0.5

Commit Message

no need to call clear()

DeltaDoc

When calling PdfContentByte reset()
  If stateList .isEmpty(),
  No longer call stateList .clear()
Example Score > 0.5

Commit Message

Commented unused constant

DeltaDoc

removed field : EuropePanel : int TITLE_FONT_SIZE

Freecol rev 2054
Example Score < 0.5

Commit Message

Fixed bug: content selector for 'editor' field uses ',' instead of 'and' as delimiter.

DeltaDoc

When calling EntryEditor.getExtra()
If ed.getFieldName().equals("editor")
call contentSelectors.add(FieldContentSelector)