Impact: An Indirect Dependency Awareness Tool For Software Development

Jordan Ell
University of Victoria
Victoria, British Columbia
jell@uvic.ca

Daniela Damian
University of Victoria
Victoria, British Columbia
danielad@cs.uvic.ca

***Abstract*—Awareness has been a largely studied field of awareness research within software engineering. Many tools and techniques been been proposed and built in order to provide software developers and stakeholders with a greater sense of workspace and task awareness within their software projects. These techniques and tools have been largely focused on detecting *direct* conflicts which arise over a project's life time or have created an exploratory ground for stakeholders to use as a means of resolving self discovered direct or *indirect* conflicts. However, detecting and providing pertinent information regarding indirect conflicts has been a largely ignored field of research partially due to its inherently largely complexity than direct conflicts. Indirect conflicts arise from changes in one software artefact affecting another. In this paper, we present Impact, a new task awareness tool directly aimed at both detecting and presenting indirect conflicts which arise inside of a software project. We introduce previous indirect conflict awareness attempts, our design and implementation as well as describe Impact's potential through two evaluation studies.**

I. Introduction

Awareness is characterized as "an understanding of the activities of others which provides a context for one's own activities" [1]. The study of awareness and its tools has become an important topic of research in software engineering especially with the new importance of distributed work and collaboration. Awareness is generally associated with both technical and social dependencies that are created and evolve over a software project's life time. The study of these dependencies has become the primary focus of most awareness related research. Task awareness has become the most prevalent field of research to understand how developers and project managers cope with these technical and social dependencies.

Tools have been created to attempt to solve task awareness related issues with moderate success [2], [3], [4], [5]. However, these tools have been designed to solve task awareness related issues at the direct conflict level. Examples of direct task awareness include knowing when two or more developers are editing same artefact, finding expert knowledge of a particular file, and knowing which developers are working in which files. Meanwhile, task awareness related issues at the indirect conflict level continue to be an issue which is largely unsolved by most coordination mechanisms [5].

Previous interviews and surveys conducted with software developers show a consensus that developers of a software project view indirect task awareness as a high priority issue in their developement [6], [7], [8]. Examples of indirect task awareness conflicts include have one's own code effect by another's source code change and finding out who might be indirectly effect by one's own code change. Indirect conflicts arising in source code are inherantly difficult to resolve as most of the time, source code analysis must be preformed in order to find the relationships harmed by these conflicts. While some awareness tools have been created with these indirect conflicts primarily in mind [8], [9], most have only created an exploratory environment which is used by developers to solve conflicts which may arise. These tools lack the ability to detect indirect conflicts that arise and alert developers to their presence inside of the software system. Some tools have started to work directly with solving indirect conflicts [10], but continue to be limited by their usefulness.

Despite software developer's need for indirect conflict awareness tools and existing exploratory indirect conflict awareness tools, detecting and alerting developers to arising indirect conflicts is still a major problem in the field of task awareness. Impact, a web based awareness tool, aims to solve this issue. In this paper, we describe Impact's design and implementation in order to both detect newly created indirect conflicts among software developers as well as alerting developers to these conflicts. By leveraging technical relationships inherent of software projects with method call graphs as well as detecting changes to these technical relationship through software configuration management (SCM) systems, Impact is able to detect indirect conflicts as well as alert developers involved in such conflicts in task awareness.

The rest of this paper is organized as follows. First, we begin by discussing similar task awareness related tools which have partially solved the issues presented by this paper and how their workings can be applied to Impact. In the following section we describe the design and implementation of Impact as an awareness tool. We then discuss a preliminary evaluation of Impact followed by a discussion of future work and conclusions.

II. Related Work

III. Impact

A. Design

B. Implementation

IV. Evaluation

V. Results

VI. Conclusion and Future Work

References

[1] P. Dourish and V. Bellotti, "Awareness and coordination in shared workspaces," in *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, ser. CSCW '92. New York, NY, USA: ACM, 1992, pp. 107–114. [Online]. Available: http://doi.acm.org/10.1145/143457.143468

[2] P. F. Xiang, A. T. T. Ying, P. Cheng, Y. B. Dang, K. Ehrlich, M. E. Helander, P. M. Matchen, A. Empere, P. L. Tarr, C. Williams, and S. X. Yang, "Ensemble: a recommendation tool for promoting communication in software teams," in *Proceedings of the 2008 international workshop on Recommendation systems for software engineering*, ser. RSSE '08. New York, NY, USA: ACM, 2008, pp. 2:1–2:1. [Online]. Available: http://doi.acm.org/10.1145/1454247.1454259

[3] J. T. Biehl, M. Czerwinski, G. Smith, and G. G. Robertson, "Fastdash: a visual dashboard for fostering awareness in software teams," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '07. New York, NY, USA: ACM, 2007, pp. 1313–1322. [Online]. Available: http://doi.acm.org/10.1145/1240624.1240823

[4] A. Sarma, L. Maccherone, P. Wagstrom, and J. Herbsleb, "Tesseract: Interactive visual exploration of socio-technical relationships in software development," in *Proceedings of the 31st International Conference on Software Engineering*, ser. ICSE '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 23–33. [Online]. Available: http://dx.doi.org/10.1109/ICSE.2009.5070505

[5] H. Khurana, J. Basney, M. Bakht, M. Freemon, V. Welch, and R. Butler, "Palantir: a framework for collaborative incident response and investigation," in *Proceedings of the 8th Symposium on Identity and Trust on the Internet*, ser. IDtrust '09. New York, NY, USA: ACM, 2009, pp. 38–51. [Online]. Available: http://doi.acm.org/10.1145/1527017.1527023

[6] D. Damian, L. Izquierdo, J. Singer, and I. Kwan, "Awareness in the wild: Why communication breakdowns occur," in *Global Software Engineering, 2007. ICGSE 2007. Second IEEE International Conference on*, aug. 2007, pp. 81 –90.

[7] C. A. Halverson, J. B. Ellis, C. Danis, and W. A. Kellogg, "Designing task visualizations to support the coordination of work in software development," in *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, ser. CSCW '06. New York, NY, USA: ACM, 2006, pp. 39–48. [Online]. Available: http://doi.acm.org/10.1145/1180875.1180883

[8] A. Begel, Y. P. Khoo, and T. Zimmermann, "Codebook: discovering and exploiting relationships in software repositories," in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ser. ICSE '10. New York, NY, USA: ACM, 2010, pp. 125–134. [Online]. Available: http://doi.acm.org/10.1145/1806799.1806821

[9] E. Trainer, S. Quirk, C. de Souza, and D. Redmiles, "Bridging the gap between technical and social dependencies with ariadne," in *Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange*, ser. eclipse '05. New York, NY, USA: ACM, 2005, pp. 26–30. [Online]. Available: http://doi.acm.org/10.1145/1117696.1117702

[10] A. Sarma, G. Bortis, and A. van der Hoek, "Towards supporting awareness of indirect conflicts across software configuration management workspaces," in *Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering*, ser. ASE '07. New York, NY, USA: ACM, 2007, pp. 94–103. [Online]. Available: http://doi.acm.org/10.1145/1321631.1321647