Microblogging in Global Software Development

Lars Klimpke

Department of General Management and Information Systems
University of Mannheim
68131 Mannheim, Germany
klimpke@uni-mannheim.de

Abstract. Despite the current trend towards global software development, many software enterprises lack effective media for communication and collaboration. Moreover, social media are gaining attraction especially in private settings but are largely neglected in the context of distributed software development. Therefore, the goal of this design-oriented research endeavor is to develop a methodology and a corresponding tool to support communication, collaboration, and traceability in global software development. In order to address this goal, a microblogging tool is being adapted to the specific needs of developers working in distributed settings.

Keywords: Global Software Development, Social Media, Microblogging, Traceability

1 Introduction

Global software development (GSD) is gaining increasing attraction. Major drivers for this trend are gains in flexibility, expected cost savings, or faster development cycles due to round-the-clock development [1]. However, GSD complicates collaboration between team members working at different sites. This is why it often does not lead to the expected outcomes. For instance, awareness is hard to achieve in terms of (1) who is working on which task, (2) who is working at a certain moment, and (3) whom to contact about what [1–4]. Furthermore, it is necessary – especially in distributed settings – to provide traceability information within the development process [5].

Another trend of the last decade is the growth of Social Media in private as well as enterprise settings. Despite the fact that these media bear high potentials especially for GSD, the role of social media usage in software engineering is not well understood [6]. For instance, microblogging with its informal character and its focus on enhancing awareness has the potential to address problems in GSD that result from temporal, spatial, and cultural distance.

In order to support GSD, this design-oriented research endeavor aims at the following objectives:

1. To develop a methodology to enhance awareness, communication, collaboration, and traceability in global software development and
2. to design a software tool that supports this methodology.

The remainder of this research proposal is structured as follows: the underlying foundations are presented in the next section, followed by a review of prior research, the deduction of the proposed approach, and the presentation of the research methodology in Section 3. Finally, Section 4 concludes the proposal.

2 Foundations

2.1 Global Software Development

Enterprises increasingly create software in so-called virtual teams that are "internationally distributed groups of people with an organizational mandate to make or implement decisions with international components and implications" [7, p. 473]. This enables people to work collectively on interdependent tasks without being in the same organization or at the same place, not even in the same time zone [8]. That means that the virtual team does not have shared workplaces, as some team members may live in North America or Asia while others live in Europe.

Today, not only large enterprises engage in GSD; it is gaining attraction even for small and medium sized enterprises [9]. Enterprises conducting GSD hope to gain flexibility, reduce development times by round-the-clock development, decrease labor costs by employing people in lower-wage countries, and access a larger and better-skilled developer pool [1–3]. But besides the advantages of GSD, virtual teams struggle with complex settings in distributed projects. In addition to common challenges of software development, virtual teams have to deal with GSD-specific challenges resulting from the geographical, temporal, and cultural distances [3, 10]. Especially the knowledge transfer across these distances is challenging [11]. Altogether, this results in higher efforts for communication and coordination and, thus, in higher development costs.

Some of these problems can be contained by maintaining traceability of the whole development project [5, 12]. In this context, traceability is "the ability to relate artifacts created during the development of a software system to describe the system from different perspectives and levels of abstraction with each other, the stakeholders that have contributed to the creation of artifacts, and the rationale that explains the form of artifacts" [13]. Thus, it is an important process that facilitates acquisition and use of process knowledge [14]. Traceability also implies the management of rationale information that is, for instance, the capturing of reasons that led to decisions regarding the design of artifacts [12]. Therefore, traceability and rationale management (TRM) is used as one common concept in this paper.

2.2 Social Media and Microblogging

Since the last decade, Social Media have been used to a greater extent in all aspects of life. However, there is no clear definition for the term Social Media in research until now. Most definitions agree on the aspect that Social Media involve user-generated content. These users are part of an online community based upon a service or a product [15]. They communicate via text, audio and/or video messaging to publish their thoughts, ideas, and opinions in order to share them with anyone around the world. Using Social Media, nearly all content is linked to a known author and can be traced back

accordingly. Social Media changed the user behavior from "a passive, reading audience into active, contributing participants" [15, p. 1]. Thus, Social Media are mainly about participation, openness, conversation, community, and connectedness. While the popularity of Social Media tools has grown over the last years especially in a private context, many enterprises are currently establishing policies regulating the use of Social Media. Some enterprises follow progressive strategies creating sets of dedicated tools to use, others try to block the use of Social Media at work as far as possible.

In GSD, Social Media can be used to support communication [6], which in turn is a key success factor [16]. Social Media tools can be distinguished into different genres with different foci: (1) (collaborative) content creation tools, (2) content sharing tools, (3) virtual worlds, and (4) social networks (based on [17]). This research endeavor focuses on microblogging tools, which can be seen as more simple types of social networks that concentrate on posting short thoughts and ideas to a personal blog and are also known as notification tools or quick-ping media [18]. Microblogs allow people to write short messages (usually around 140 characters) in a rather informal way and, thus, support ad-hoc communication [19]. The most prominent examples are Twitter[1] and the enterprise microblogging tool Yammer[2].

3 Using Microblogging for Software Development

3.1 Related Research

This research endeavor aims at integrating microblogging into the working environment of developers. The idea of integrating communication media into an integrated development envoronment (IDE) is not entirely new. For instance, Fitzpatrick et al. developed an event notification system that integrates the Concurrent Version System (CVS) and chat functionalities with the goal to increase awareness, coordination, and communication [20].

Handel et al. introduced instant messaging capabilities into development environments in order to increase awareness [21]. Additionally, Sinha et al. aimed at supporting collaboration and awareness in distributed requirements management and developed a collaborative tool with functions for informal communication and change management [22]. Their tool was integrated into the Eclipse IDE and supports synchronous as well as asynchronous communication around requirements.

More recently, research on the use of social media and microblogging in software development emerged. For instance, Storey et al. discussed the use of social media in software development [6]. They reported a trend from integrated development environments to collaborative development environments and social development environments. Additionally, they demanded further research regarding the use of social media in software development.

Reinhardt presents an approach that integrates microblogging features into an IDE [19]. This approach facilitates ad-hoc communication via twitter, however, does not

[1] http://www.twitter.com (03/15/11)
[2] http://www.yammer.com (03/15/11)

include features like the support for traceability and rationale management as intended in this research endeavor.

Guzzi et al. implemented a tool that integrates a microblogging-like environment into the Eclipse IDE in order to aid software developers in understanding and documenting the development progress [23]. Their overall goal was to support software maintenance and to avoid the loss of knowledge. Therefore, they automatically collected interaction data from the IDE. Interestingly, Guzzi et al. discovered that the typical length of messages created by software developers is approximately 55 characters. Thus, 140 characters as used in Twitter seem to be enough to satisfy the needs of developers [23].

Apart from academic efforts, a couple of tools that integrate Twitter into the Eclipse IDE emerged, as, for instance, Tweethub[3], Twitclipse[4], Twikle[5], and Twitterclipse[6]. All these tools provide Twitter user interfaces that are integrated into the Eclipse IDE. A different approach is realized by the Snipper tool[7]. This tool is intended to let developers share code snippets via Twitter.

All tools mentioned generally focus on displaying microblogs and provide the ability to send messages to Twitter accounts. They lack the ability to support TRM as it is intended in this research endeavor (by linking messages with source files they relate to, for instance) and are not integrated into an enterprise environment. Additionally, none of these approaches has been evaluated in an appropriate way in order to prove its usefulness. Nevertheless, it is obvious that microblogging integration is gaining attraction in the software development community. However, no current tool or collaborative development environment supports all the activities for global software engineering [24].

3.2 Proposed Approach and First Results

The steps of this research endeavor are briefly summarized in Table 1. As a first step, case studies with eight small and medium-sized enterprises (SME) that work in globally distributed settings have been conducted [9]. These case studies revealed current problems of SME in GSD. Among others, SME report communication issues, missing domain knowledge and knowledge transfer as well as spatial distance and time differences as major obstacles in GSD [9]. In order to address these problems, the objective of this research endeavor is to develop a methodology that increases awareness, communication, collaboration, and TRM in GSD and to design a tool that demonstrates the methodology. This methodology is intended to aid developers in GSD by providing access to the social capital of the developers, to support asynchronous and synchronous communication, and to provide an additional means for documentation.

As a second step, requirements for such an approach were deduced from Media Synchronicity Theory (MST) [25, 26] as well as from the concept of Social Capital (SC) [27–29]. In a nutshell, the following requirements were deduced:

[3] http://wiki.eclipse.org/TweetHub (03/15/11)
[4] http://twitclipse.sourceforge.net (03/15/11)
[5] http://www.creative-mindworks.de/twikle (03/15/11)
[6] http://marketplace.eclipse.org/content/twitterclipse (03/15/11
[7] http://marketplace.eclipse.org/content/snipper-code-sharing-service (03/15/11)

Table 1. Research Endeavor

STEP	RESULTS / PLANNED ACTIVITY	STATUS
Case studies	Need for support for communication and collaboration identified.	Done
Theoretical deduction of requirements	Deduction of requirements from media synchronicity theory and the concept of social capital. Assessment of social media and communication media regarding these requirements. Decision to concentrate on microblogging.	Done
Prove of relevance for practice	Case study to prove relevance for practice of the theoretically deduced requirements.	In progress
Market research	Decision to use Eclipse and status.net.	Done
Development of the methodology	Development of guidelines that support communication and collaboration of developers in distributed settings and that allows them to realize TRM in a simple way.	In progress
Adaption of a web application	Familiarization with status.net, configuration, and implementation of add-ons.	In progress
Development of an IDE plug-in	Development of a plug-in for Eclipse that supports microblogging on the one hand and facilitates better traceability on the other hand.	Planned
Evaluation	Evaluation of the approach in an experimental setting in a student project with experimental and control group.	Planned

- Informal communication in terms of (1) communication that is not directly work related as this aims at creating team building and trust and in terms of (2) the informal character of the messages as this encourages non-native speakers to communicate in the project language should be supported.
- Synchronous communication should be enabled but not be enforced.
- Awareness of team members should be enhanced.
- Team members should be enabled to maintain and use existing as well as "indirect" SC by stimulating their awareness of each other. Indirect SC is the SC of ones contacts, for instance.
- Connecting different actors should be facilitated.
- The approach should be lightweight and straightforward to use. Thus, appropriate solutions should be integrated into the usual working environment of the users.

Additionally, the following requirements were derived from software engineering practice:

- Users should be able to document any changes and discussions in a simple way. Thus, TRM should be supported.
- The solution should fit to enterprise settings. Public solutions like Twitter, for instance, should not be used in an enterprise context, due to security issues.

These theoretically deduced and conceptually derived requirements are being proved for practice relevance in an intermediary step. An assessment of communication media and social media regarding these requirements has led to the decision to concentrate on microblogging to support GSD, especially since microblogging supports synchronous

as well as asynchronous communication and provides an easy means for the enhancement of users' awareness of each other [30]. Additionally, microblogging is a rather passive medium that reduces email overload, i.e., users are not – like with email – forced to read and work on messages [31]. Finally, microblogging functionalities can easily be integrated into other tools in order to complement these with the respective features.

Besides the development of a methodology and the adaption of a web application, it is planned to develop a plug-in for an IDE. This plug-in is intended to be integrated into the common working environment of developers. Additionally, the plug-in should allow for connecting discussions via the microblogging tool directly with the part of the code they are related to in order to support TRM.

As a third step, extensive market research led to the decision to use the Eclipse IDE[8] and the microblogging-tool status.net[9]. The market research for social media comprised 71 tools in total. All of them have been analyzed regarding their license model and functionality. The status.net tool already provides a lot of necessary functions for the web application and is available under a creative commons attribution license and, therefore, fulfilled the requirements best. Eclipse was chosen due to its license model and its widespread use in the software development community.

The fourth step is the development of the methodology, followed by the adaption of status.net and the development of the Eclipse plug-in. Until now, the development phase of the methodology and the web application has started. The next steps will be to finish development on these tasks and to develop the Eclipse plug-in. Finally, the approach will be evaluated regarding usefulness and effectiveness in an experimental setting, i.e., in a student's project with control group and experimental group.

By creating and evaluating an IT artifact, this research endeavor follows a design-science approach [32]. Table 2 summarizes the compliance with the requirements for design science research according to Hevner et al. [32].

4 Conclusion

Using microblogging to support awareness, communication, collaboration, and TRM is a promising approach for tackling ongoing problems in GSD that needs further academic advice. A step into this direction is provided by this research endeavor that aims at integrating microblogging into a development environment. Herewith, awareness between distributed team members will be enhanced, team members will be able to exchange their knowledge on urgent tasks in an easy and informal way, and traceability of software development projects will be supported.

Acknowledgment

This doctoral research is being supervised by Prof. Armin Heinzl and is supported by the German state of Baden-Württemberg within the research project "GlobaliSE".

[8] http://www.eclipse.org (03/15/11)
[9] http://status.net (03/15/11)

Table 2. Mapping against Design Science Guidelines

GUIDELINE	CONTRIBUTION
Design as an Artifact	The research outcomes (methodology and tool) can be mapped against methods and instantiation.
Problem Relevance	The research problem addresses current issues in GSD and the need for a solution is derived from empirical data.
Design Evaluation	Utility and efficiency of the designed artifacts will be evaluated in an experimental setting.
Research Contributions	The design artifacts and the design construction extend and improve existing knowledge in GSD.
Research Rigor	The design principles and the need for such a solution are deduced from theory.
Design as as Search Process	Iteration loops between artifact development and theoretical work lead to the final artifact.
Communication of Research	Technical details will be provided for technical audiences; implications and guidelines will be provided for management oriented audiences.

References

1. Herbsleb, J.D., Moitra, D.: Global Software Development. IEEE Software **18**(2) (2001) 16–20
2. Herbsleb, J.D.: Global Software Engineering: The Future of Socio-Technical Coordination. In: Future of Software Engineering. (2007) 188–198
3. Conchúir, P.J., Olsson, H.H., Fitzgerald, B.: Global Software Development: Where are the Benefits? Communications of the ACM **52**(8) (2009) 127–131
4. Kotlarsky, J., Oshri, I.: Social Ties, Knowledge Sharing and Successful Collaboration in Globally Distributed System Development Projects. European Journal of Information Systems **14** (2005) 37–48
5. Hildenbrand, T.: Improving Traceability in Distributed Collaborative Software Development – A Design Science Approach. Volume 33. Peter Lang, Frankfurt (2008)
6. Storey, M.A., Treude, C., van Deursen, A., Cheng, L.T.: The Impact of Social Media on Software Engineering Practices and Tools. In: Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research. (2010) 359–364
7. Maznevski, M.L., Chudoba, K.M.: Bridging Space over Time: Global Virtual Team Dynamics and Effectiveness. Organization Science **11**(5) (2000) 473–492
8. Lipnack, J., Stamps, J.: Virtual Teams: Reaching across Space, Time, and Organizations with Technology. John Wiley & Sons, Inc., New York, NY, USA (1997)
9. Klimpke, L., Kramer, T., Betz, S., Nordheimer, K.: Globally Distributed Software Development in Small and Medium-Sized Enterprises in Germany: Reasons, Locations, and Obstacles. In: Proceedings of the 19th European Conference on Information Systems. (2011)
10. Noll, J., Beecham, S., Richardson, I.: Global Software Development and Collaboration: Barriers and Solutions. ACM Inroads **1**(3) (2010) 66–78
11. Oshri, I., Fenema, P.V., Kotlarsky, J.: Knowledge Transfer in Globally Distributed Teams: The Role of Transactive Memory. Information Systems Journal **18**(6) (2008) 593–616
12. Klimpke, L., Hildenbrand, T.: Towards End-to-End Traceability: Insights and Implications from Five Case Studies. In: Proceedings of the Fourth International Conference on Software Engineering Advances. (2009) 465–470

13. Spanoudakis, G., Zisman, A.: Software Traceability: A Roadmap. In: Handbook of Software Engineering and Knowledge Engineering. World Scientific Publishing, River Edge,USA (2004) 395–428
14. Mohan, K., Kumar, N., Benbunan-Fich, R.: Examining Communication Media Selection and Information Processing in Software Development Traceability: An Empirical Investigation. IEEE Transactions on Professional Communication **52**(1) (2009) 17–39
15. Katajisto, L.: Implementing Social Media in Technical Communication. In: International Professional Communication Conference. (2010) 236–242
16. Carmel, E., Agarwal, R.: Tactical Approaches for Alleviating Distance in Global Software Development. IEEE Software (2001) 22–29
17. Lietsala, K., Sirkkunen, E.: Social Media – Introduction to the Tools and Processes of Participatory Economy. Tampere University Press (2008)
18. McFedries, P.: Technically Speaking: All A-Twitter. IEEE Spectrum **44**(10) (2007) 84–84
19. Reinhardt, W.: Communication is the Key – Support Durable Knowledge Sharing in Software Engineering by Microblogging. In: Proceedings of the 1st International Workshop on Software Engineering within Social Software Environments. Volume 150. (2009) 329–340
20. Fitzpatrick, G., Marshall, P., Phillips, A.: CVS Integration with Notification and Chat: Lightweight Software Team Collaboration. In: Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work. (2006) 49–58
21. Handel, M., Herbsleb, J.D.: What is Chat Doing in the Workplace? In: Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work, A (2002)
22. Sinha, V., Sengupta, B., Chandra, S.: Enabling Collaboration in Distributed Requirements Management. IEEE Software **23** (2006) 52–61
23. Guzzi, A., Pinzger, M., van Deursen, A.: Combining Micro-Blogging and IDE Interactions to Support Developers in their Quests. In: IEEE International Conference on Software Maintenance. (2010)
24. Lanubile, F., Ebert, C., Prikladnicki, R., Vizcaino, A.: Collaboration Tools for Global Software Engineering. IEEE Software **27**(2) (2010) 52–55
25. Dennis, A.R., Valacich, J.S.: Rethinking Media Richness: Towards a Theory of Media Synchronicity. In: Proceedings of the 32nd Annual Hawaii International Conference on System Sciences. (1999)
26. Dennis, A.R., Fuller, R.M., Valacich, J.S.: Media, Tasks, and Communication Processes: A Theory of Media Synchronicity. MIS Quarterly **32**(3) (2008) 575–600
27. Nahapiet, J., Ghoshal, S.: Social Capital, Intellectual Capital, and the Organizational Advantage. Academy of Management Review **23**(2) (1998) 242–266
28. Tsai, W.: Social Capital, Strategic Relatedness and the Formation of Intraorganizational Linkages. Strategic Management Journal **21**(9) (2000) 925–939
29. Rottman, J.W.: Successful Knowledge Transfer within Offshore Supplier Networks: A Case Study Exploring Social Capital in Strategic Alliances. Journal of Information Technology **23**(1) (2008) 31–43
30. Ehrlich, K., Shami, N.S.: Microblogging Inside and Outside the Workplace. In Cohen, W.W., Gosling, S., eds.: Proceeedings of the Fourth International AAAI Conference on Weblogs and Social Media. (2010)
31. Günther, O., Krasnova, H., Riehle, D., Schöndienst, V.: Modeling Microblogging Adoption in the Enterprise. In: Proceedings of the 15th Americas Conference on Information Systems. (2009)
32. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly **28**(1) (2004) 75–105