Towards a Semantic Web of Community, Content and Interactions

Anupriya Ankolekar
Human-Computer Interaction Institute, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
anupriya@cs.cmu.edu

ABSTRACT

The Web plays a critical role in hosting Web communities, such as open source software (OSS) communities, their content in the form of software code and interactions around code design and bug resolution processes. However, due to the highly distributed and interdependent nature of their work as well as the high turnover rate of its members, OSS communities face significant challenges. Community members must peruse large quantities of information in order to stay aware of each other's activities and to participate meaningfully in community activities such as bug resolution. This work explores the use of Semantic Web software agents to facilitate bug resolution by enabling developers to navigate the vast information base of a project more meaningfully and by identifying people with the right expertise to assist in the resolution of a bug .

Keywords

Open source software communities, Semantic Web, agents

1. INTRODUCTION

The Web is emerging as a significant enabler of computer-supported cooperative work (CSCW) by hosting numerous communities; their content, in the form of knowledge and experiences; and their interactions, through Web-based collaboration tools, such as discussion forums, mailing lists, code repositories, Wikis, etc. Open source software (OSS) communities, which typically form around an OSS development project, are a prominent and impressively productive example. These draw participation from different types of members: users, contributors and core developers. The largest group is that of users, who download the code and report bugs. Contributors, in addition, submit bug fixes and feature enhancements to the code. They are likely to participate in technical discussions and keep abreast of the latest developments in the software. At the centre of the community is a small group of core developers, who review contributed code and select a subset to be committed to an 'official' release of the software.

As a consequence of interactions between and among these groups, knowledge and implementation experiences around the software are captured implicitly in Web-based collaboration tools [3]. For instance, most OSS communities use a Web-based bug tracking tool which supports bug-centred asynchronous discussions in the form of 'bug reports'. Bug reports capture the bug resolution process, just as archived discussions on mailing lists and Web-based forums capture online discussions. Contributors and developers regularly peruse bug reports to participate in ongoing bug resolution or to pick new bugs to work on. A particularly knotty or thorny bug may trigger additional mailing list discussions in order to reach a larger audience of developers. When the bug is eventually fixed, the fix will be committed to a versioned code repository, such as CVS, which logs changes with brief explanatory comments.

Due to the highly distributed nature of OSS software development and the fluidity of their communities, where developers spend limited time working on any single project, OSS communities face a number of challenges. These challenges are most evident within the bug resolution process. A major component of fixing bugs and producing software enhancements is understanding the source code, which is typically achieved through discussions with the developer of the code. In the OSS context, though, the original developer may not be present or around anymore. In order to understand the software, its requirements and design, possible causes of the bug and potential fixes, an OSS bug fixer must therefore wade through a vast reservoir of information: requirements and design documents, hundreds of messages, other related bug reports and even the source code itself. Thus, bug fixing activities usually involve significant collation of information from various sources. Since no single OSS developer may have expertise in the relevant code, bug resolution becomes a communal activity. Other OSS developers pitch in to contribute insights and solutions. Bug resolution thus also involves locating people with the right expertise, to bring their knowledge to bear on the bug.

Previous work on supporting OSS developers, such as Hipikat [2], allows developers to navigate through the Eclipse project information by using lexical analysis to identify related discussions and code. The Expertise Browser [4] [5] analyses CVS change files of an OSS project to determine experts in an area by identifying developers who have made numerous changes to the corresponding code and are presumably well-versed with the code. Expertise Browser focusses

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...$5.00.

on code repository logs, but valuable information about expertise can also be gleaned from participation in technical discussions and bug resolution.

Providing support for OSS bug resolution activities is challenging because they involve understanding the work of other individuals through indirect means, such as bug reports, documents, forum comments and the like. In addition, this information is often poorly linked, making navigation beyond chronological or keyword-based search difficult. This work posits that enriching Web-based community information with explicit semantics, creating a *Semantic Web for OSS communities*, enables agents to facilitate OSS bug resolution processes. The Semantic Web [1] is a vision to enrich information on the Web with semantic metadata that can be processed and reasoned about by machines. By enabling unstructured or semi-structured information to be processed based on some representation of its content, Semantic Web technologies facilitate the retrieval of semantically relevant information and the integration and maintenance of information from various sources.

Given a hierarchy of semantically meaningful concepts (i.e. an *ontology*) for an OSS community, Semantic Web agents can annotate messages with ontological concepts that they refer to. These messages can be processed by other agents to link together related discussions or potentially duplicate bugs. An OSS developer can then navigate discussions and other OSS content on the basis of the concepts they refer to. As such, an ontology enables concept-based navigation of the discussions and provides a context to interpret the material. Ontological concepts also offer fine-grained expertise location, as opposed to package-level expertise location. A secondary focus of this work is to explore the extent to which ontologies can be constructed with the help of artefacts generated through OSS interactions. Supporting the OSS bug resolution process using ontologies will help developers, especially those unfamiliar with the code, locate information required to fix a bug quicker. This, we hypothesize, will significantly reduce the time to fix bugs.

2. PROPOSED WORK

First, an ontology will be constructed to support bug resolution within a particular OSS community. It will be constructed through a systematic analysis of the OSS community artefacts and will contain the following sub-ontologies:

- The community sub-ontology will identify and describe various roles and people within the community, such as users, developers, contributors, documenters, bug fixers and so forth. The people performing these roles will be identified through their participation in the OSS community.
- The code sub-ontology will describe information about the code, such as requirements, use cases, architectural documents and implementation experiences, in addition to the implementation itself. The bug sub-ontology will describe classes of bugs and their characteristics, while the discussions sub-ontology will describe community discussions, for example posts in forums, bug reports, commit logs etc.
- The bug resolution process sub-ontology will model various stages of the bug resolution process, where agents can provide assistance. These stages include bug recognition, diagnosis of the bug, development of a fix and finally closure activities.

Thus far, a code sub-ontology and knowledge base has been constructed that describes the code implementation and links it to the CVS change history, bug reports and patches. This knowledge base is being extended with the help of text analysis tools to cover conceptually meaningful fragments of information from the text of comments and documentation. Techniques will be developed to automatically annotate messages with concepts in the ontology. The next stage of the planned work involves developing algorithms and agents that make use of the ontology and annotations to monitor and participate in the bug resolution process, linking related interactions, matching bugs to people who can help fix them and providing them with related information.

3. EVALUATION

To evaluate the impact of the developed Semantic Web on bug resolution interactions, the resolution of bugs with and without the use of Semantic Web agents will be compared, in the context of the OpenACS community. This community develops the OpenACS platform [6] for hosting Web communities, and uses the same platform to manage its own software development process and interactions.

The planned evaluation is two-fold: verifying the accuracy of the matching algorithms and examining how well the recommendations of the algorithms correspond to human judgement. For the former, the evaluation will utilise both live and historic data about OpenACS bugs to determine, given a bug, whether the 'experts' recommended by the Semantic Web agents actually end up participating in the resolution of that bug. The resolution of a random sample of fixed bugs will be 're-enacted' with Semantic Web agents, in order to determine whether Semantic Web agents can reveal crucial information about the bug at an earlier stage. For the latter, developers involved with the resolution of examined bugs (both beginner and expert) will be surveyed to determine whether the discussions and people suggested by the Semantic Web agent correspond to human judgement. Finally, we plan to observe the use of the Semantic Web system in the field, by some members of the OpenACS community, and compile various statistics, including elapsed time between various bug resolution stages.

4. ACKNOWLEDGMENTS

I would like to thank Katia Sycara, James Herbsleb, Robert Kraut and Chris Welty for their constant guidance and encouragement.

5. REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.

[2] D. Cubranic and G. C. Murphy. Hipikat: Recommending pertinent software development artifacts. In *Proc. of ICSE*, pages 408–418, Portland, OR, May 2003.

[3] T. J. Halloran and W. L. Scherlis. High quality and open source software practices. In *Proc. of 2nd Workshop on Open Source Software Engineering, ICSE*, Orlando, FL, May 2002.

[4] D. W. McDonald and M. S. Ackerman. Expertise recommender: a flexible recommendation system and architecture. In *Proc. of CSCW*, pages 231–240, 2000.

[5] A. Mockus and J. Herbsleb. Expertise browser: A quantitative approach to identifying expertise. In *Proc. of ICSE*, pages 503–512, Orlando, FL, May 2002.

[6] OpenACS: Open architecture community system. http://openacs.org/.