Working Session:
Information Retrieval Based Approaches in Software Evolution

Andrian Marcus[1], Andrea De Lucia[2], Jane Huffman Hayes[3], Denys Poshyvanyk[1]

[1]*Department of Computer Science*
Wayne State University
Detroit, MI 48202
313 577 5408
amarcus@wayne.edu,
denys@wayne.edu

[2]*Dipart. di Matem. e Informatica*
Università di Salerno
Via ponte don Melillo,
84084, Fisciano (SA), Italy
+39 089 963376
adelucia@unisa.it

[3]*Department of Computer Science*
University of Kentucky
301 Rose Street
Lexington, KY 40506
859 257 3171
hayes@cs.uky.edu

Abstract

During software evolution a collection of related artifacts with different representations are created. Some of these are composed of structured data (e.g., analysis data), some contain semi-structured information (e.g., source code), and many include unstructured information (e.g., text). Research efforts exist that are trying to extract, represent, and analyze the unstructured information in software. Information retrieval (IR) techniques are used quite successfully in the past years to represent and extract textual information from software artifacts, with application to many maintenance tasks.

This working session will focus on the state on the art in the application of IR-based techniques to support software maintenance activities. The session aims to identify the main research and practical issues in the field, to determine future work directions, and to foster collaborations among the participants.

1. Introduction and Rationale

Software is comprised of a multitude of artifacts; some of them are intended to be read by the compiler, while many others are intended to be read by developers. This is especially true during software evolution, when developers have to deal with large software, often written by others.

The user centric information is often expressed in natural language and it is embedded in documentation and source code. This information is very important for the developers to understand a great deal of the *why* and *what* of the software system, as much as the source code is useful to understand the *how* of the software. Natural language external documentation (e.g., requirements, design documents, user manual, etc.), comments, and identifiers in the source code encode to a large degree the domain of the software and capture design decisions, change requests, developer information, etc. This unstructured information is referred to as *semantic*, as opposed to *structural*, which is expressed mainly by the source code and other data intensive artifacts, such as analysis information.

The single developer/maintainer development model did not need capturing much of this information, as the working and long term memory of the developer often sufficed to store such information. Today, the increasing size and complexity of software needs large development groups, often distributed geographically. Storing and sharing the semantic information is much needed today. More than that, given the large amount of it, tools are necessary for its storage, retrieval, and analysis, before it is delivered to the users.

2. State of the Art

In the past decade, researchers proposed information retrieval (IR) models to address these problems related to the semantic information in existing software. Early models were used to construct software libraries [13] and more recent work focused on specific software maintenance or development tasks such as:

- Traceability link recovery [1, 5, 8, 12, 15]
- Concept location [17, 19, 24]
- Software and web site modularization and reverse engineering [9, 10, 14, 21]
- Requirements engineering [3, 18]
- Software reuse [7, 13, 23]
- Impact analysis [2]
- Quality assessment and software measurement [11, 16, 20], etc.

These IR based approaches to software engineering problems differ not only in their scope, but also in their underlying indexing mechanism, corpus construction, or

data analysis method. A general model can be described with the following steps:

1. A corpus is created using the source code and other linguistic software artifacts, such as the external documentation. Various processing methods are employed in the corpus construction, some based on natural processing techniques, such as word stemming. Each document in the corpus corresponds to a specific software element, such as a file, a class, or a method.
2. An IR method is used to index the corpus, such as vector space models [22], Latent Semantic Indexing [6], Bayes classifiers, or other probabilistic models [4], etc. A semantic space of the software system is created.
3. A similarity measure between the documents in the corpus is defined and similarities are computed among the corresponding software elements. These measures are commonly referred to as semantic similarities.
4. The semantic similarities are used to solve the maintenance or development task at hand. Some approaches combine these measures with additional data extracted with structural software analysis tools, such as: dependencies, software change data, execution traces, test cases, etc.

3. Open Issues and Problems

The working session has several complementing goals. First, it aims at clearly defining the state of the art in the filed, briefly described above. As the field grows, researchers and practitioners need to agree on a common terminology, as the current work by different groups is somewhat incoherent. We need to assess how far this field came to date and how far it can go in the future.

In addition, we want to identify which issues are already answered by research and ready for practical applications and which are still open or unaddressed. Several questions will be directly addressed during the working session and many more will be raised on the spot:

- How can we refine and improve the general model, presented above? Does the model suit all current and future applications?
- Do certain IR methods suit specific software maintenance problems, or we can use any of them for any task?
- Is the field mature enough to talk about benchmarking?
- What new applications in software evolution exist for the IR-based approaches?
- What are the major practical problems with the current state of the art: efficiency, scalability, recall and precision, etc.? Are there specific problems associated with different IR methods?
- Who among the current researchers can collaborate on future projects?
- Is there available software produced by any research group? Can we initiate and maintain an open source effort in the area?
- How can we best integrate IR methods with other techniques for the analysis of unstructured information (e.g., natural language processing)? What is the trade-off?
- How can we bridge the work of the software maintenance community and other groups from areas like requirements engineering, programming languages, etc?
- Is there a need for future, organized meetings like this working session?

4. Session Format

The working session will have 90 minutes and will consist of three parts.

It will start with short interactive presentations given by some of the participants, which will be solicited in advance and selected by the organizers. These presentations will focus on existing approaches and techniques.

Following these presentations, all the participants will participate in an open brainstorming session, which will focus on identifying open issues in the field, new challenges, etc. Questions will be asked and answers provided by the participants.

The final part will be devoted to recapitulate and reiterate the unanswered items from the previous two parts and to build a roadmap for future events, research, and collaborations among the participants.

5. Expected Outcome of the Session

A website for the working session will be developed and maintained by the organizers. The discussions and presentations from the session will be summarized and publicized on the website and other appropriate venues.

We expect that this session will be the first in a succession of future events that will focus on this research area and will also include related fields.

6. References

[1] Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., and Merlo, E., "Recovering Traceability Links between Code and Documentation", *IEEE Transactions on Software Engineering*, 28, 10, October 2002, pp. 970 - 983.

[2] Antoniol, G., Canfora, G., Casazza, G., and Lucia, A., "Identifying the Starting Impact Set of a Maintenance Request:

A Case Study", *in Proceedings 4th European Conference on Software Maintenance and Reengineering (CSMR'00)*, Zurich, Switzerland, February 29 - March 03 2000, pp. 227-230.

[3] Clelang-Huang, J., Settimi, R., Duan, C., and Zou, X., "Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability", *in Proceedings International Requirements Engineering Conference (RE'05)*, Paris, France, 2005, pp. 135-144.

[4] Crestani, F., Lalmas, M., Van Rijsbergen, C. J., and Campbell, I., "Is this document relevant?…probably: a survey of probabilistic models in information retrieval", *ACM Computing Surveys*, 30, 4, 1998, pp. 528-552.

[5] De Lucia, A., Fasano, F., Oliveto, R., and Tortora, G., "Enhancing an Artefact Management System with Traceability Recovery Features", *in Proceedings IEEE International Conference on Software Maintenance (ICSM'04)*, Chicago, IL, September 11-17 2004, pp. 306-315.

[6] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R., "Indexing by Latent Semantic Analysis", *Journal of the American Society for Information Science*, 41, 1990, pp. 391-407.

[7] Frakes, W., "Software Reuse Through Information Retrieval", *in Proceedings 20th Hawaii International Conference On System Sciences (HICSS'87)*, Kona, HI, January 1987, pp. 530-535.

[8] Hayes, J. H., Dekhtyar, A., and Sundaram, S. K., "Advancing Candidate Link Generation for Requirements Tracing: The Study of Methods", *IEEE Transactions on Software Engineering*, 32, 1, January 2006, pp. 4-19.

[9] Kawaguchi, S., Garg, P. K., Matsushita, M., and Inoue, K., "Mudablue: An automatic categorization system for open source repositories", *in Proceedings the 11th Asia-Pacific Software Engineering Conference (APSEC'04)*, 2004, pp. 184-193.

[10] Kuhn, A., Ducasse, S., and Girba, T., "Enriching Reverse Engineering with Semantic Clustering", *in Proceedings IEEE Working Conference On Reverse Engineering (WCRE'05)*, Pittsburgh, PA, November 8-11 2005, pp. 113—122.

[11] Lawrie, D., Feild, H., and Binkley, D., "Leveraged Quality Assessment Using Information Retrieval Techniques", *in Proceedings 14th IEEE International Conference on Program Comprehension (ICPC'06)*, Athens, Greece, June 14-16 2006, pp. 149-158.

[12] Lormans, M. and Van Deursen, A., "Can LSI help Reconstructing Requirements Traceability in Design and Test?" *in Proceedings 10th European Conference on Software Maintenance and Reengineering (CSMR'06)*, Bari, Italy, March 12 2006, pp. 47-56.

[13] Maarek, Y. S., Berry, D. M., and Kaiser, G. E., "An Information Retrieval Approach for Automatically Constructing Software Libraries", *IEEE Transactions on Software Engineering*, 17, 8, 1991, pp. 800-813.

[14] Maletic, J. I. and Marcus, A., "Supporting Program Comprehension Using Semantic and Structural Information", *in Proceedings 23rd International Conference on Software Engineering (ICSE'01)*, Toronto, Ontario, Canada, May 12-19 2001, pp. 103-112.

[15] Marcus, A., Maletic, J. I., and Sergeyev, A., "Recovery of Traceability Links Between Software Documentation and Source Code", *International Journal of Software Engineering and Knowledge Engineering*, 15, 5, October 2005, pp. 811-836.

[16] Marcus, A. and Poshyvanyk, D., "The Conceptual Cohesion of Classes", *in Proceedings IEEE International Conference on Software Maintenance* (ICSM'05), Budapest, Hungary, September 25-30 2005, pp. 133-142.

[17] Marcus, A., Sergeyev, A., Rajlich, V., and Maletic, J., "An Information Retrieval Approach to Concept Location in Source Code", *in Proceedings 11th IEEE Working Conference on Reverse Engineering (WCRE'04)*, Delft, The Netherlands, November 9-12 2004, pp. 214-223.

[18] och Dag, J. N., Gervasi, V., Brinkkemper, S., and Regnell, B., "A Linguistic-Engineering Approach to Large-Scale Requirements Management", *IEEE Software*, 22, 1, 2005, pp. 32-39.

[19] Poshyvanyk, D., Gael-Gueheneuc, Y., Marcus, A., Antoniol, G., and Rajlich, V., "Combining Probabilistic Ranking and Latent Semantic Indexing for Feature Identification", *in Proceedings 14th IEEE International Conference on Program Comprehension (ICPC'06)*, Athens, Greece, June 14-16 2006, pp. 137-148.

[20] Poshyvanyk, D. and Marcus, A., "The Conceptual Coupling Metrics for Object-Oriented Systems", *in Proceedings 22nd IEEE International Conference on Software Maintenance (ICSM'06)*, Philadelphia, PA, September 25-27 2006, pp. to appear.

[21] Ricca, F., Tonella, P., Girardi, C., and Pianta, E., "An Empirical Study on Keyword-based Web Site Clustering", *in Proceedings 12th IEEE International Workshop on Program Comprehension (IWPC'04)*, Bari, Italy, 2004, pp. 204-213.

[22] Salton, G. and McGill, M., Introduction to Modern Information Retrival, McGraw-Hill, 1983.

[23] Ye, Y. and Fischer, G., "Supporting Reuse by Delivering Task-Relevant and Personalized Information", *in Proceedings IEEE/ACM International Conference on Software Engineering (ICSE'02)*, Orlando, FL, May 19-25 2002, pp. 513-523.

[24] Zhao, W., Zhang, L., Liu, Y., Sun, J., and Yang, F., "SNIAFL: Towards a Static Non-Interactive Approach to Feature Location", *ACM Transactions on Software Engineering and Methodologies*, 2006, pp. to appear.