A Combination Approach for Enhancing Automated Traceability

Xiaofan Chen	xche044@aucklanduni.ac.nz
John Hosking	john@cs.auckland.ac.nz
John Grundy	jgrundy@swin.edu.au

Motivation

- Traceability systems extract links between documentation and source code e.g. to aid understanding and maintenance
- Key metrics:
  - Precision = correct links recovered / total recovered
  - Recall = correct retreived / total # correct
- How to extract links with both high precision and recall??

Our approach

- Limitations of typically used Vector Space Model (VSM):
  - Low precison at low cut points
  - Low recall at high cut points
  - Miss some links
- Our approach: to combining three supporting techniques with VSM: Regular Expressions, Key Phrases, Clustering

Regular Expression

- Regular Expression (RE)
  - Find all of the occurrences of class names in documents
  - Use two regular expressions, e.g. "Control.java":
    - (.*)(^a-zA-Zo-9\-)<C-?o-?n-?t-?r-?o-?l>(^a-zA-Zo-9\-)(.*)
    - (.*)(^a-zA-Zo-9\-)<each part of package name>(^a-zA-Zo-9\-)(.*)

- Expands the retrieved link sets at high cut points

Key Phrases

- Key Phrases (KP)
  - Extract key phrases from comments of code
  - Find alternative words to the class name or words indicating what tasks the class fulfills e.g. class "RefAddr" - query is "RefAddr OR ref addr OR ref OR addr"
  - Add these key phrases to VSM queries

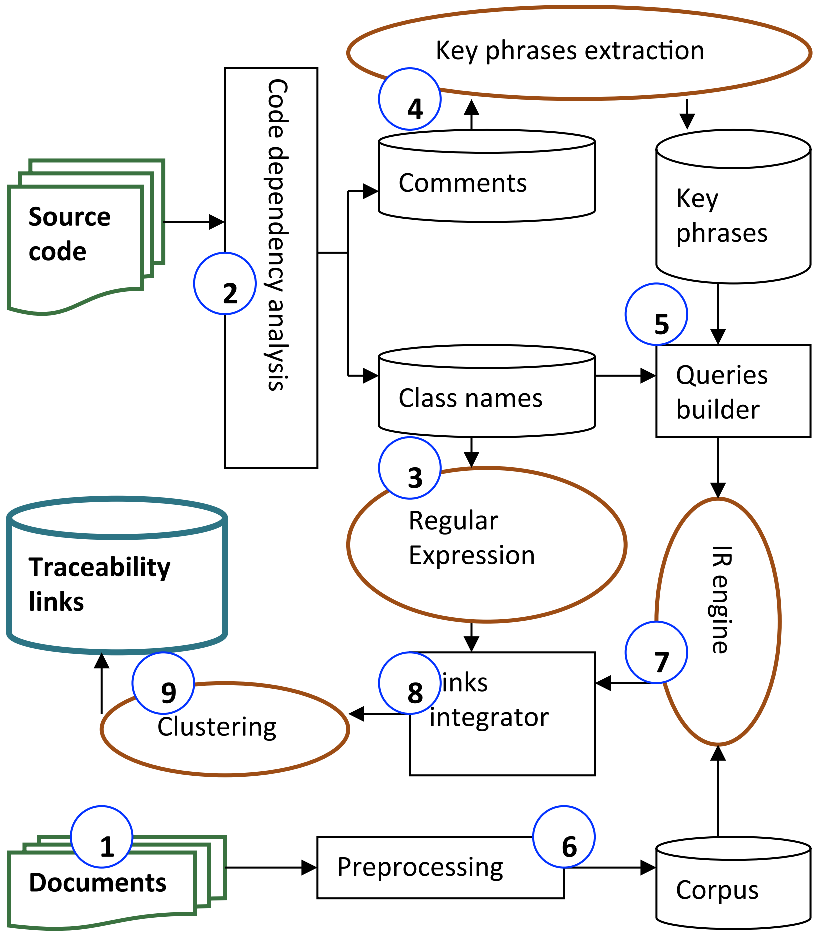
- Extracts links missed by VSM

Clustering

- Clustering
  - Utilize the inherent hierarchical structure in documents to cluster extracted traceability links
  - Use modified K-mean algorithm
    - Three main steps: initialization, assignment, and removal
- Eliminates many fault links at low cut points (though does eliminate a few true links)

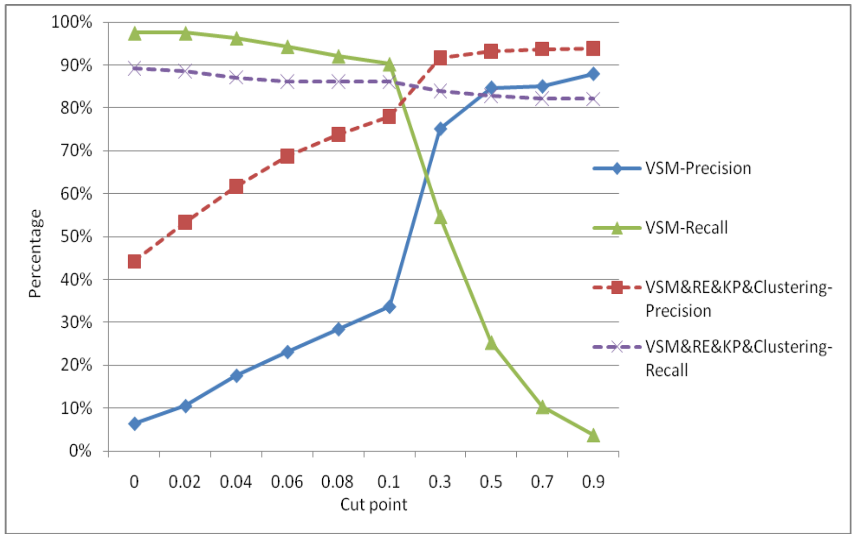
Implementation (see paper!)

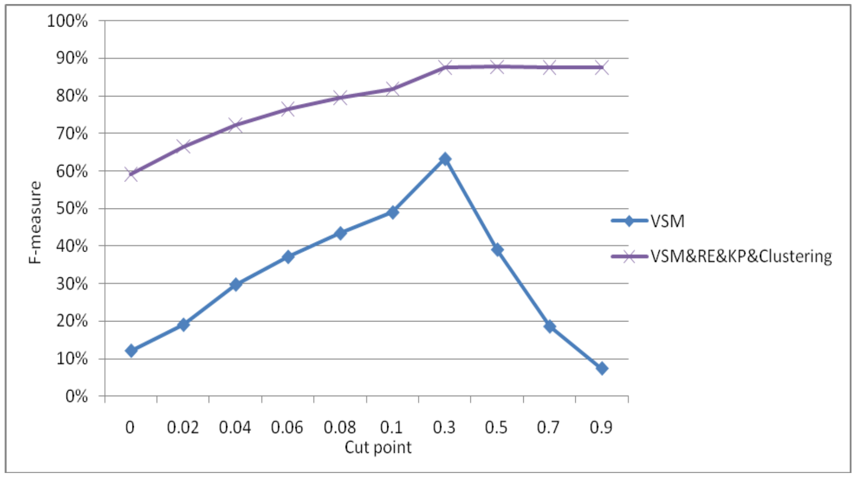
1. Divide documents into small subdocuments
2. Extract class names and comments in code
3. Find documents that directly mention class names by RE
4. Extract key phrases from comments by KP
5. Combine extracted key phrases with class names to form VSM queries
6. Preprocessing documents
7. Retrieve links by VSM (IR engineer)
8. Merge links
9. Refine extracted links by Clustering



Preliminary Expermental Results

- Case study: JDK1.5
- Top figure:
  - Precision is significantly improved
  - Recall is much higher than for VSM
- Bottom figure:
  - Our approach is more effective than VSM if precision and recall are equally important

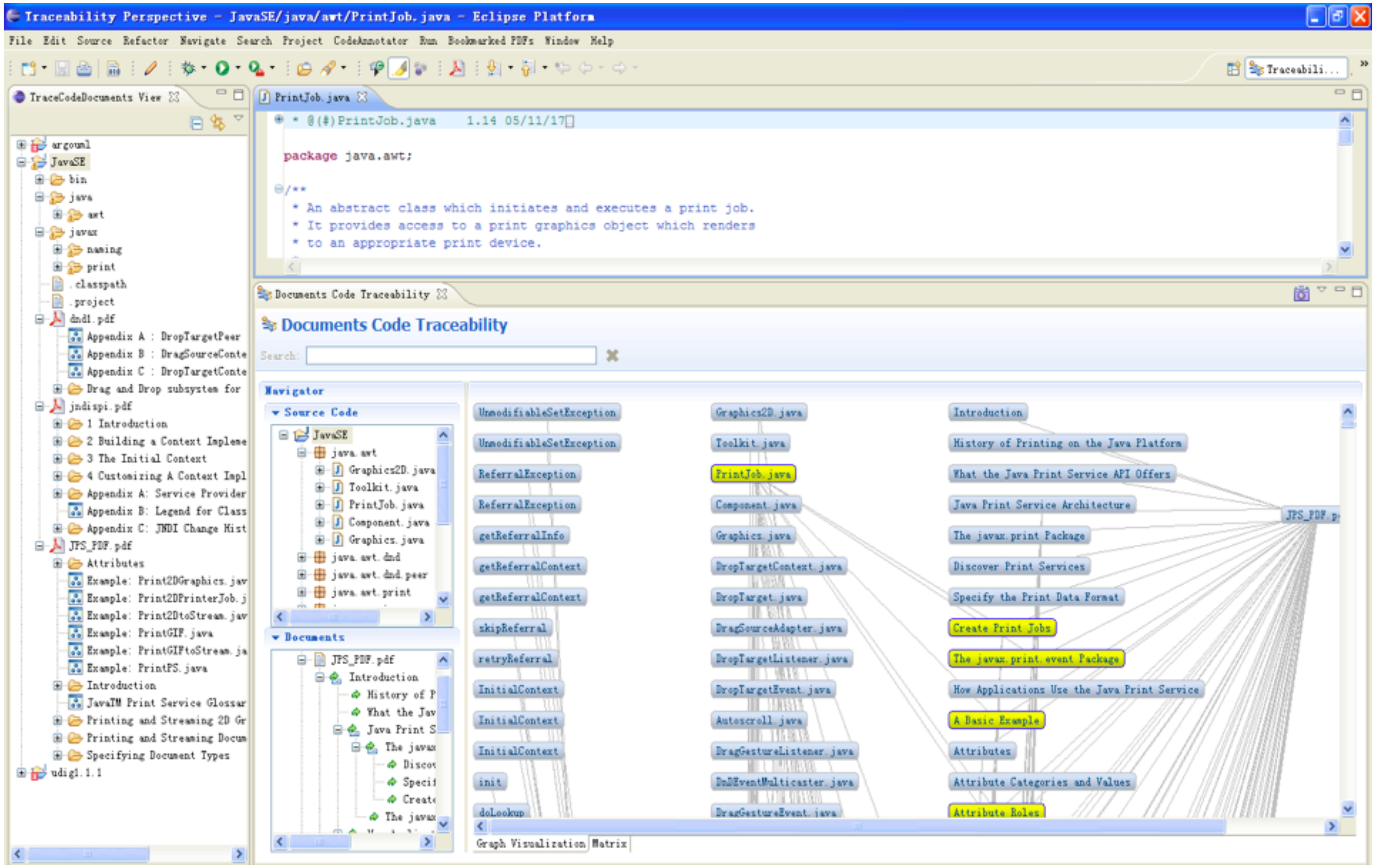




Future Work

- Further case studies – now done for JDK1.5, ArgoUML, Freenet & JMeter
- Allow users to configure thresholds, select some or all techniques to apply to extract links
- Allow users to create or edit links
- Visualization tool to visualize extracted traceability links

Example from prototype #1



```
 * @(#)PrintJob.java     1.14 05/11/17

package java.awt;

/**
 * An abstract class which initiates and executes a print job.
 * It provides access to a print graphics object which renders
 * to an appropriate print device.
```

References

1) Chen, X., Hosking, J.G., Grundy, J.C. and Amor, R. Development of robust traceability benchmarks, 2013 Australasian Conference on Software Engineering (ASWEC 2013), Melbourne, Australia, July 2013, IEEE CS Press.
2) Chen, X., Hosking, J.G. and Grundy, J.C. Visualizing Traceability Links between Source Code and Documentation, 2012 IEEE International Symposium on Visual Languages and Human-Centric Computing, Innsbruck, Austria, Sept 30-Oct 4 2012, IEEE CS Press.
3) Chen, X. and Grundy, J.C. Improving Automated Documentation to Code Traceability by Combining Retrieval Techniques, In proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering, Nov 6-10 2011, IEEE Press.
4) Chen, X., Hosking, J.G. and Grundy, J.C. A Combination Approach for Enhancing Automated Traceability, New Ideas and Emerging Results Track, In Proceedings of the 2011 International Conference on Software Engineering (ICSE2011), Honolulu, Hawaii, USA, May 21-28 2011.