ISSN 2186-7437

NII Shonan Meeting Report

No. 2016-3

Mining & Modeling Unstructured Data in Software - Challenges for the Future

Sonia Haiduc
Takashi Kobayashi
Michele Lanza
Andrian Marcus

March 7–10, 2016

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Mining & Modeling Unstructured Data in Software – Challenges for the Future

Organizers:
Sonia Haiduc (Florida State University)
Takashi Kobayashi (Tokyo Institute of Technology)
Michele Lanza (University of Lugano)
Andrian Marcus (The University of Texas at Dallas)

March 7–10, 2016

To analyze, comprehend, and reverse engineer software projects and their software development processes, developers rely on various sources of information. Bug reports, execution logs, mailing lists, code review reports, change logs, requirements documents, and the actual source code contain implicit developer knowledge about the project and past development efforts. Most of this knowledge is captured as unstructured information, that is, natural language text used to exchange information among people.

Researchers in the Information Retrieval (IR), Data Mining (DM), and Natural Language Processing (NLP) fields have experimented with various techniques (such as, Latent Dirichlet Allocation and Vector Space Model) and ad - hoc approaches to enable the mining of unstructured data from software artifacts. However, these techniques were not designed to work with the complexities and peculiarities of unstructured software engineering data, and thus are not readily applicable to the software engineering research domain.

The challenges for both researchers and practitioners are to determine the appropriate set of techniques to tackle the problem at hand and to understand how to use them effectively.

The goal of this Shonan meeting is to facilitate the cross-fertilization of three diverse research communities, namely the one on mining software repositories, the one on mining unstructured data, and the one on software summarization. We believe that at the intersection of these three research fields lies a vast and still underexplored research territory, which can only be investigated if approaches developed by the three communities and merged in a synergetic way. Given the wide range of expertise needed in the research on mining unstructured data from software artifacts, collaborations are often necessary.

The meeting aims to tackle these challenges and make mining unstructured data clear, accessible, and applicable to the software engineering domain. We achieved this via three paths:

- First, we invited peers to give a written description of their experiences with mining unstructured data, by sharing the techniques they used, the challenges they faced, and the solutions that they found successful.

- Second, we encouraged discussion and dissemination of the presented work in following extended group discussion sessions.
- Third, we organized a group discussion in the form of a panel, according to the “ fishbowl ” technique, to identify and discuss topics that are most relevant to the meeting participants. By collecting available techniques, solutions, and challenges yet to be overcome, we aim to advance the state-of-the-art in mining unstructured software engineering data.

The meeting aims to address the following topics:

- Applications of unstructured data mining techniques to support software maintenance, software reverse engineering tasks (e.g., feature location, traceability), and for enhancing software quality;
- Novel sources of unstructured data, such as mobile app stores, phone records, screenshots, interviews, or wiki pages;
- Usage of NLP, IR, and ML techniques for mining unstructured data;
- Classification and dissemination of techniques for extracting unstructured data;
- Identification of open research challenges and proposed solutions;
- Approaches for handling imperfect data, such as summarization approaches;
- Novel extractors for unstructured data and performance evaluation with respect to existing techniques;
- Linking of unstructured and structured data for richer information;
- Negative results (“ what did not work ”) when mining unstructured data, and experience reports;
- Large-Scale mining of Unstructured Data in Big Data environments;

We aim to facilitate in-depth discussions of techniques for mining unstructured data, their similarities and differences, applications in modern Data Mining, as well as potential pitfalls and problems.

Overview of Talks

Island Parsing (Tutorial)

Luca Ponzanelli and Andrea Mocci (University of Lugano, Switzerland)

Artifacts containing natural language, like Q&A websites (e.g., Stack Overflow), tutorials, and development emails, are essential to support software development, and they have become a popular subject for software engineering research. The analysis of such artifacts is particularly challenging because of their heterogeneity: these resources consist of natural language interleaved with fragments of multiple programming and markup languages. In this tutorial, I will focus on our efforts towards a systematic approach to model contents of such artifacts, enabling holistic analyses that fully exploit their intrinsic heterogeneous nature. In particular, I will illustrate our StORMeD framework (http://stormed.inf.usi.ch), and how its parsing service can be effectively used to implement a holistic summarizer for Stack Overflow discussions, that takes into account both the narrative and the structured fragments extracted and modeled from the contents.

srcML for MUD (Tutorial)

Jonathan Maletic (Kent State University, USA)

The tutorial is intended for those interested in constructing custom software analysis and manipulation tools to support research. srcML (srcML.org) is an infrastructure consisting of an XML representation for C/C++/C#/Java source code along with efficient parsing technology to convert source code to-and-from the srcML format. The briefing describes srcML, the toolkit, and the application of XPath and XSLT to query and modify source code. Additionally, a hands-on tutorial of how to use srcML and XML tools to construct custom analysis and manipulation tools will be conducted.

Finding the Meaning of Life in API Documentation

Martin Robillard (McGill University, Canada)

Learning resources are crucial for helping developers learn to use software development technologies. However, the gap between the information needs of developer and externalized knowledge to meet these needs shows no sign of closing. I will discuss the problem of automatically finding information that explains a certain technology, and discuss how far we got with a technique to find information that explains how to use API types.

Using NLP to Identify Meaningful Sentences in Informal Documentation

Christoph Treude (University of Adelaide, Australia)

Sentences on Stack Overflow or other informal documentation sites are often not meaningful on their own without their surrounding code snippets or the question that prompted a given answer. Based on a study to identify sentences from Stack Overflow that are related to a particular API type and that provide insight not contained in the API documentation of that type, we discuss a set of NLP features that can help identify sentences that are meaningful on their own, including co-occurring part-of-speech tags and the presence of the verb “to be”.

Towards Trace-Any: Interactive and Transitive Recovery of Traceability Links

Hironori Washizaki (Waseda University, Japan)

Recovering missing important links from software is the key to success of its maintenance such as specifying locations that need correction. Towards tracing any software material at any abstraction level, this talk discusses two techniques for recovering traceability links: log-based interactive recovery involving link recommendation and user feedback, and, transitive recovery by connecting different links.

Is code behaving as expected? Extracting expected behavior from natural language artefacts

Alessandra Gorla (IMEDEA, Spain)

Natural language artefacts often encode important information regarding the expected behavior of code artefacts. Analyzing natural language artefacts to extract such information and using it to check whether it matches the actual behavior of code artefacts can highlight the presence of faults or covert behavior. In this talk I will present three ideas along this line. The first one is about using Android app descriptions and comparing them against implementations to identify covert – and often malicious – behaviour. The second one has the same goals, but uses the text of UI element labels instead of app descriptions. Finally, the third one is about analyzing Javadoc comments to automatically generate test oracles to highlight faults in Java methods.

Generating Tests for Android Apps from Natural Language Bug Reports and App Reviews

Alex Orso (Georgia Institute of Technology, USA)

As confirmed by a recent survey conducted among developers of the Apache, Eclipse, and Mozilla projects, two extremely challenging tasks during maintenance are reproducing and debugging field failures – failures that occur on user machines after release. Unfortunately, the information provided by users in bug reports or, even worse, app reviews is in most cases too limited to allow for reproducing, further investigating, and ultimately understanding a failure. To help developers with these tasks, we plan to leverage program analysis and NLP techniques to (1) infer a set of step that can lead to a failure from a natural language bug report or app review, (2) match these steps to graphical elements of the app, source code elements, or both, and (3) synthesize test cases that mimic the reported field failure and can be used to debug it. Because this project is in its very initial phase, the main goal of this talk is to introduce the project’s motivation and goals, present some very preliminary results, and discuss the next steps of the work.

Extracting the Essence of Software Systems ’ Architectures through Unstructured-Data Mining

Nenad Medvidovic (University of Southern California, USA)

Engineers frequently neglect to carefully consider the impact of their changes to a software system. As a result, the software system ’ s architecture eventually deviates from the original designers ’ intent and degrades through unplanned introduction of new and/or invalidation of existing design decisions. Architectural decay increases the cost of making subsequent modifications and decreases a system ’ s dependability, until engineers are no longer able to effectively evolve the system. At that point, the system ’ s actual architecture must be recovered from the implementation artifacts. However, this is a time-consuming and error-prone process, and leaves critical issues unresolved: the problems caused by architectural decay will likely be obfuscated by the system ’ s many elements and their interrelationships – the epitome of unstructured data- thus risking further decay. In this talk I will focus on pinpointing the locations in a software system ’ s architecture that reflect architectural decay, the points in time when that decay tends to occur, and the reasons why that decay occurs. Specifically, I will present an emerging catalogue of commonly occurring symptoms of decay – architectural “ smells ” . I will illustrate the occurrence of smells identified in the process of recovering the architectures of a large number of real-world systems. I will also highlight the relationship between architectural smells and the much better understood code smells. Finally, I will touch upon several undesirable but common occurrences during the evolution of existing systems that directly contribute to decay. I will conclude by identifying a number of simple steps that engineers can undertake to stem software system decay.

Interactive Code and Knowledge Search Supported by Text Analysis

Xin Peng (Fudan University, China)

Open-source code repositories and question-and-answer websites provide a huge amount of useful code and development knowledge. However, developers often feel it hard to find required code and answers by using keyword-based search. To improve the current practice, we think it is beneficial to provide more advanced code and knowledge search support that can better capture both the intent of developers ’ search requests and the meaning of code fragments and questions. In this talk, I will introduce our past and ongoing works on interactive code and development knowledge search supported by text analysis.

Quality of Source Code Lexicon

Venera Arnaoudova (Washington State University, USA)

It has been well documented that a large portion of the cost of any software lies in the time spent by developers in understanding a program ’ s source code before maintenance, repairs, or updates can be undertaken. To understand software, developers spend a considerable amount of time reading the source code lexicon, i.e., the identifiers (names of programming entities such as classes or variables) and comments that are used by developers to embed domain concepts and to communicate with their teammates. In this talk we will review existing metrics for lexicon quality and how lexicon quality has been related to program understanding and to software quality.

Part-Of-Speech Tagging of Source Code Identifiers and Comments

Jonathan Maletic (Kent State University, USA)

An approach for using heuristics and static program analysis information to markup part-of-speech for program identifiers is presented. It does not use a natural language part-of-speech tagger for identifiers within the code. A set of heuristics is defined akin to natural language usage of identifiers usage in code. Additionally, method stereotype information, which is automatically derived, is used in the tagging process. The approach is built using the srcML infrastructure and adds part-of-speech information directly into the srcML markup.

Mining Fine-Grained Code Changes to Resolve Merge Conflicts

Katsuhisa Maruyama (Ritsumeikan University, Japan)

I believe that fine-grained code changes behind merge conflicts are useful for resolving those conflicts. My talk presents an idea for supporting such resolution using a tool that both records fine-grained code changes of Java source code and extracts a particular part of them.

Mining IDE Interaction Data

Roberto Minelli (University of Lugano, Switzerland)

Developers continuously interact with Integrated Development Environments (IDEs) while working. These interactions carry a lot of actionable information. For example, researchers recorded the navigation paths followed by developers, and used this information to support source code exploration. However, most of the potential of interaction data is largely unexplored and unused. In our research we developed DFlow, an IDE interaction profiler. We collected over 700 hours of development time recorded with DFlow. In this talk we will illustrate how to interpret and mine this novel and complex source of information. We will understand how developers spend their time inside the IDE, how they navigate source code, and how the user interface of an IDE might impact on their productivity. At the end of our talk, we will also illustrate our vision: Interaction-Aware IDEs, IDEs that improve programmers ’ productivity by leveraging all the fine-grained IDE interactions.

Linking between Unstructured Software Artifacts with Structural Flavor

Shinpei Hayashi (Tokyo Institute of Technology, Japan)

Linking different types of software artifacts, e.g., requirements-to-code, requirements-to-rationale, or code-to-code, is a key technique in software engineering, and the textual information of them are utilized for matching the artifacts. Since typical software artifacts also have behavioral and/or structural aspects even if they are based on natural language descriptions, additional consideration of such aspects is useful for a better matching. In this talk, I will show some example experiences of utilizing the (semi-)structure of textual information of software artifacts for linking them.

List of Participants

- Venera Arnaoudova, Washington State University, USA
- Daniel German, University of Victoria, Canada
- Michael Godfrey, University of Waterloo, Canada
- Alessandra Gorla, IMDEA, Spain
- Sonia Haiduc, Florida State University, USA
- Shinpei Hayashi, Tokyo Institute of Technology, Japan
- Takashi Kobayashi, Tokyo Institute of Technology, Japan
- Nicholas Kraft, ABB Research, USA
- Michele Lanza, University of Lugano, Switzerland
- Jonathan Maletic, Kent State University, USA
- Andrian Marcus, University of Texas at Dallas, USA
- Katsuhisa Maruyama, Ritsumeikan University, Japan
- Collin McMillan, University of Notre Dame, USA
- Nenad Medvidovic, University of Southern California, USA
- Marija Mikic-Rakic, Google, USA
- Roberto Minelli, University of Lugano, Switzerland
- Andrea Mocci, University of Lugano, Switzerland
- Vincent Ng, University of Texas at Dallas, USA
- Masao Ohira, Wakayama University, Japan
- Alessandro Orso, Georgia Institute of Technology, USA
- Xin Peng, Fudan University, China
- Martin Pinzger, University of Klagenfurt, Austria
- Luca Ponzanelli, University of Lugano, Switzerland
- Martin Robillard, McGill University, Canada
- Christoph Treude, University of Adelaide, Australia
- Hironori Washizaki, Waseda University, Japan
- Thomas Zimmermann, Microsoft Research, USA

Meeting Schedule

Check-in Day: March 6(Sun)

- Welcome Banquet

Day1: March 7 (Mon)

- 09:00–10:30 - Introduction
- 10:30–11:00 - Coffee Break
- 11:00–12:30 - Tutorial 1 : “Island Parsing”
- 12:30–14:00 - Lunch
- 14:00–15:30 - Talks and Fishbowl conversation 1: *Code inside the MUD*
- 15:30–16:00 - Coffee Break
- 16:00–17:30 - Talks and Fishbowl conversation 2: *MUD outside the Code*

Day2: March 8 (Tue)

- Group Photo Shooting
- 09:00–10:30 - Talks and Fishbowl conversation 3: *MUD inside the Code*
- 10:30–11:00 - Coffee Break
- 11:00–12:30 - Tutorial 2 : “srcML for MUD”
- 12:30–14:00 - Lunch
- 14:00–15:30 - Talks and Fishbowl conversation 4: *Other MUDdy stuff*
- 15:30–17:30 - Walk and Talk – informal session during hiking –

Day3: March 9 (Wed)

- 09:00–10:30 - Breakout Sessions 1
- 10:30–11:00 - Coffee Break
- 11:00–12:30 - Breakout Sessions 2
- 12:30–14:00 - Lunch
- 14:00–22:00 - Excursion (Kamakura) & Dinner at Misaki-kan Hon-ten featuring Maguro.

Day4: March 10 (Thu)

- 09:00–11:30 - Breakout summaries & + Wrap up
- 12:00–14:00 - Lunch