Understanding changes with Torch

Verónica Uquillas Gómez
vuquilla@vub.ac.be
About me


PhD candidate (interuniversity program)

✓ Vrije Universiteit Brussel, Belgium
  - Software Language Lab / prof. dr. Theo D’Hondt

✓ Université Lille1, France
  - RMoD Team / prof. dr. Stéphane Ducasse
Conceptual process of change integration*

*in open source software*
Conceptual process of change integration

Characterize and Understand Changes

*in open source software
Integration is difficult
Integration is difficult

- Requires expertise on the system
Integration is difficult

- Requires expertise on the system
- May demand a lot of time
Integration is difficult

- Requires expertise on the system
- May demand a lot of time
- Not enough support for integrators*

*Note: This is not a specific condition, but rather a general observation about the context of integration.
Integration is difficult

- Requires expertise on the system
- May demand a lot of time
- Not enough support for integrators*

*deciding when changes should be integrated or not
Integrators are using ...
Integrators are using ...
Integrators are using ...
We want to support...
We want to support...

- Integrators ★
We want to support...

- Integrators
  ✓ aiding in understanding changes
We want to support...

- Integrators
  - aiding in understanding changes
  - taking decisions about the integration process
We want to support...

- Integrators
  - aiding in understanding changes
  - taking decisions about the integration process
- Developers
We want to support...

- Integrators
  ✔ aiding in understanding changes
  ✔ taking decisions about the integration process

- Developers
  ✔ controlling their changes before publishing
The Torch Dashboard: Changes from SLICE-Issue1709-EnhancedTextDiffBuilder (ancestor) to SLICE-Issue1709-EnhancedTextDiffBuild
Torch
Torch

- Visualization tool
Torch

- Visualization tool
  - provides an overview of changes
Torch

- Visualization tool
  - provides an overview of changes
  - uses structural & symbolic information
Torch

- Visualization tool
  - ✓ provides an overview of changes
  - ✓ uses structural & symbolic information
- Written in Smalltalk - Pharo
Torch

- Visualization tool
  - provides an overview of changes
  - uses structural & symbolic information
- Written in Smalltalk - Pharo
- Integrated with Monticello 1
Extracting Changes

Scenario two: changes in present context
delta D-C'
version D
latest version of trunk (svn) / head (git)

Scenario one: changes in the past
version C'
delta C'-B
version B
common ancestor

version C

version A
The dashboard shows ...

- Packages
  - Classes | Traits
    - Methods
    - Variables
  - Protocols*
- Authors
- Relationships

*secondary entities in the dashboard
Change Operations

- Additions
- Modifications
- Removals
- Movements (classes)
The Torch Dashboard main components
Visual Representation

- Rectangles
  - ✓ packages, classes, traits, methods

- Triangles
  - ✓ variables

- Edges
  - ✓ class-inherits-class
  - ✓ class-uses-trait
  - ✓ class-is-extended-in-package
Component: Change visualizations
**Component: Change visualizations**

<table>
<thead>
<tr>
<th>Package-centric</th>
<th>Class-centric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packages</strong></td>
<td><strong>Classes</strong></td>
</tr>
<tr>
<td>changed</td>
<td></td>
</tr>
</tbody>
</table>

*Different change visualizations offered by Torch*
Component: Change visualizations

<table>
<thead>
<tr>
<th>Package-centric</th>
<th>Class-centric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packages</strong></td>
<td><strong>Classes</strong></td>
</tr>
<tr>
<td>changed</td>
<td>structural</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Component: Change visualizations

<table>
<thead>
<tr>
<th></th>
<th><strong>Package-centric</strong></th>
<th><strong>Class-centric</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packages</strong></td>
<td>changed</td>
<td>changed and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dependents</td>
</tr>
<tr>
<td><strong>Class visual representation</strong></td>
<td>structural</td>
<td>structural</td>
</tr>
</tbody>
</table>

Different change visualizations offered by Torch
## Component: Change visualizations

<table>
<thead>
<tr>
<th>Package-centric</th>
<th>Class-centric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packages</strong></td>
<td><strong>Classes</strong></td>
</tr>
<tr>
<td>changed</td>
<td>changed and dependents</td>
</tr>
<tr>
<td>condensed</td>
<td>structural</td>
</tr>
</tbody>
</table>

Different change visualizations offered by Torch
Component: Change visualizations

<table>
<thead>
<tr>
<th>Package-centric</th>
<th>Class-centric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packages</strong></td>
<td><strong>Classes</strong></td>
</tr>
<tr>
<td>changed</td>
<td>structural</td>
</tr>
<tr>
<td>changed and unchanged</td>
<td>condensed</td>
</tr>
<tr>
<td>changed and unchanged</td>
<td>condensed</td>
</tr>
<tr>
<td>changed and unchanged</td>
<td>condensed</td>
</tr>
</tbody>
</table>

Different change visualizations offered by Torch
## Component: Change visualizations

<table>
<thead>
<tr>
<th>Package-centric</th>
<th>Class-centric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packages</strong></td>
<td><strong>Classes</strong></td>
</tr>
<tr>
<td>changed</td>
<td>changed and dependents</td>
</tr>
<tr>
<td>changed and unchanged</td>
<td>condensed</td>
</tr>
</tbody>
</table>

**Symbolic Clouds**

Different change visualizations offered by Torch
Package Structure

Visualization: Changed Packages (details)
Class Representation

Visual representation of changed classes
Omnipresent source code

```
instance method  Protocol: browse  Author: AndyKelliens

browseAllObjectReferencesTo: anObject except: objectsToExclude ifNone: aBlock
"Bring up a list inspector on the objects that point to anObject.
If there are none, then evaluate aBlock on anObject."
| aList shortName |
aList := anObject pointersToExcept: objectsToExclude.
aList := PointerFinder pointersTo: anObject except: objectsToExclude.
aList size > 0 ifFalse: [^aBlock value: anObject].
shortName := (anObject name ifNil: [anObject printString]) contractTo: 20.
aList inspectWithLabel: 'Objects pointing to ', shortName
```
Omnipresent class structure
Omnipresent class structure
Component: Summaries

- Metrics with overview of changes
  - ✔ Entities x Operations
  - ✔ Users x Operations
- Filter the changes
Component: Conventions

- Colors represent change operations on entities
- Borders distinguish some entities
- Icons emphasize changes
Component: Parameters

- Affect the changes visualizations component
  - Class status: added, modified, removed, unchanged, **changed***
  - Relationships: inter- or intra- packages

*Changed := added | modified | removed*
Component: Change list

- Shows selected changes
- Resembles the Monticello change list, but ...
  - ✓ class + metaclass = one class
  - ✓ adds extra information
  - ✓ sorted
Component: Change details

- Offer textual information of changes
Component: Change details

- Offer textual information of changes
Component: Change details

- Offer textual information of changes
Torch in action

Package slices: PharolInbox and PharoTreatedInbox repositories
What do you think it happened?
Removing a feature (I)

Removed **Pen** and **PenPointRecorder**
Removing a feature (II)
Removing a feature (II)
PointerFinder

PointersTo: anObject

self deprecated: 'Use ProtoObject>>pointersTo instead.'

"Find all occurrences in the system of pointers to the argument anObject:"

"(PointerFinder pointersTo: Browser) inspect."

^ self pointersTo: anObject except: #()
Removing a feature (III)

Removed **PointerFinder** by deprecating its API
Introducing a feature
Introducing a feature

Introduced **PopupChoice** / **RequestDialog**
Changing Hierarchies

- Hierarchies may be distributed in different packages
- Showing inter-package relationship is needed
- Still... We can show it better!!!
Changing Hierarchies

- Hierarchies may be distributed in different packages
- Showing inter-package relationship is needed
- Still... We can show it better!!!
Changing Hierarchies

- Hierarchies may be distributed in different packages
- Showing inter-package relationship is needed
Changing Hierarchies

- Hierarchies may be distributed in different packages
- Showing inter-package relationship is needed
- Still... We can show it better!!!
Changing Hierarchies

- Hierarchies may be distributed in different packages
- Showing inter-package relationship is needed
- Still... We can show it better!!!
Changing Hierarchies

Visualization: Changed classes (details)
Changing Hierarchies
Changing Hierarchies
Changing Hierarchies

Pushed up **indexOfAnyOf:** / Introduced **findFirstInByteString:startingAt:**
A yellow box next to the class’ name represents a modified comment.
Editing comments

- Packages (138)
- Classes (1031)
- Methods (23434)
- Variables (2886)

A yellow box next to the class' name represents a modified comment
Can you see a change pattern?
Symbolic Clouds

- Provide
  - ✓ Vocabulary involved in changes
  - ✓ Hints towards the developers’ intentions*
- Methods’ source code ➔ vocabulary
- Three symbolic clouds

*if there is little vocabulary involved
Replacing method calls

Visualization: Symbolic clouds
Symbolic Clouds (II)
Symbolic Clouds (II)

Parameters \texttt{at:ifAbsent: at:ifAbsentPut: at:put:}

Mixed Symbols (added + removed)

Symbolic clouds showing replaced method calls
Symbolic Clouds (II)

Mixed Symbols (added + removed)

Parameters `at:ifAbsent:at:ifAbsentPut:`

Added Symbols (method calls + class references + attribute accesses)

Removed Symbols (method calls + class references + attribute accesses)

Symbolic clouds showing replaced method calls
Summary

- Torch v1  -  http://soft.vub.ac.be/torch
  - change characterization
  - change overview
  - omnipresent contextual diff
Summary

- Torch v1 - http://soft.vub.ac.be/torch
- ✓ change characterization
- ✓ represent contextual diff
- understanding changes