Visualizing Source Code History

Alex Bradley

CPSC 533C Project Update
University of British Columbia
November 18, 2009
Problem

• Common software development task: compare revisions of a source code file to identify important changes
  – See how a feature evolved over time
  – See which authors participated in development of the code
  – Find when a bug was introduced (and by whom)
Problem

* For simplicity, phase out top-level refactorings. We should be able to do most of the same stuff with ‘contextual’ refactorings.

- Make the menuProvider extension a little less kludgy by providing explicit factory classes.

- Now that refactoring stuff is in a separate plugin, add some generics. :-)

- Move jQuery-based refactorings out into their own plugin.
Problem

Comparing two revisions: well supported
Comparing **two** revisions: well supported

Viewing **revision notes** at a glance: well supported
Problem

• But how to compare many revisions?
  – Look for evolution of code over several revisions of concern (in my experience, \sim 4-10)
  – Look at overall picture of all revisions to find where interesting stuff might have happened
Solution: Basic Ideas

- Small multiples view for detailed comparison
  - Two alternatives considered:
    - Revisions side by side in a row
    - Revisions in a two-column grid
- “History flow” to display all revisions
  - Each revision as a vertical pixel stripe
  - Lines in revision = horizontal pixel lines in stripe
    - Coloured according to authorship, statement type, code age...
  - Full-text views of a few revisions of interest embedded in flow (focus+context)
Prototype: single row view
Prototype: two-column view
Sketch: history flow
Prototype: “zoom” (change font size)

- At readable font sizes (~6-10), just helps to give more reading space
- When very small, gives coarse-grained visual overview
  - May provide intermediate point (in terms of “compression”) between full-size text and history flow?

For simplicity, phase out top-level refactorings. We should be able to do most of the same stuff with “contextual” refactorings, and the latter are...
Related Work: SeeSoft

Related Work: TableLens

<table>
<thead>
<tr>
<th>Years In Major</th>
<th>Career At Bats</th>
<th>Career Hits</th>
<th>Career Avg</th>
<th>Salary 87</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wade Boggs</td>
<td>5</td>
<td>2778</td>
<td>978</td>
<td>0.35205182</td>
</tr>
<tr>
<td>Don Mattingly</td>
<td>5</td>
<td>2223</td>
<td>737</td>
<td>0.33153397</td>
</tr>
</tbody>
</table>

Related Work: History Flows

Related Work: History Flows

Related Work: Visual Code Navigator

Implementation Approach

• Initially considered standalone Java app
• Decided to implement as Eclipse plugin using Java SWT
  – Disadvantage: complexity of Eclipse plugin framework and SWT
  – Advantages:
    • Easy access to good Java CVS library (internal to Eclipse)
    • Easy access to other Eclipse features (e.g., syntax highlighting)
    • Integration with existing IDE makes it easy to apply to real code
Milestones

- **Friday, November 6**: Development environment set up; prototype capable of accessing CVS repository and downloading code. Small-multiples prototype started.
- **Monday, November 16**: Prototype of small-multiples views complete. Prototype of focus+context view started.
- **Wednesday, November 18**: Project update presentation.
- **Friday, November 27**: Prototypes of all views complete.
- **Thursday, December 10**: Final implementation complete. Final presentation and report drafted.
- **Monday, December 14**: Final project presentations.
- **Wednesday, December 16**: Final report submitted.
Milestones

• **Friday, November 6:** Development environment set up; prototype capable of accessing CVS repository and downloading code. Small-multiples prototype started. ✔

• **Monday, November 16:** Prototype of small-multiples views complete. ✔ Prototype of focus+context view started.

• **Wednesday, November 18:** Project update presentation. ✔

• **Friday, November 27:** Prototypes of all views complete.

• **Thursday, December 10:** Final implementation complete. Final presentation and report drafted.

• **Monday, December 14:** Final project presentations.

• **Wednesday, December 16:** Final report submitted.
Work Remaining

• Implement history flow view
  – See what types of colouring we can provide
  – Add corresponding colouring to small-multiples views

• Add graphical comparison-to-neighbours markings to current small-multiples views

• Other UI conveniences
  – Scrolling revision panes in tandem, resizing columns, suppressing revision notes, etc.

• Test with a realistic code task, find limitations of approach
  – Given limited screen space, how many revisions can we effectively compare at once?
Questions/suggestions?