Semantic Concept Based Retrieval of Software Bug Report with Feedback

Tao Zhang, Byungjeong Lee, Hanjoon Kim, Jaeho Lee, Sooyong Kang, and Ilhoon Shin

***Abstract*—Mining software bugs provides a way to develop reliable software. Developers can improve software quality by retrieving, analyzing and fixing software bugs. In this paper we present a technique for semantic concept based retrieval of software bug report. The technique combines folksonomy, keyword and facet-based retrieval methods satisfying developers and users' need. The technique improves the efficiency of software bug report retrieval by applying semantic concepts. Developers are likely to find the reason of software failure and fix the bugs by using this technique. Finally, we provide a case study to show the feasibility of our technique.**

***Index Terms*—Software mining, bugs classification, semantic concepts, folksonomy.**

I. Introduction

Software quality and development productivity have been considered important. Moreover, they recently became more critical issues as many developers and technologies are involved in constructing software systems. Software bug [1], [2] is a fault or defect in the software. Bug repositories contain information about software failure including how the failure occurred and how it was fixed. If appropriate software bug report can be retrieved to fix them in the development, software quality and productivity may be improved.

To achieve the task, software bug information have been extracted and predicted from history [3], [4]. Semantic web has been applied to provide an enhanced interface for bug resolution message and assess the quality of related software artifacts [5], [6]. Some studies have proposed a generic interface and data structure to handle trouble/bug management [7], [8]. Unified data model was used to support semantic bug search [7]. If they consider developers and users' feedback, the quality of bug search is likely to be improved.

For these reasons, we propose a software bug report retrieval technique which employs a semantic concept based classification [9]. Developers and users retrieve appropriate software bug data by entering keywords. This technique utilizes semantic concepts to improve the accuracy of retrieval. Moreover, to satisfy developers and users' need, the system allows them to submit the feedback information

This technique combines keyword search and folksonomy [10] search for software bugs retrieval. As a popular classification technique, folksonomy allows users to label the documents on the web according to their meaning. Even if there are many advantages as folksonomy, the major drawback is a lack of semantic information. In order to get rid of this disadvantage of folksonomy, we apply a semantic concept model to enhance semantic analysis.

Our paper is organized as follow: Section II presents related work. Section III describes the processing of software bug data retrieval system. Section IV presents an algorithm of semantic concept model-based technique. In Section V, we demonstrate the feasibility of the retrieval system by the experiment. Finally, we summarize our work and introduce future work in Section VI.

II. Related Work

In recent studies, web-based bug tracking systems were developed to offer large archives of useful troubleshooting advice. A semantics-based bug search systems [7], [11] have been proposed to implement retrieval of software bugs. The system took advantage of the semi-structured data found in widely used bug tracking systems. In the work, the authors describe how to crawl bug tracking system and to extract data and apply a multi-vector representation (MVR) to bug reports to enable semi-structured bug data search on the bug database. The semantics-based bug search system has gotten the better result because of using semantics. A unified data model to store bug tracking data has been derived from the analysis of the most popular systems. The data model has already defined classes and properties that can be used to produce an ontology in the RDF schema language. The crawled data was fed into a semantic search engine.

An enhanced semantic interface to bug resolution messages in Dhruv [5] takes users to cross-links page, which provides further detail on the clicked term. Also, a number of message recommendations of people, source files, and bug reports are provided. Dhruv determines these recommendations by taking into account the semantic cross-links of each term in the message.

III. Retrieval of Software Bug Data

As described in Section I, with our retrieval technique, developers and users can find appropriate software bug data and related solutions. In order to enhance the accuracy of

Manuscript received October 15, 2012; revised March 20, 2013. This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2011-0026461).

T. Zhang and B. Lee are with the School of Computer Science, University of Seoul, Seoul, Korea (e-mail: kerryking@ ieee.org, bjlee@uos.ac.kr).

H. Kim and J. Lee are with the Department of Electrical and Computer Engineering, University of Seoul, Seoul, Korea (e-mail: khj@uos.ac.kr, jaeho@uos.ac.kr).

S. Kang is with the Division of Computer Science and Engineering, Hanyang University, Seoul, Korea (e-mail: sykang@hanyang.ac.kr).

I. Shin is with the Department of Electronics and Information Engineering, Seoul National University of Science & Technology, Seoul, Korea (email: ilhoon.shin@snut.ac.kr).

DOI: 10.7763/IJIMT.2013.V4.421

retrieval, it is important to consider semantic concept model and get rid of ambiguity of tags. Thus, our system is designed to satisfy the following requirements:

- Goal: By providing high-quality tags, developers and users can retrieve related software bugs and solutions.
- Utility: Since different users have different perspectives, the system should adapt to users' interests.
- Lightweight: The system should require developers and users to do litter work to get high-quality result of software bugs retrieval.
- Robustness: Even if the number of users' varieties is large, the robustness of the system is still maintained.

In order to implement the above the requirements of the system, we apply a semantic concept model based technique to recommend high-quality tags to developers and users. Fig. 1 shows a processing of the system.

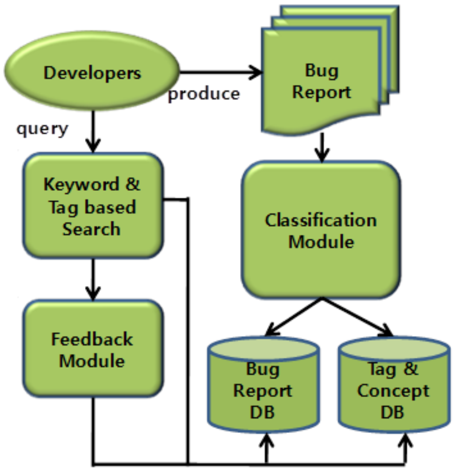


Fig. 1. System processing.

At the beginning of retrieval process, Developers can produce software bug and its solution report and label it by adding a tag. Similar tags are clustered by semantic analysis. If users input a query which includes the keyword and the facet information, our sys-tem recommends related tags. Developers and users can rate the recommended tags to improve the quality of tag recommendation. Finally, developers get the software bug and its solution that they want by submitting recommended tags. The detailed description about the modules of the proposed system is following:

Classification Module: This module is used to cluster related tags into categories. After developers created a new software bug report which includes a bug description and related solution and label it by tags, ac-cording to the clustering algorithm, the tags will be grouped into clusters.

Keyword & Tag based Search: This module is the core of our system. In this module, we apply semantic concept model. Actually, the semantic concept model is a hierarchical structure including tags and related se-mantic concept. The semantic concept model describes the relationship between tags and related semantic concepts. Thus, it helps users and developers retrieve related software bugs description and solution information. If user chooses an appropriate semantic concept, our system shows a result list of software bugs.

User feedback: After our system shows the results list of software bugs and related solutions, users can rate related solutions and provide the comments. This module manages and stores users' feedback information. The module records user information in order to improve the performance of the retrieval system.

IV. Semantic Concept Based Technique

A. *Keyword Based Retrieval*

If developers only enter appropriate keyword and facet information, the related software bug report should be showed. If the information developers input is the same as the data of database, our retrieval system shows directly developers related bug report.

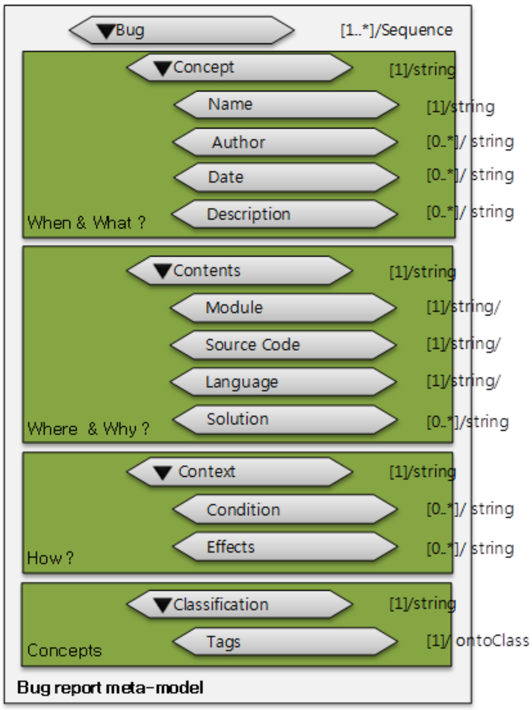


Fig. 2. Bug report meta-model.

We propose 4C meta-model for the bug report, which has been extended from 3C model [12] (Fig. 2). The bug meta-model has concept, contents, context, and classification properties. This model must be simple and easy to use for semantic search. The concept property indicates basic information such as bug name, author, date and description. The contents property represents software information including the bug such as module, source code, language and solution. The context property describes a situation which the bug occurred within including condition and effects. The classification property shows tags labeled by developers and users such as type information. Other information can be added to each property if necessary.

B. *Folksonomy*

Keywords users input to label resources are called tags. Developers can use any tag to notate the software bugs. Tags are clustered to classify bugs.

Fig. 3 shows an example of tagging. In this example, the frequency of using bug "Slowdown" to label module "Process Scheduler" is the highest (8 times) and the frequency of using tag "Process Terminated" to label the module is the second highest (6 times). Thus, "Slowdown" and "Process Terminated" are classified into the same category with respect to module "Process Scheduler". Other bugs are also classified into a category with respect to related modules like this.

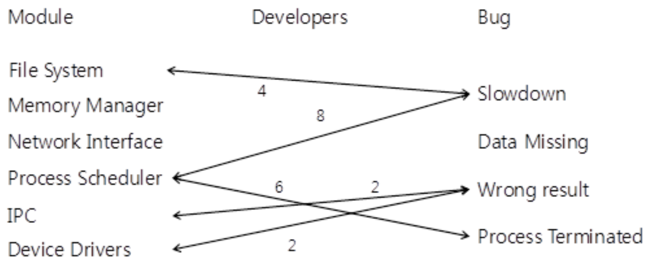


Fig. 3. An example of software tagging.

C. Semantic Concept Model

We apply semantic concept model to eliminate tag ambiguity. Thus, the accuracy of tag recommendation can be reinforced. In this study, the idea of concept network [9] is used to share the common understanding of semantic concepts to describe software bugs. So in our system, semantic concept model consists of tags and related semantic concepts. Formally, we have the following definition.

Definition 1: (i) The tag hierarchy is a part of semantic concept model. (ii) Let $t_1, t_2..., t_k$ be the sequence of tags, with their semantic concepts. (iii) The tag hierarchy consists of tag clusters and related semantic concepts.

Definition 2: In order to implement tag clustering, it is necessary to compute the similarity of tags. Given "sim (t_i, t_j)" as the similarity value of tags and tag t_i, t_j are represented by the vector $\vec{V_i}, \vec{V_j} \in R^n$. Thereinto, "sim ($t_i$, t_j)" is computed by popular cosine similarity method [13]:

$$\mathrm{sim}(t_i, t_j) = \cos(\vec{v_i}, \vec{v_j}) = \frac{\vec{v_i} * \vec{v_j}}{\|\vec{v_i}\| * \|\vec{v_j}\|}$$

$$= \frac{\sum_{k=1}^{n} w_{ki} \times w_{kj}}{\sqrt{\sum_{k=1}^{n} w_{ki}^2} \times \sqrt{\sum_{k=1}^{n} w_{kj}^2}}$$

where w_{ki} is the weight of tag t_i added to module m_k.

The value of w_{ki} is defined by TF-IDF[13]:

$$w_{ki} = tf_{ki} \times \log\frac{N}{n}$$

where tf_{ki} stands for the frequency of tag t_i added to module m_k, N is the total number of tag frequency in database, and n is the number of modules where tag t_i occurs at least one.

Fig. 4 describes an example of semantic concept model. For example, concept Control Logic Error is related to tag cluster Control Statement Error including if, for, and while statement errors and concept Memory Management Error is related to tag cluster Memory Statement Error including Array Index, Allocation, and Free errors. Two concepts are related to each other in the figure. Actually, these semantic concepts explain different bugs of software program. By using cosine similarity method to measure the similarity of tags, tags are clustered and form the semantic concept model [14].

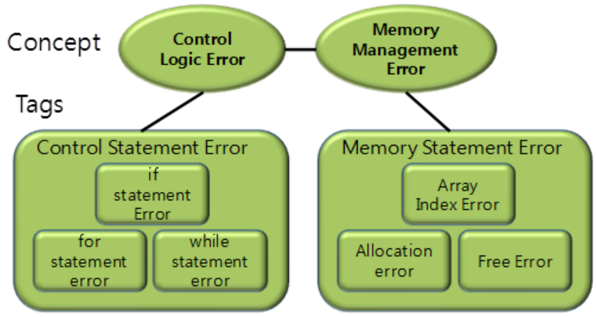


Fig. 4. An example of semantic concept model.

D. Users' Feedback

As described in the previous section, a feedback module is used to allow users to rate appropriateness of software bug report and provide the comments. Feedback is presented to reflect users' interests. It helps the system improve the quality of retrieval.

User's feedback information includes the rating-score that user gives the software bug, the average score of rated software bug and the users' comments. Section V will describe how to submit the feedback information.

V. Case Study

In order to demonstrate the feasibility of the technique based on semantic concept model, we prototyped a software bug report retrieval system. Fig. 5 shows a screenshot of the software bug retrieval in our system. The figure shows that a developer enters a keyword "email management" and related facet information. When the developer clicks search button to search related bugs, our system recommends appropriate tags and related semantic concepts. Actually, the semantic concepts are bug types of the module. In order to implement tag clustering algorithm and consider inserting further tags, we set the similarity threshold (θ) to 0.85 in the experiment [9].

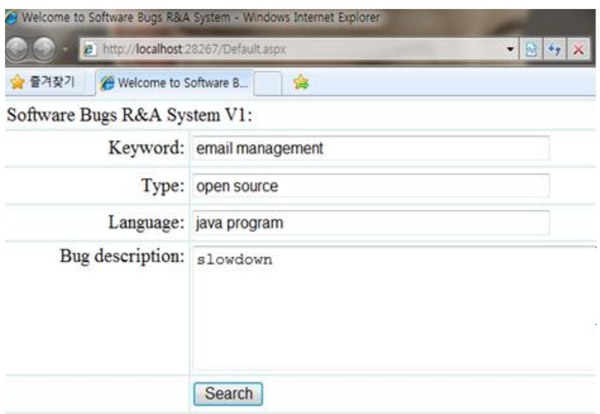


Fig. 5. Process of software bugs retrieval.

As described in Section IV, Fig. 6 shows recommended semantic concepts and modules. The developer can choose any module and appropriate concept. In this example, the developer chose a module "Mail Manage" and related semantic concept "Data error" which is used to describe the software bug types of "Mail Manage".

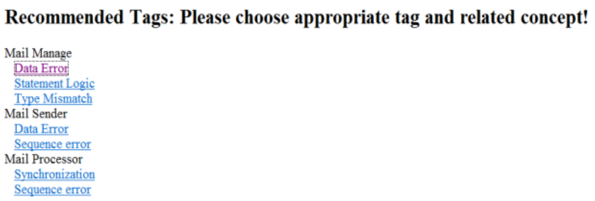


Fig. 6. Tag recommendation.

Fig. 7 shows a result list of software bugs when a developer chooses related module and semantic concept. Five bug reports have been ranked according to user score. Each report includes information about the software bug report, such as program name, type, language, etc. A developer can choose any bug item to check the details of the bug report.

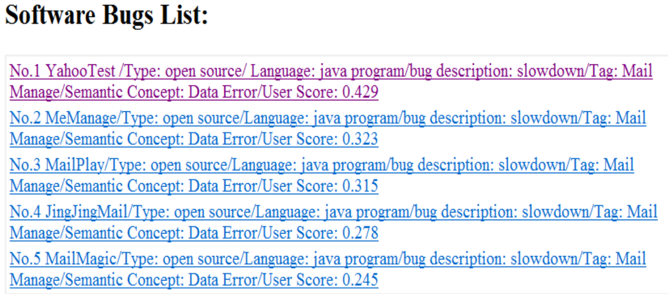


Fig. 7. Result list of software bugs.

In Fig. 8, the detailed description of a software bug report is displayed to a developer. The report shows the details of Program "Yahoo Test". Among the details, "User Score" stands for the average score of users' feedback and "Solution" stands for the debugging method of this bug. In the software bug report, a developer can give the feedback score from 1 to 5 and related comments to improve the quality of software bug report retrieval.

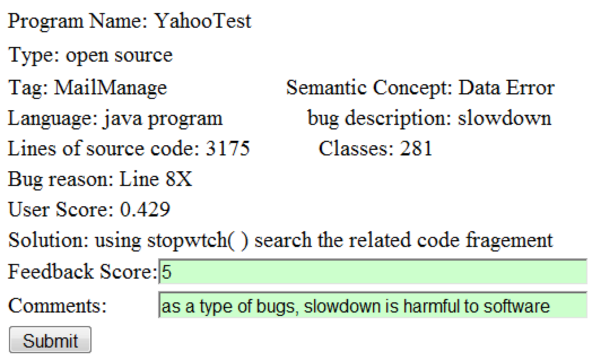


Fig. 8. Software bug report.

VI. Conclusion

In this paper, we applied a semantic concept based technique with feedback to retrieve software bug in-formation. In this technique, developers can get better result by entering a keyword and the facets. Developers produce new software bug report and label it by adding a tag. In order to improve the efficiency of software bug retrieval, semantic concept model has been designed to remedy the defects of keyword-based search. Furthermore, our technique recommends related tags to developers and users.

In the future, we will study a re-rank algorithm to rank related software bug reports. We think that the algorithm will help further improve the quality of our retrieval system. In addition, it is necessary to investigate more features of software bugs to supplement the bug reports and extend the set of semantic concepts.

References

[1] E. J. Weyuker, R. M. Bell, and T. J. Ostrand, "We're Finding Most of the Bugs, but What are We Miss-ing?" in *Proc. International Conference on Software Testing*, pp. 313 - 322, 2010.

[2] S. K. Lukins, N. A. Kraft, and L. H. Etzkorn, "Source Code Retrieval for Bug Localization Using Latent Dirichlet Allocation," in *Proc. Working Conference on Re-verse Engineering*, pp. 155-164, 2008.

[3] V. Dallmeier and T. Zimmermann, "Extraction of Bug Localization Benchmarks from History," in *Proc. IEEE/ACM International Conference on Automated Software Engineering*, pp. 433 - 436, 2007.

[4] T. Zimmermann, N. Nagappan, and A. Zeller, "Predicting Bugs From History," *Software Evolution*, Springer, 2008, ch. 4, pp. 69 - 88.

[5] A. Ankolekar, K. Sycara, J. Herbsleb, R. Kraut, and C. Welty, "Supporting Online Problem-Solving Communities with the Semantic Web," in *Proc. International Conference on World Wide Web*, pp. 575 - 584, 2006.

[6] P. Schuegerl, J. Rilling, and P. Charland, "Enriching SE ontologies with bug report quality," in *Proc. International Workshop on Semantic Web Enabled Software Engineering*, 2008.

[7] H. M. Tran, C. Lange, G. Chulkov, J. Schonwalder, and M. Kohlhase, "Applying Semantic Techniques to Search and Analyze Bug Tracking Data," *Journal of Network and Systems Management*, vol. 17, no. 3, pp. 285-308, 2009.

[8] M. Langer and M. Nerb, "Defining a trouble report format for the seamless integration of problem management into customer service management," in *Proc. Workshop of the OpenView University Association*. 1999.

[9] H. J. Kim, S. J. Lee, B. J. Lee, and S. Y. Kang, "Build-ing Concept Network-based User Profile for Per-sonalized Web Search," in *Proc. International Conference on Computer and Information Science*, pp. 567 - 572, Aug. 2010.

[10] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme, "Information Retrieval in Folksonomies: Search and Ranking," *Lecture Notes in Computer Science*, vol. 401, pp. 411-426, 2006.

[11] H. M. Tran, G. Chulkov, and J. Schönwälder, "Crawling Bug Tracker for Semantic Bug Search," *Lecture Notes in Computer Science*, vol. 5273, pp.55-68, 2008.

[12] S. Edwards, "The 3C Model of Reusable Software Components," in *Proc. Third Annual Workshop: Methods and Tools for Reuse*, 1990.

[13] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison Wesley, 1999.

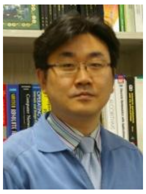
[14] T. Zhang, B. Lee, H. Kim, and S. Kang, "Utilizing feedback based Complementary Classification Techniques for Retrieval of Software Require-ments," in *Proc. Korea Conference on Software Engineering*, vol. 12, no. 1, pp. 419 - 425, 2010.

Tao Zhang received his B.S and M.E. degrees from Northeastern University, China, in 2005 and 2008, respectively. His research interests include software maintenance and evolution, data mining, grid computing, wireless network and so on. Current, he is a Ph.D. candidate in Computer Science at the University of Seoul. Now he is a graduate student member of IEEE and ACM.

Byungjeong Lee received his B.S., M.S., and Ph.D. degrees in Computer Science from Seoul National University, Korea, in 1990, 1998, and 2002, respectively. He was a research member at the Software Lab. of Hyundai Electronics Corp. from 1990 to 1998. Currently, he is an associate professor of Computer Science at the University of Seoul, Korea. His current research interests include software engineering, software evolution, and web engineering.

Hanjoon Kim received the B.S. and M.S. degrees in Computer Science and Statistics from Seoul National University, Seoul, Korea, in 1994 and 1996 and the Ph.D. degree in Computer Science and Engineering from Seoul National University, Seoul, Korea, in 2002, respectively. He is currently an associate professor at the School of Electrical and Computer Engineering, University of Seoul, Korea. His current research interests include text mining, machine learning, and intelligent information retrieval.

Jaeho Lee received the B.S. and the M.S. degrees in Computer Science from Seoul National University, Seoul, Korea, in 1985 and 1987, respectively and the Ph.D. degree in Computer Science and Engineering from the University of Michigan, Ann Arbor, Michigan, USA, in 1997. He joined the faculty at the University of Seoul in 1998. His interests are in artificial intelligence, intelligent service robots, and autonomous systems.



Sooyong Kang received his B.S. degree in mathematics and the M.S. and Ph.D. degrees in Computer Science, from Seoul National University, Seoul, Korea, in 1996, 1998, and 2002, respectively. He was then a Postdoctoral researcher in the School of Computer Science and Engineering, SNU. He is now with the Division of Computer Science and Engineering, Hanyang University, Seoul. His research interests include Operating System, Multimedia System, Storage System, Flash Memories and Next Generation Nonvolatile Memories.

Ilhoon Shin received the B.S., the M.S., and the Ph.D. degrees in Computer Science and Engineering from Seoul National University, Seoul Korea, in 1998, 2000, and 2005, respectively. He is currently an assistant professor of the Department of Electronics and Information Engineering at Seoul National University of Science & Technology. His research interests include storage systems, embedded systems, and operating systems.