Using Information Retrieval to Support Software Evolution

Denys Poshyvanyk
Ph.D. Candidate

SEVERE Group @

Wayne State University
Software is Everywhere

- Software is pervading every aspect of life
- Software is difficult to make reliable and bug-free
  - no physical limitations
  - expressive and flexible, but complex
  - invisible
- Software is difficult to design and maintain
  - as much as 80% of 100 billion lines of code in production are unstructured, patched, badly documented [van Vliet’00]
  - up to 90% of the total cost of software are spent on maintenance [Erlikh’00]
Addressed Problem

• The software developer has to maintain large software systems with:
  - Little or no domain knowledge
  - Absence of the original developer
  - Badly organized, missing, or out of date documentation
Software Maintenance Requires

- **Structural Information** - the structural aspects of the source code (e.g., dependencies, architecture)

- **Dynamic information** - behavioral aspects of the program (e.g., feature implementation overlap)

- **Lexical Information** - nature of the problem domain derived from names - identifiers, comments, documentation, and other artifacts

- **Historical Information** - history of changes (co-changes) of different software artifacts (e.g., CVS logs - rationale for changes)
Multitude of Software Artifacts

Source code

Call graphs

Design docs

Bug Reports

Documentation

Execution Traces
Research Focus

- **Structural Information** - the structural aspects of the source code (e.g., dependencies, architecture)

- **Dynamic information** - behavioral aspects of the program (e.g., feature implementation overlap)

- **Lexical Information** - nature of the problem domain derived from names - identifiers, comments, documentation, and other artifacts

- **Historical Information** - history of changes (co-changes) of different software artifacts (e.g., CVS logs - rationale for changes)
Multitude of Software Artifacts

Source code

Call graphs

Design docs

readAndDispatch -- org.eclipse.swt.widgets.Display
checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.graphics.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- org.eclipse.swt.widgets.Display
filterMessage -- org.eclipse.swt.widgets.Display
windowProc -- org.eclipse.swt.widgets.Control
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
Combining Structural, Dynamic and Lexical Information
Extracting Lexical Information from Existing Software Systems

- Expensive - large amounts of domain knowledge needs to be encoded

- Existing approaches use natural language processing [Shepherd’07] and knowledge based methods [Etzkorn’97]

- Our approach is to use less expensive methods
Latent Semantic Indexing (LSI)

- Vector space model based Information Retrieval method [Dumais 94, Berry 95, Deerwester 90]

- It has been used successfully as a method to represent aspects of the meaning of words and passages in natural language

- Ignores word ordering and semantically unimportant terms (e.g., ‘the’, ‘and’, ‘for’), syntactic relations, and morphology

- It does not utilize a predefined grammar or vocabulary
Why Use LSI?

• Captures essential semantic info via dimensionality reduction

• Overcomes problems with polysemy and synonymy

• (Programming and natural) language independent

• Easy to apply on the source code
LSI in Software Engineering

- Abstract data type identification [Maletic’01]
- Traceability link recovery [Marcus’03]
- Concept location [Marcus’04]
- Software artifact management [DeLucia’04]
- Requirements traceability [Hayes’04]
- Conceptual cohesion of classes [Marcus’05]
- Software clustering [Kuhn’05]
- Conceptual coupling of classes [Poshyvanyk’06]
- Feature location [Poshyvanyk’07]
Extracting and Indexing Lexical Information with Information Retrieval

• Parsing source code and extracting documents
  - corpus is a collection of documents (e.g., methods)

• Removing non-literals and stop words
  - common words in English, standard function library names, programming language keywords

• Preprocessing: split_identifiers and SplitIdentifiers

• Indexing and retrieving semantic information with Latent Semantic Indexing
How Does LSI Work on Source Code?

```java
public void run(IProgressMonitor monitor) throws InvocationTargetException, InterruptedException {
    if (m_iFlag == 0)
        processCorpus(monitor, checkUpdate());
    else if (m_iFlag == 2)
        processCorpus(monitor, UD_UPDATECORPUS);
    else
        processQueryString(monitor);

    if (monitor.isCanceled())
        throw new InterruptedException("The long running
```

**Singular Value Decomposition (SVD):**

\[
[nm] = [nk] [kk] [mk]', k < n
\]

**Similarity Measure:**

Cosine of the contained angle between the vectors

**Consider only ‘k’ largest singular values for the LSI subspace**
Currently Addressed Maintenance Tasks

- Feature (or concept) location in source code
- Impact analysis in source code
- Fault prediction in source code
- Traceability link recovery
Currently Addressed Maintenance Tasks

• Feature (or concept) location in source code
• Impact analysis in source code
• Fault prediction in source code
• Traceability link recovery
Concept Location in Source Code

- Change request
- Concept Location
- Impact Analysis
- Implementation
- Change Propagation
- Testing
Concept Location in Practice and Research

- **Static**
  - Dependency based search [Rajlich’00]
  - IR methods [Marcus’04]

- **Dynamic**
  - Execution traces - Reconnaissance [Wilde’92]
  - Scenario based probabilistic ranking [Antoniol’06]

- **Combined**
  - Profiling with concept analysis [Eisenbarth’03]
  - Feature dependencies [Salah’05]
  - Feature evolution [Greevy’05]
  - PROMESIR [Poshyvanyk’07]
Concept Location with
Regular Expressions

BSTnode BST::insert(KeyType key, ValueType value)
// lookup key, create key-value, if not present
return pointer to node with key
{
    BSTnode n = BSTnode(key);
    n.item.second = value;
    return n;
}

BSTnode BST::getnode(KeyType key)
// lookup key, create key-value, if not present
// return pointer to node with key
{
    pair<KeyType, ValueType> item(key, 0); // create item to be inserted
    if (_root == 0)
    {
        BSTnode new_node = new BSTnode(0, 0, item);
        _root = new_node;
        return _root;
    }
    pair<BSTnode, bool> result = find(key);
    if (result.second)
    {
        // true, node exists
        return result.first;
    }
    else
    { // false, create node
        BSTnode parent = result.first;
        BSTnode new_node = new BSTnode(0, 0, item);
        if (key < parent->key)
        { // parent->left
            parent->left = new_node;
        } else
        { // parent->right
            parent->right = new_node;
        }
        return new node;
    }
}
Our Motivation
Concept Location with Information Retrieval

[Marcus’04]

BSTNode BST::insert(KeyType key, ValueType value)
// lookup key, create key-value if not present
// return pointer to node with key
{
    BSTNode * n = getNode(key);
    n->item.second = value;
    return n;
}

BSTNode BST::getNode(KeyType key)
// lookup key, create key-value if not present
// return pointer to node with key
{
    pair<KeyType, ValueType> item(key, 0); // create item to be inserted
    if (_root == 0)
    {
        BSTNode new_node = new BSTNode(0, 0, 0, Item);
        _root = new_node;
        return _root;
    }

    pair<BSTNode*, bool> result = find(key); // search for key
    if (result.second) // location found
    {
        return *result.first;
    }
    else
    {
        BSTNode* parent = result.first;
        BSTNode* new_node = new BSTNode(0, 0, parent, Item);
        if (key < parent->key)
        {
            parent->left = new_node;
        }
        else
        {
            parent->right = new_node;
        }
        return *new_node;
    }
}

IRISS

Find What: set screen view

IRISS Version 1.0
Wayne State University

About

Exit

<table>
<thead>
<tr>
<th>Class</th>
<th>Method</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td>setScreenParams</td>
<td>0.813055</td>
</tr>
<tr>
<td>ScaleObjectTool</td>
<td>mouseDragged</td>
<td>0.807305</td>
</tr>
<tr>
<td>Camera</td>
<td>setScreenParamsParalle</td>
<td>0.80328</td>
</tr>
<tr>
<td>Camera</td>
<td>setSize</td>
<td>0.788526</td>
</tr>
<tr>
<td>ViewerCanvas</td>
<td>setScale</td>
<td>0.723998</td>
</tr>
<tr>
<td>GLCanvasDrawer</td>
<td>prepareView3D</td>
<td>0.705305</td>
</tr>
<tr>
<td>ViewerCanvas</td>
<td>scaleChanged</td>
<td>0.700639</td>
</tr>
<tr>
<td>JitterModule</td>
<td>setZScale</td>
<td>0.690434</td>
</tr>
<tr>
<td>JitterModule</td>
<td>setXScale</td>
<td>0.690117</td>
</tr>
<tr>
<td>ImageModule</td>
<td>setXScale</td>
<td>0.690104</td>
</tr>
</tbody>
</table>
IRiSS

<table>
<thead>
<tr>
<th>Problems</th>
<th>Javadoc</th>
<th>IRiSS View</th>
<th>Method</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td>setScreenParams</td>
<td>0.813055</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ScaleObjectTool</td>
<td>mouseDragged</td>
<td>0.807305</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camera</td>
<td>setScreenParamsParalle</td>
<td>0.80328</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camera</td>
<td>setSize</td>
<td>0.788526</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ViewerCanvas</td>
<td>setScale</td>
<td>0.723998</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GLCanvasDrawer</td>
<td>prepareView3D</td>
<td>0.705305</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ViewerCanvas</td>
<td>scaleChanged</td>
<td>0.700639</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JitterModule</td>
<td>setZScale</td>
<td>0.690434</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JitterModule</td>
<td>setXScale</td>
<td>0.690117</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ImageModule</td>
<td>setXScale</td>
<td>0.690104</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

IRiSS

Output

IRiSS - persistent cache

- CCrystalTextView::WrapLineCached | confidence: 0.87 |
- CCrystalTextView::InvalidateLineCache | confidence: 0.84 |
- CCrystalTextView::DrawSingleLine | confidence: 0.82 |
- CCrystalTextView::OnDraw | confidence: 0.79 |
- CCrystalTextView::SetFont | confidence: 0.68 |
- CCrystalTextView::OnSize | confidence: 0.59 |
- CCrystalTextView::CalculateActualOffset | confidence: 0.57 |

---

Google Eclipse Search

Enter the search string:

Animation Preview

Search Results:

- ActionEditorWindow.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/animation (11 matches)
- AnimatorPreview.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/animation (19 matches)
- CActorObject.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/animation/object (2 matches)
- CustomDistortionTrack.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/animation/distortion (10 matches)
- CylinderMapping.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/texture (12 matches)
- LinearMapping3D.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/texture (12 matches)
- ObjectInfo.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/animation/object (13 matches)
- ObjectTextureDialog.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/texture (6 matches)
- ProcedurePositionTrack.java - Art of Illusion/ArtOfIllumion/src/artOfIllumion/animation (6 matches)
Feature Location with Information Retrieval and Formal Concept Analysis

[Poshyvanyk’07]
Feature Location with Software Reconnaissance

Scenario NOT exercising the feature

readAndDispatch -- org.eclipse.swt.widgets.Display
checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.graphics.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- org.eclipse.swt.widgets.Display
filterMessage -- org.eclipse.swt.widgets.Display
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control

Scenario exercising the feature

checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.graphics.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- 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.ProgressBar
windowProc -- org.eclipse.swt.widgets.Control
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Control
Feature Location with Scenario-based Probabilistic Ranking [Antoniol’06]

Scenario NOT exercising the feature
readAndDispatch -- org.eclipse.swt.widgets.Display
checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.graphics.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- 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
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control

Scenario exercising the feature
checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.graphics.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- org.eclipse.swt.widgets.Display
filterMessage -- org.eclipse.swt.widgets.Display
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control
windowProc -- org.eclipse.swt.widgets.Display
WM_TIMER -- org.eclipse.swt.widgets.Control

Knowledge-based filtering
Probabilistic ranking of events
Shortcomings

• Static analysis:
  - Highly dependent on naming conventions and the developer’s talent to write good queries
  - Ignores other existing relationships between software components (such as, dependencies)
  - May miss important parts of the source code

• Dynamic analysis:
  - Execution traces are large even for small systems
  - Selecting multiple scenarios may be difficult
  - Filtering the traces is equally problematic - best filtering methods still return hundreds of methods
Probabilistic Ranking Of Methods and Information Retrieval

Scenario NOT exercising the feature

readAndDispatch -- org.eclipse.swt.widgets.Display
checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- 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

Scenario exercising the feature

checkDevice -- org.eclipse.swt.widgets.Display
isDisposed -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Display
runPopups -- org.eclipse.swt.widgets.Display
checkDevice -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
drawMenuBars -- org.eclipse.swt.widgets.Device
windowProc -- org.eclipse.swt.widgets.Display
windowProc -- org.eclipse.swt.widgets.Display
windowProc -- org.eclipse.swt.widgets.Display
windowProc -- org.eclipse.swt.widgets.Control

[Poshyvanyk’07]
Feature Location with PROMESIR

- Feature identification - decision making problem in presence of uncertainty

- Static (LSI) and dynamic (SPR) methods:
  - LSI queries static documents
  - SBP analyzes dynamic traces of execution scenarios

- Complementary results are combined via affine transformation
Scenario-based Probabilistic Ranking

• Collecting Execution Traces
  - **Multiple** (ir) relevant scenarios are executed to collect traces
  - processor emulation (VALGRIND for C++) to improve the precision of data collection
  - byte code instrumentation (JIKES for Java)

• Knowledge-based filtering to eliminate noisy events

• Probabilistic ranking
  - events are re-weighted (Wilde’s equation is renormalized)
Feature Location with PROMESIR

1. Query
2. Scenarios
3. Events
4. Execution traces
5. Indexes
6. Ranks
7. Results
Example of using PROMESIR

- Locating a feature in JEdit
- Feature: “showing **white-space** as a visible symbol in the **text area**”

**Steps:**
- Run two scenarios
- Run query
- Explore results
First Scenario Exercising the Feature in JEdit

Start Tracing
First Scenario Exercising the Feature in JEdit

```java
public class JEditTextArea extends JComponent {
    // constructor
    public JEditTextArea(View view) {
        // adds the normal cursor
        Normal = newCursor(Cursor.TEXT_CURSOR);
        Hand = newCursor(Cursor.HAND_CURSOR);
        
        // enables events
        enableEvents(AWTEvent.FOCUS_EVENT_MASK | AWTEvent.KEY_EVENT_MASK);

        // view
        this.view = view;

        // initialize some misc. stuff
        urlSelection = new Vector();
        selection = new Vector();
        chunkCache = new ChunkCache(this);
        painter = new TextAreaPainter(this);
        painter.addExtension(new ToolTipTextAreaExtension(this));
        gutter = new Gutter(view, this);
        listenerList = new EventListenerList();
        caretEvent = new MutableCaretEvent();
```
Second Scenario **NOT** Exercising the Feature in JEdit

```java
public class JEditTextArea extends JComponent {
   
   //{{{{ JEditTextArea constructor.
   //**
   // * Creates a new JEditTextArea.
   //**
   public JEditTextArea(View view) {
      
      //add by lzq.
      Normal=new Cursor(Cursor.TEXT_CURSOR);
      Hero=new Cursor(Cursor.HAND_CURSOR);
      
      enableEvents(AWTEvent.FOCUS_EVENT_MASK | AWTEvent.KEY_EVENT_MASK);
      
      this.view = view;
      
      //{{{{ Initialize some misc. stuff.
      //add by lzq.
      urlselection = new Vector();
      selection = new Vector();
      chunkCache = new ChunkCache(this);
      painter = new TextAreaPainter(this);
      painter.addExtension(new ToolTipTextAreaExtension(this));
      gutter = new Gutter(view, this);
      listenerList = new EventListenerList();
      caretEvent = new NotableCaretEvent();

   }
```
Example of using PROMESIR - Results

- Number of methods identified by SPR - 284
- The position of the first relevant method according to IR ranking - 56
- Position of the first relevant method according to PROMESIR - 7
Case Studies

• Locating features associated with bugs in:
  - Mozilla (4,853 classes; 53,617 methods)
  - Eclipse (7,648 classes; 89,341 methods)

• Case study objectives:
  - Compare PROMESIR with stand-alone feature location approaches: LSI and SPR
## Locating Features in Mozilla

<table>
<thead>
<tr>
<th>Bug #</th>
<th>LSI</th>
<th>SPR</th>
<th>PROMESIR</th>
<th>PROMESIR over LSI</th>
<th>PROMESIR over SPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>182192</td>
<td>37</td>
<td>94</td>
<td>2</td>
<td>18×</td>
<td>47×</td>
</tr>
<tr>
<td>216154</td>
<td>76</td>
<td>332</td>
<td>6</td>
<td>12×</td>
<td>55×</td>
</tr>
<tr>
<td>225243</td>
<td>24</td>
<td>472</td>
<td>6</td>
<td>4×</td>
<td>80×</td>
</tr>
<tr>
<td>209430</td>
<td>49</td>
<td>509</td>
<td>1</td>
<td>49×</td>
<td>509×</td>
</tr>
<tr>
<td>231474</td>
<td>18</td>
<td>494</td>
<td>1</td>
<td>18×</td>
<td>494×</td>
</tr>
</tbody>
</table>
# Locating Features in Eclipse

<table>
<thead>
<tr>
<th>Bug #</th>
<th>LSI</th>
<th>SPR</th>
<th>PROMESIR</th>
<th>PROMESIR over LSI</th>
<th>PROMESIR over SPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>5138</td>
<td>7</td>
<td>268</td>
<td>1</td>
<td>7×</td>
<td>259×</td>
</tr>
<tr>
<td>31779</td>
<td>2</td>
<td>170</td>
<td>1</td>
<td>2×</td>
<td>165×</td>
</tr>
<tr>
<td>74149</td>
<td>5</td>
<td>456</td>
<td>3</td>
<td>2×</td>
<td>156×</td>
</tr>
</tbody>
</table>
Single Trace Information Retrieval (SITIR)

Single Execution Trace

Source Code

Information Retrieval Engine
Collecting Execution Traces in SITIR

- Java Platform Debugger Architecture (JPDA)$^1$
  - Infrastructure to build end-user debugging applications for Java platform

- JPDA highlights:
  - Debugger works on a separate virtual machine
  - Minimal interference of a tracing tool with a subject program
  - Separate thread-based traces
  - Marked traces (start/stop recording)

$^1$http://java.sun.com/javase/technologies/core/toolsapis/jpda/
Case Studies

- Locating features in JEdit
- Locating features associated with bugs in Eclipse

Case study objectives:
- Compare SITIR with other approaches
- Study the impact of full and marked traces on the results
- Study the impact of user queries on the results
# Multiple Developers and Queries for JEdit Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Dev</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>1</td>
<td>search find phrase word text</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>search final all forward backward case sensitive</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>find search locate match indexof findnext</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>searchdialog find findbtn searchselection save searchfiles set searchandreplace</td>
</tr>
<tr>
<td>Show whitespace</td>
<td>1</td>
<td>red dot newline whitespace view show display tab</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>show hide whitespace blank space display</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>symbol replace changecolor setvisible addlayer whitespace loadsymbol</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>userinput textareapainter paint whitespace newline pnt</td>
</tr>
<tr>
<td>Add marker</td>
<td>1</td>
<td>marker select word display text</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>add remove marker markers</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>select highlight mark change background</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>buffer addmarker marker selection</td>
</tr>
</tbody>
</table>
## Results for Different Users and Queries for JEdit Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Dev</th>
<th>Full Trace</th>
<th>Marked Trace</th>
<th>LSI</th>
<th>SITIR full</th>
<th>SITIR marked</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Search</strong></td>
<td>1</td>
<td>1477</td>
<td>202</td>
<td>61</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1477</td>
<td>202</td>
<td>243</td>
<td>57</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1477</td>
<td>202</td>
<td>32</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1477</td>
<td>202</td>
<td>189</td>
<td>36</td>
<td>11</td>
</tr>
<tr>
<td><strong>Show white space</strong></td>
<td>1</td>
<td>1462</td>
<td>284</td>
<td>956</td>
<td>152</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1462</td>
<td>284</td>
<td>626</td>
<td>130</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1462</td>
<td>284</td>
<td>497</td>
<td>104</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1462</td>
<td>284</td>
<td>78</td>
<td>23</td>
<td>8</td>
</tr>
<tr>
<td><strong>Add marker</strong></td>
<td>1</td>
<td>1478</td>
<td>304</td>
<td>26</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1478</td>
<td>304</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1478</td>
<td>304</td>
<td>3242</td>
<td>662</td>
<td>160</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1478</td>
<td>304</td>
<td>20</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>
Locating Features in Eclipse - Results

<table>
<thead>
<tr>
<th>Feature</th>
<th>SINGLE TRACE</th>
<th>SPR</th>
<th>PROMESIR</th>
<th>SITIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>721</td>
<td>268</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Add files</td>
<td>740</td>
<td>170</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Search</td>
<td>771</td>
<td>456</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>
Summary for SITIR

- Uses only single scenario
- Less sensitive to scenario selection
- Unobtrusive tracing mechanism
- IR-based indexing of source code can be easily extended to other languages
- SITIR results are comparable or better than other feature location methods
Currently Addressed Maintenance Tasks

- Feature (or concept) location in source code
- Impact analysis in source code
- Fault prediction in source code
- Traceability link recovery
Impact Analysis in Software

- Change request
- Concept Location
- Impact Analysis
- Implementation
- Change Propagation
- Testing
Impact Analysis in Object-Oriented Systems Using Structural Coupling Measures [Briand’99]

- Coupling between classes (CBO) [Chidamber’04]
- Response for class (RFC) [Chidamber’04]
- Message passing coupling (MPC) [Li’93]
- Data abstraction coupling (DAC) [Li’93]
- Information-flow based coupling (IPC) [Lee’95]
Conceptual Coupling between Methods - Example

- Methods from MySecManager class in Mozilla

<table>
<thead>
<tr>
<th>Method</th>
<th>Policy</th>
<th>Wrapper</th>
<th>Exception</th>
<th>Security</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanCreateWrapper</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CanCreateInstance</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CanGetAccess</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Conceptual Coupling in Object-Oriented Systems

- Measures overlap of shared names in comments and identifiers

Conceptual coupling between A and B = 0.56
Evaluation Methodology

- For a given software system, a set of bug reports is mined from the bug tracking repository.

- The set of classes, which are changed to fix bugs, is mined from concurrent version system.

- For each changed class, conceptual and structural coupling metrics are computed and the other classes in the software system are ranked using these measures.

- Select top classes in each ranked list of classes produced by every metric.
Example

- The bug #232570 reports some problems associated with ‘ldap2.server.position values for ab pane and search order’ in Mozilla

- A starting point, class `nsAbDirectoryQuery` is identified using a feature location technique

- We need to identify other classes that need to be modified (4,853 classes in Mozilla!)
Example

- Compute structural and conceptual coupling of nsAbDirectoryQuery and all the other classes in Mozilla

<table>
<thead>
<tr>
<th>Rank</th>
<th>Conceptual coupling</th>
<th>Values</th>
<th>Information flow coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>nsAbQueryLDAPMessageListener</td>
<td>0.86</td>
<td>nsDebug</td>
</tr>
<tr>
<td>2</td>
<td>nsAbMDBDirectory</td>
<td>0.81</td>
<td>nsAFlatString</td>
</tr>
<tr>
<td>3</td>
<td>nsAbDirectoryQuerySimpleBoolExpression</td>
<td>0.79</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>nsAbLDAPDirectory</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>nsAbView</td>
<td>0.72</td>
<td></td>
</tr>
</tbody>
</table>
Precision vs. Recall

All classes in Mozilla (4,853)

Precision = \[
\frac{|\text{RelRetrieved}|}{|\text{Retrieved}|}
\]

Recall = \[
\frac{|\text{RelRetrieved}|}{|\text{Rel in Collection}|}
\]
Precision and Recall: Example

- Total # of modified classes - 9
- Precision = 2/5 * 100% = 40%
- Recall = 2/9 * 100% = 22%

<table>
<thead>
<tr>
<th>Rank</th>
<th>Conceptual coupling</th>
<th>Information flow coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Classes</td>
<td>Values</td>
</tr>
<tr>
<td>1</td>
<td>nsAbQueryLDAPMessageListener</td>
<td>0.86</td>
</tr>
<tr>
<td>2</td>
<td>nsAbMDBDirectory</td>
<td>0.81</td>
</tr>
<tr>
<td>3</td>
<td>nsAbDirectoryQuerySimpleBoolExpression</td>
<td>0.79</td>
</tr>
<tr>
<td>4</td>
<td>nsAbLDAPDirectory</td>
<td>0.76</td>
</tr>
<tr>
<td>5</td>
<td>nsAbView</td>
<td>0.72</td>
</tr>
</tbody>
</table>
Mining Bugs and Actual Changes

- Bugzilla for Mozilla contains 256,613 bug entries
- Bugs between version 1.6 and 1.7 (1,021 entries)
- Mined 391 bug entries:
  - With approved patches
  - Officially closed (resolved)
  - Contained modifications to at least 2 classes

- Statistics:
  - Average # of modified classes per bug report - 7.3
  - Standard deviation - 14.6

- Outliers removed:
  - bug report #226439, contained the record number of modified classes (149!)
Using Conceptual Coupling to Rank Classes in Mozilla

Recall and Precision vs. Threshold

- Precision
- Recall
Using Coupling between Objects (CBO) to Rank Classes in Mozilla
Using Information-based Flow Coupling to Rank Classes in Mozilla
Conclusions

• Combining lexical, structural and dynamic information shows promise in addressing a number of maintenance tasks

• Using LSI to retrieve lexical information proves to be efficient and fairly precise
Current and Future Work

- Investigate combinations of Information Retrieval and Natural Language Processing techniques to analyze lexical information in software artifacts
- Combing textual information with project history to support other maintenance tasks:
  - Fault-prediction
  - Traceability link recovery and maintenance
Related Publications


Contact

Denys Poshyvanyk
denys@wayne.edu
www.cs.wayne.edu/~denys

SEVERE Group @

Wayne State University