Agile Software Reuse Recommender

Frank McCarey
Department of Computer Science,
University College Dublin,
Belfield, Dublin 4, Ireland
frank.mccarey@ucd.ie

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence—*Miscellaneous*; H.3.3 [**Information Search and Retrieval**]: [Information filtering , Retrieval models]; D.2.13 [**Software Engineering**]: Reusable Software —*Reusable libraries*

General Terms

Design, Reliability, Experimentation

Keywords

Agile Reuse, Recommender Systems, Data Mining

1. RESEARCH PROBLEM

The demand for organisations to produce new or enhanced software implementations quickly in response to an ever changing environment has fuelled the use of Agile Processes. In such processes, the role of analysis, design models and documentation in the creation and evolution of the software is often marginalised [1]. Emphasis is placed on working code and other artifacts that provide value to the customer; in many cases source code is considered the only deliverable that truly matters. Little support documentation coupled with time constraints makes reuse of Agile components and reuse by Agile developers particularly difficult.

A mature software organisation will maintain a large growing repository of reusable components from previous projects. Traditionally, the reuse of such components has been largely ad-hoc. However, increased competitiveness and the growing complexity of software, has reinforced the need for a more structured approach to reuse. Key to this is the need for systems and tools which effectively support this software engineering approach. The inadequacy of conventional support tools has left developers unmotivated to reuse and feeling overwhelmed by a growing repository of reusable components. Frequently, the time taken to locate and integrate a reusable component will be perceived as too costly and outweighing any potential reuse benefits.

These challenges facing developers are the main motivation for our work. Reuse rewards and must be fostered. We must assist and encourage developers in making full use of large component repositories by complementing component retrieval with component recommendation, this will allow developers discover or locate components in a time efficient manner. The recommendation approach should be consistent with the principles of Agile development; reusable components currently being developed should not need any additional documentation and reuse of such components should be appealing, straightforward and require little additional developer effort.

2. RASCAL

Our research hypothesis is that *Collaborative Filtering* (CF) [2] can be used to recommend software components to a developer. CF systems are founded on the belief that users can be clustered. Users in a cluster share preferences for particular items and are likely to agree on future items. We have developed a reuse support tool named RASCAL which employs a CF algorithm to recommend a candidate set of software components to a developer. Importantly, RASCAL may recommend components which the developer is unaware off and hence unlikely to have searched for. RASCAL is designed specifically to support Agile Reuse; no additional support documentation or developer effort is required.

Empirical results are encouraging; on average, there is a 43% likelihood that RASCAL will correctly recommend the next component to a developer. Also, when little is known about a new class we can nevertheless make reasonably good recommendations. RASCAL cheaply extracts knowledge for a software company from its code repositories and exploits this knowledge in future developments. We believe RASCAL will aid developers whilst improving their productivity, enhance the quality of their code and promoting reuse.

3. ACKNOWLEDGMENTS

Thanks to my dissertation advisor, Dr. Mel Ó Cinnéide. Funding for research is provided by IRCSET (RS/2003/127).

4. REFERENCES

[1] D. Turk, R. France and B. Rumpepe. Limitations of Agile Software Process. In *Proceedings of Third International Conference on eXtreme Programming and Agile Processes in Software Engineering*, 2002.

[2] B. M. Sarwar, G. Karypis, J. A. Konstan and J, Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW10*, pages 285-295 2001.

ICSE'05, May 15–21, 2005, St. Louis, Missouri, USA.
ACM 1-58113-963-2/05/0005.