Exemplar: A Search Engine For Finding Highly Relevant Applications

Mark Grechanik, Chen Fu, Qing Xie, Collin McMillan, Denys Poshyvanyk and Chad Cumby

What do we look for when reusing code?
Problem And Solution Spaces

Problem Space

- Sweet, love, harmony, ...

Solution Space

- Encrypt, send, receive, XML, ...

Requirements Document

- INSTRUCTION MANUAL
  Step 1, Open this booklet.
Our Goal

locate
secure

Abstractions
search

Realizations
Our Goal

locate

service

send

secure

XML
Fundamental Problems

- Mismatch between the high-level intent reflected in the descriptions of applications and their low-level implementation details

- Concept assignment problem
  - to identify how high-level concepts are associated with their implementations in source code

```python
s = socket.socket(proto, socket.SOCK_DGRAM)
s.sendto(teststring, addr)
buf = data = receive(s, 100)
while data and '\n' not in buf:
data = receive(s, 100)buf += data
```
Example Programming Task

Write an application to record musical instrument data to a file in the MIDI file format.
What Search Engines Do

Keyword

match

descriptions of apps

Record midi file

match

This program captures MIDI data...

app₁

...
What Search Engines Do

keyword match descriptions of apps

app₁

...
What Search Engines Do

Keyword match descriptions of apps

The Vocabulary Problem
What Search Engines Do

Keyword match descriptions of apps

The Vocabulary Problem
Poorly Described Applications

• Many application repositories are polluted with poorly functioning projects.

• Matches between keywords from the queries with words in the descriptions of the applications do not guarantee that these applications are relevant.
How Does It Work Now?

- Download application.
- Locate and examine fragments of the code that implement the desired features.
- Observe the runtime behavior of this application to ensure that this behavior matches requirements.

This process is manual since programmers:

- study the source code of the retrieved applications
- locate various **API calls**
- read information about these calls in help documents

Still, it is difficult for programmers to link high-level concepts from requirements to their implementations in source code.
Exemplar uses help documents to produce the names of the API calls in return to user queries thereby expanding these queries. The richness of these vocabularies makes it more likely to find matches, and produce different API calls. If some help document does not contain a desired match, some other document may yield a match.

How Does Exemplar Work?
How Exemplar Works

Exemplar uses help documents to produce the names of the API calls in return to user queries thereby expanding these queries. The richness of these vocabularies makes it more likely to find matches, and produce different API calls. If some help document does not contain a desired match, some other document may yield a match.
Exemplar uses help documents to produce the names of the API calls in return to user queries thereby expanding these queries. The richness of these vocabularies makes it more likely to find matches, and produce different API calls. If some help document does not contain a desired match, some other document may yield a match.
Search widely used library API documents. These documents contain rich vocabularies -> more likely to find right match
How Exemplar Works

**“midi”**

**“Obtains a MIDI IN receiver”**

MidiDevice.getReceiver() ➔ MidiQuickFix
javax.sound.midi.MidiDevice.getReceiver()  
... Obtains a MIDI IN receiver through which the MIDI device may receive MIDI data ...

javax.imageio.ImageWriter.write()  
... Appends a complete image stream containing a single image ...

java.awt.geom.AffineTransform.getScaleY()  
... scaling element (m11) of the 3x3 affine transformation matrix ...

AffineTransform.getScaleY()  
AffineTransform.createInverse()  
Jazilla

ShortMessage.ShortMessage()  
MidiDevice.getReceiver()  
MidiEvent.MidiEvent()  
Tritonus
Query Expansion

- Reduce this query/document mismatch by expanding the query with keywords that have a similar meaning to the set of relevant documents.

- New keywords come from help documents.

- Initial query is expanded to include the names of the API calls whose semantics unequivocally reflects specific behavior of the matched applications.
Solving An Instance of the Concept Assignment Problem

• API calls from help documents are linked to their locations in the applications source code.

• Programmers can navigate directly to these locations and see how high-level concepts from queries are implemented in the source code.
Intuition For Ranking

- More directly matched words -> higher ranking
- More API calls used -> higher ranking
  - Since API calls implement high-level concepts, more implemented concepts mean that the application is more relevant
- If API calls are connected using a dataflow -> higher ranking
Three Ranking Scores

Word Occurrences Score (WOS)

Relevant API Calls Score (RAS)

Dataflow Connections Score (DCS)

“midi”

Exemplar ranks applications higher when their descriptions contain keywords from the query.

An application’s RAS score is raised if it makes more calls to relevant methods in the API.

If two relevant API calls share data in an application, Exemplar ranks that application higher.

“record midi file”

String dev = getDevice();
String buf[] = A.readMidi(msg);
B.write(buf);
Hang In There, A Demo Is Coming
Experiment

To compare Exemplar and Sourceforge

• We need input from participants, there is no way to do it automatically

We follow a standard IR strategy for evaluation of search engine

• We use search engines that use equivalent large-scale code repositories
Structure of The Experiment

Participants were given tasks

- A short description of an application or some feature

Participants choose keywords that describe this task best

- Selecting keywords is their choice

Using search engine participants find and evaluate applications and rank them using their judgments

- Their evaluations are based on their confidence that they obtain by evaluating the source code of retrieved applications
Ranking

1. Completely irrelevant – there is absolutely nothing that you can use from this retrieved project, nothing in it is related to your keywords. The project may not even be uploaded to Sourceforge, only its description exists.

2. Mostly irrelevant – only few remotely relevant code snippets or API calls in the project.

3. Mostly relevant – a somewhat large number of relevant code snippets or API calls in the project.

4. Highly relevant – you are confident that you can reuse code snippets or API calls in the project.
## Experimental Design and Results

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Group</th>
<th>Search Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Magenta</td>
<td>Exemplar with connectivity</td>
</tr>
<tr>
<td></td>
<td>Green</td>
<td>Sourceforge</td>
</tr>
<tr>
<td></td>
<td>Yellow</td>
<td>Exemplar with API calls, no connectivity</td>
</tr>
<tr>
<td>2</td>
<td>Magenta</td>
<td>Exemplar with API calls, no connectivity</td>
</tr>
<tr>
<td></td>
<td>Green</td>
<td>Exemplar with connectivity</td>
</tr>
<tr>
<td></td>
<td>Yellow</td>
<td>Sourceforge</td>
</tr>
<tr>
<td>3</td>
<td>Magenta</td>
<td>Sourceforge</td>
</tr>
<tr>
<td></td>
<td>Green</td>
<td>Exemplar with API calls, no connectivity</td>
</tr>
<tr>
<td></td>
<td>Yellow</td>
<td>Exemplar with connectivity</td>
</tr>
</tbody>
</table>
Thirty Nine Participants

- 26 participants are Accenture employees who work on consulting engagements as professional Java programmers for different client companies.
- Remaining 13 participants are graduate students from the University of Illinois at Chicago who have at least six months of Java experience.
- 17 had programming experience with Java ranging from 1 to 3 years.
- 22 participants have more than 3 years of Java experience.
- 11 participants reported prior experience with Sourceforge.
- 18 participants reported prior experience with other search engines.
- 11 said that they never used code search engines.
- 26 participants have bachelor degrees and 13 have master degrees in different technical disciplines.
Interesting Fact – The Cost of This Study

- Professional experienced programmers are very expensive, they charge more than $50 per hour
- Accenture rate is $150 per hour
  - $26 \times 150 \times 8 = $31,200
- Additional costs run for close to $10K
  - Renting laptops with preinstalled images
  - Conference room with internet access
  - Various expenses
- Total cost is around $40,000
Rejected Null Hypothesis

- The primary null hypothesis is that there is no difference in the numbers of Cs and Ps between participants who ranked results for Sourceforge versus Exemplar search engines.

- An alternative hypothesis to $H_0$ is that there is statistically significant difference in the numbers of Cs and Ps between participants who ranked results for Sourceforge versus Exemplar search engines.
Precision
Conclusions

- Exemplar is effective in the solution domain where it helps developers to find applications that contain relevant code fragments with API calls.

- Exemplar is available at [www.xemplar.org](http://www.xemplar.org)

- Exemplar is currently used by different programmers from all over the world.
Exemplar Users and Developers

Write something...

Mark Grechanik

Can businesses be built around code search engines?

View Members

Members Admins Not Yet Replied
Blocked Requested

Kevin Sullivan

Alessio Di Stasio

Vijay Dheeraj Reddy Mandadi

Tao Xie

Shyam Sunder Santoshi

Palak Jain

Make Admin

Make Admin

Make Admin

Make Admin

Make Admin

Make Admin

Close
Thank you!

Questions?

The user enters a high-level query.

http://www.xemplar.org/
The search returns a list of projects, their descriptions, and their scores.

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Relevance Score</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MidiQuickFix</td>
<td>100%:45.59%</td>
<td>MidiQuickFix allows you to directly edit the events in a Midi file. It is intended to make it easy to find and fix problems, such as setting volume and pan values for a track, without the need for a complex Midi sequencing program.</td>
</tr>
<tr>
<td>Saiph</td>
<td>100%:30.71%</td>
<td>Java-based (multiplatform) tool for algorithmic musical composition. Saiph generates sequences made of tracks made of segments with musical events, currently notes and MIDI controllers. It supports MIDI and MusicXML file output.</td>
</tr>
<tr>
<td>PJLMidiParser</td>
<td>100%:0%</td>
<td>PJLMidiParser provides efficient parsers, written in Java, for MIDI files. It is like XML SAX parsers in that it is event-driven; the parsing is initiated and then triggers callback handlers in response to events in the MIDI file.</td>
</tr>
<tr>
<td>Tritonus</td>
<td>0%:100%</td>
<td>Tritonus is an independent implementation of the Java Sound API (<a href="http://www.javasoft.com/products/java-media/sound/index.html">http://www.javasoft.com/products/java-media/sound/index.html</a>).</td>
</tr>
<tr>
<td>TuxGuitar</td>
<td>0%:82.8%</td>
<td>TuxGuitar is a multitrack guitar tablature editor and player written in Java-SWT. It can open GuitarPro, PowerTab and TabEdit files.</td>
</tr>
</tbody>
</table>
The programmer can view a list of API calls and their locations within projects.

<table>
<thead>
<tr>
<th>File</th>
<th>Line No</th>
<th>API Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3.0/tritonus-0.3.0.tar.gz/tritonus-0.3.0/src/javaz/sound/midi/MetaMessage.java</td>
<td>93</td>
<td>javax.sound.midi.MetaMessage.MetaMessage</td>
</tr>
<tr>
<td>0.3.0/tritonus-0.3.0.tar.gz/tritonus-0.3.0/src/javaz/sound/midi/Track.java</td>
<td>54</td>
<td>javax.sound.midi.MidiEvent.getTick</td>
</tr>
<tr>
<td>0.3.0/tritonus-0.3.0.tar.gz/tritonus-0.3.0/src/javaz/sound/midi/Track.java</td>
<td>54</td>
<td>javax.sound.midi.MidiEvent.MidiEvent</td>
</tr>
<tr>
<td>0.3.0/tritonus-0.3.0.tar.gz/tritonus-0.3.0/src/javaz/sound/midi/Track.java</td>
<td>98</td>
<td>javax.sound.midi.MidiEvent.getTick</td>
</tr>
<tr>
<td>0.3.0/tritonus-0.3.0.tar.gz/tritonus-0.3.0/src/javaz/sound/midi/Sequence.java</td>
<td>79</td>
<td>javax.sound.midi.Sequence.createTrack</td>
</tr>
<tr>
<td>0.3.0/tritonus-0.3.0.tar.gz/tritonus-0.3.0/src/javaz/sound/midi/MidiSystem.java</td>
<td>60</td>
<td>javax.sound.midi.MidiSystem.getMidiDeviceProviders</td>
</tr>
</tbody>
</table>