Creating and Evolving Software by Searching, Selecting and Synthesizing ($S^3$) Relevant Source Code

Denys Poshvyvanyk, William and Mary

Mark Grechanik, Accenture Technology Labs & University of Illinois, Chicago
How Many Open Source Applications Are There?

- Sourceforge.net reports that they host 180,000 projects as of August 1, 2008.

- There are dozens of other open source repositories containing tens of thousands of different applications.

- Companies have internal source control management systems containing hundreds of thousands of applications.
Problem

• Finding/checking existing software matching high-level user requirements
  – Would reduce the cost of many software projects
  – Would provide users with examples of different implementations

• Challenges:
  – Finding relevant applications is difficult
  – Evaluating retrieved applications is difficult
What Search Engines Do

“encrypt compress XML data”
What Search Engines Do

“encrypt compress XML data”

Matching keywords

descriptions of apps

SOURCEFORGE.net
Fundamental Problems

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

• Concept assignment problem

<table>
<thead>
<tr>
<th>High level concept</th>
<th>Code snippet implementing “Send data”</th>
</tr>
</thead>
</table>
| “Send data”        | 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 |

• Many application repositories are polluted with poorly functioning projects
Working without a Tool

• Find relevant application(s)
• 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
Our Goal

"encrypt compress XML data"
Our Goal

“encrypt compress XML data”
Key observations

- While studying retrieved apps developers:
  - locate various *API calls*
  - read information about these calls in help documents
- Help docs are supplied by the same vendors whose packages/APIs are used in software
- Programmers read and rely on these API docs
- Help docs are written and reviewed by many developers
- Help documents are usually more verbose and accurate than project descriptions
How S³ System Works

- Automatically matching words in user queries against API help docs instead of:
  - searching in project descriptions;
  - searching in source code.
- S³ uses help documents to produce a list of relevant API calls.
S³ Architecture
Current Status

• Restricting the scope to Java projects
• Challenges:
  – How to automatically locate and download the latest version of the software (e.g., from sourceforge)?
  – How to automatically locate the correct entry point (i.e., main) for static analysis?
  – How to reduce the time for the static analysis?
  – How and when to update API call dictionary?
  – Testing other ranking heuristics
• Evaluation is pending (some preliminary results at the poster session)
Related Work

• CodeFinder/Helgon
• ParseWeb
• CodeBroker
• Hipikat
• Automated Method Completion (AMC)
• Strathcona
• Prospector
• XSnippet
• Google code search, Krugle,...
Conclusions & Future Work

• $S^3$ recommends/checks relevant applications based on:
  – analysis of relevant API help documents;
  – analysis of actual API calls.
• Indexing available open-source projects and pre-computing data and control flow among API calls
• Analyzing multiple releases of the same project