Workshop report from Web2SE: First Workshop on Web 2.0 for Software Engineering

Christoph Treude
University of Victoria
Victoria, Canada
e-mail: ctreude@uvic.ca

Margaret-Anne Storey
University of Victoria
Victoria, Canada
e-mail: mstorey@uvic.ca

Kate Ehrlich
IBM TJ Watson Research Center
Cambridge, MA, USA
e-mail: katee@us.ibm.com

Arie van Deursen
Delft University of Technology
Delft, The Netherlands
e-mail: arie.vandeursen@tudelft.nl

DOI: 10.1145/1838687.1838699
http://doi.acm.org/10.1145/1838687.1838699

Abstract

Web 2.0 technologies such as wikis, blogs, tags and feeds have been adopted and adapted by software engineers. With Web2SE, we provide a venue for pertinent work by highlighting current state-of-the-art research, by identifying research areas, and by discussing implications of Web 2.0 on software engineering. This paper reports on the paper presentations and the discussions among participants at Web2SE 2010 as well as on future directions of the Web2SE community.

1 Introduction and Motivation

Web 2.0 mechanisms such as wikis, blogs, tags and feeds have changed the way we communicate, work and play online. Some of these technologies have made their way into collaborative software engineering processes and modern software development platforms such as IBM's Jazz which draws its inspiration from the Web[1]. Web 2.0 is built around an architecture of participation where user data is aggregated as a side-effect of using applications [O'R05]. It is both a usage paradigm and a technology paradigm that uses the Web as a platform [Mur07].

Research on the use of Web 2.0 in software development is still in its early stages. One goal of Web2SE is to investigate how Web 2.0 technologies can improve software development practices. However, as Ian Davis[2] puts it, Web 2.0 is more than just mere technologies, it is an attitude. Web 2.0 implies that processes and tools are socially open, and that content can be used in several different contexts. Web2SE aims at improving our understanding of how this attitude, manifested in technologies such as mashups or dashboards, can change the culture of collaborative software development.

Web2SE has four main goals:

- Summarize the current state-of-the-art research with regard to the use of Web 2.0 technologies in software development. Web 2.0 technologies such as wikis [Lou06], facebook [BPZ10], blogs [PM09], social bookmarking [MFK06] and tagging [SRS+09] have already in part been adopted by software developers and by development environments. Projects such as Mozilla's web-based framework for code editing Bespin[3] and websites such as stackoverflow[4] also bring Web 2.0 into software development.

- Explore how Web 2.0 technologies in software development could be further leveraged. Some Web 2.0 technologies such as micro-blogging are still fairly new and it is not clear if and how they can improve software development, although projects have started to adopt twitter as a supporting mechanism[5]. Micro-blogging could also be useful to distributed software teams by providing status updates for increased awareness and by providing a channel for questions and information sharing.

- Investigate to what extent the "socially open" attitude of Web 2.0 applies to software development. A main characteristic of most Web 2.0 technologies is that they get better the more people use them. Web 2.0 applications are increasingly data-driven and the main advantage of web applications is the extent to which users add their own data [O'R05]. How to balance an architecture of participation and individual productivity is an open question.

- Explore how Web 2.0 technologies can be incorporated into software engineering processes and methods. Using Web 2.0 tools to assist with communication and coordination is only one application of Web 2.0 tools for software development teams. It is even less explored how Web 2.0 technologies can change software development processes and methods.

[1] http://jazz.net/projects/content/project/plans/jia-overview/
[2] http://iandavis.com/blog/2005/07/talis-web-20-and-all-that
[3] https://bespin.mozilla.com/
[4] http://stackoverflow.com/
[5] E.g., http://wiki.eclipse.org/twitter

Web2SE 2010 at the International Conference on Software Engineering (ICSE) 2010 in Cape Town, South Africa, was attended by 24 registered participants. The program was organized in four sessions: In three sessions a total of eight papers were presented and discussed by all participants. For each paper, the discussion was initiated by a number of questions carefully prepared in advance by one or two workshop participants. The paper discussions were followed by an interactive panel session "The Wisdom of the Crowd – A Mashup of Emergent Web2SE topics". The workshop participants were invited to discuss the themes that emerged during the workshop. In the spirit of the workshop topic, we used Web 2.0 tools throughout the workshop to take notes collaboratively. While our Google Waves[6] suffered from low bandwidth, twitter with the #web2se hashtag[7] was quite active.

The remainder of this paper is structured as follows. After a short section on the current use of Web 2.0 mechanisms in software development, we outline the papers that were presented at Web2SE in Section 3, and we review the panel discussion in Section 4. Section 5 concludes the report by outlining future work of the Web2SE community.

2 Background

The use of Web 2.0 in software development crosscuts many of the traditional categories in software development. Web 2.0 mechanisms are used across various development processes and tools and they can support activities ranging from requirements engineering and development to testing and documentation. Social media features such as wikis, blogs or mashups can either be used as an adjunct or be integrated into development environments.

Some Web 2.0 mechanisms such as **wikis** were designed with development collaboration in mind [LC01]. Their use in software engineering is widespread, in particular for documentation activities. **Blogs** are used by developers to communicate "how-to" information, to highlight features and for requirements engineering [PM09]. **Microblogs** such as twitter are also used in communication and they could potentially be integrated into IDEs [vDMC+10]. Social **tagging** can be used by software developers to tag issues [TS09], source code [SRS+09] or projects, e.g. on Freshmeat[8]. **Feeds** provide updates when content frequently changes. Tools such as Jazz use feeds to create awareness [TS10b]. **Social networks** enable the creation of virtual communities. Taking this approach to software development, frameworks such as Codebook [BPZ10] allow developers to connect with artifacts and to keep track of dependencies and connections. **Crowdsourcing** enables new forms of user involvement and can be used during requirements engineering or testing[9].

[6] https://wave.google.com/wave/
[7] http://twitter.com/web2se
[8] http://freshmeat.net/
[9] E.g., http://www.google.com/websiteoptimizer

3 Accepted Papers

3.1 How Web 2.0 is Used in Software Engineering

The paper presentations started with a session on the current use of Web 2.0 by software developers to set the stage for the rest of the workshop. The first paper in this session by Black *et al.* [BHB10] reported preliminary results on a pilot survey of social media use in global systems development. The authors created an online survey for developers of software systems and web applications who use social media at work. The results of the survey indicate that social media tools such as twitter and instant messaging can improve communication. In particular, 91% of the respondents indicated that the use of social media has improved their working life.

The second paper by Black and Jacobs [BJ10] on the use of Web 2.0 to improve software quality reported on industrial case studies from several domains to provide insights into the emergent state of the art. Using real world experiences from early adopters, the authors show how using social media can enhance and improve the development experience. They argue that social media tools are starting to become mainstream. The software developers in their studies used a broad range of Web 2.0 tools including Google Wave, twitter, wikis and forums. In times of outsourcing and distributed teams, the software industry is in a perfect position to take advantage of the latest social media trends.

3.2 Mining Software Repositories and Web 2.0

The use of Web 2.0 technologies by software developers leaves traces and artifacts in the development repositories. What we can learn from these artifacts was the topic of the second session at Web2SE.

In their paper on different schemata and metadata for tags, Treude and Storey looked at the implications of how software artifacts are tagged by software developers [TS10c]. Social tagging has been adopted in various contexts from source code to issue trackers and build definitions. The success of tagging in software development is usually attributed to the simplicity of tags; however, the details of different tagging systems vary significantly in terms of metadata, schemata and semantics. The authors argue that academia and industry have to be aware of these differences and have to start examining their implications.

D'Ambros *et al.* [DLR10] presented an IDE enhancement called Commit 2.0. Commit 2.0 enriches commit comments using software visualizations that are generated based on the changes related to a commit at different granularity levels. In addition, developers can annotate the visualizations. Commit 2.0 was motivated by the observation that commit comments in software development are useful both for developers and researchers; however, they are limited in the sense that IDEs only support text for change documentation.

In their work on a newsfeed system on top of the Codebook software repository mining platform, Begel and Zimmermann [BZ10] describe how software development teams can find relevant information using newsfeeds of activities mined from software repositories. The authors argue that development teams who work collaboratively need to be aware of what other developers are doing in order to manage the risk of taking on dependencies.

3.3 Towards Web 2.0 Tools for Software Engineering

Looking at concrete tool efforts, the afternoon session of Web2SE started with an exploration of the use of mashups in software development by Grammel *et al.* [GTS10]. The authors observed that software engineering tool research is often focused on small stand-along tools rather than comprehensive development environments. They argue that researchers should instead provide developers with flexible environments and tools, and then study how developers appropriate and tailor these tools. Mashups are particularly aimed towards analytic activities involving multiple information sources.

In their paper on collective intelligence during prerequirements analysis, Liang *et al.* [LAHX10] argue that requirements engineering is a social activity in which stakeholders have to work together closely. Current tools are often deficient in supporting the dynamic social interactions and collective decisions required. The authors propose to address these challenges using Web 2.0 technologies in three steps: obtaining requirements knowledge through tagging, transforming tags into requirement ontologies, and supporting decision-making through reasoning based on the ontologies.

Tansey and Stroulia [TS10a] describe Annoki, a collaboration platform built on top of a popular wiki software. Annoki supports collaboration in software development by extending the functionality of a wiki in terms of better organization, access management, creation assistance and graphical displays of content. The current use of Annoki is discussed in the paper. The largest user group is a research lab where Annoki is used to manage research and development activities on a daily basis.

4 Panel Discussions

4.1 Discussion Points

At the beginning of the Web2SE panel discussion, a quick survey among the participants revealed the 8 potential panel topics shown in Table 1. The number of votes for each topic is also shown in Table 1. We decided to discuss the top 3 topics at the workshop, and the results of the discussion are outlined in the following sections.

Panel Topic	Votes
Information overload	13
Privacy	11
IDE to browser	10
Democratization	6
Adoption / Barriers	5
Surprising uses	4
Mobile devices	2
Web 2.0 channels	1

Table 1: Votes for Panel Topics

4.2 Information Overload

One of the disadvantages of Web 2.0 is that it can easily generate too much content which can be a distraction. There is a desire by developers to only receive content that is relevant to their work.

While the participants agreed that information overload is an issue when using Web 2.0 technologies in software development, they also found that the concept of filtering does not really fit into Web 2.0 style thinking. Rather than filtering content based on external criteria, Web 2.0 implies filtering through people. To do so, roles can be assigned to certain individuals and the individual with the most influential role does the filtering for other individuals. However, the idea of certain roles being more influential than others contradicts the democratization that comes with many Web 2.0 technologies.

Several alternatives to filtering were discussed at the workshop:

- Generate less information.
- Generate summaries.
- Voting, where the topic with the most positive votes ("thumbs up") is defined as the most relevant topic. This approach is for example used by stackoverflow. As a potential problem with this approach, the "tyranny of the majority" was mentioned. Also, some of the voting systems currently in place are encountering the problem that users add irrelevant content. This phenomenon, for example observed in Digg[10], was referred to as the "tipping point of social media". The general trade off can be described as "curation vs. popularity".
- Context-sensitive labeling. One workshop participant suggested using our software engineering knowledge to relate information and to put information in context.
- Automated categorization.
- Interaction mining.
- Interruption management. A good solution to interruption management might also reduce the problem of information overload, if the information is displayed only

[10] http://digg.com/

at suitable times. As an example it was mentioned that many individuals only deal with their email inbox during certain times every day – a similar approach might be fruitful when dealing with Web 2.0 artifacts such as tweets.

Based on this discussion, it was asked whether there is an evolutionary drive to consume as much information as possible, and whether governments should regulate our consumption of social media.

The importance of process when using Web 2.0 was emphasized by several workshop participants. For example, in a company setting, it is crucial to know whether reading all company tweets is required, whether replies to emails are expected within a certain time frame, and which Web 2.0 channels (such as wiki, twitter, email or IM) are to be used for certain kinds of communication. It is important that these assumptions are formalized. Often, the underlying relationship between individuals plays a role here – an email response to a manager is treated differently from an email response to a colleague. These relationships are often implicit.

As Web 2.0 becomes more important, switching off services such as twitter or facebook for maintenance reasons becomes problematic. A recent example was given when the U.S. government intervened with a scheduled maintenance of twitter during the protests following the 2010 election in Iran. For companies relying on Web 2.0 mechanisms, this implies that Web 2.0 services need to be cloneable – the companies have to be in charge of the service and not be subject to outages caused outside of the company.

This observation led into the discussion of the next panel topic: privacy. Digital privacy requires new laws that are still being developed and continuously emerge. It also poses the question whether we as researchers should study how Web 2.0 mechanisms are currently used considering those privacy concerns. There is a considerable risk of not knowing how Web 2.0 is used if we refrain from studying it, though.

4.3 Privacy and Ethics

The workshop discussion on privacy and ethics started with the example of the use of twitter at conferences, in particular with the question whether it is ethical to quote other individuals such as keynote speakers in conference related tweets. What happens if individuals are mis-quoted? Does self-correction by the mis-quoted individual or other members of the community work in practice?

These questions emphasize the importance of etiquette and ethics of using twitter. While several companies have explicit rules about how their employees should use blogging, only very few companies regulate the use of twitter. Social media law is an emerging topic, and while some lawyers understand social media, that does not hold true for all of them. An additional layer of complexity is added to law around social media by international differences. While the law in one country might be clear-cut, software is often developed across borders and different jurisdictions have to be taken into account.

The discussion then turned to a researcher's ethical obligations to Open Source developers before reading their communication channels. In terms of legal obligations, it depends on the license – but there are also moral obligations to be considered. As a general approach it was suggested to try to communicate with the individuals who created the artifacts. Another open question is whether data should be anonymized at the beginning, i.e. before the researcher starts the analysis, or at the end, i.e. for publication.

A general problem identified in this discussion is that we do not have good metaphors for privacy in social media. One participant suggested that employers reading social media artifacts from their employees are comparable to employers reading all notes in a filing cabinet and then using this information in performance reviews. Another participant suggested a "public art project" as a metaphor. Open Source developers are performing in public, although they may not be aware of it. A follow-up question that arises is whether a company owns the interaction that a developer had with an IDE.

4.4 IDE to Browser

The topic of the last panel discussion was the potential move of the IDE into the browser. The discussion started with the observation that this is not a simple yes/no question, but that the underlying question is "How much of the IDE should be in the browser?" It is also important to note that keeping development data in the cloud does not imply using a browser as IDE.

An alternative to moving the IDE into the browser is adding Web 2.0 plug-ins into existing IDEs. One of the reasons against moving the entire IDE into the browser is that developing a fast editor is hard enough in a desktop environment and that browsers are not fast enough yet. Other than that, the organization on the server in particular in terms of naming schemes and concurrency were mentioned as challenges of moving the IDE into the browser. Concurrency could potentially be addressed by social conventions.

The following reasons were given for moving the IDE to the browser:

- Accessibility. It would be easier to access the IDE from devices such as smart phones.
- Collaboration. Web applications are inherently collaborative.
- Data integration. Web applications make it easier to integrate data from various sources.
- Same configurations for everybody. Updates to the IDE can be centralized.
- Superficial reasons. By many, web applications are seen as "the future", and IDEs should be part of that.

These projects were mentioned as examples for efforts of moving the IDE to the browser:

- WebVelocity Smalltalk[11]
- Palm WebOS[12]
- Heroku[13]
- Bespin[14]
- EyeOS[15]
- Atlas[16]

In the case of Atlas, one participant mentioned that it was supposed to be a web-based IDE but the community voted against it because of code ownership fears.

To conclude the discussion, we had a vote among the workshop participants asking if IDEs should move to the browser. The result of the vote is shown in Table 2.

Answer	Votes
Yes	10.0
No	4.5
Maybe	1.0

Table 2: Votes: Should IDEs move to the browser?

5 Conclusions and Future

Web2SE identified a broad range of research areas and challenges with regard to the use of Web 2.0 mechanisms by software developers. It also became obvious that the use of Web 2.0 is a crucial topic for collaborative software development, and that we are starting to see more and more evidence of successful use of Web 2.0 technologies by software developers.

The use of Web 2.0 mechanisms is a moving target, and inventions are adopted and adapted faster than we can study them. To understand these fast moving socio-technical environments, the Web2SE community will continue to work towards its objectives: study the current use of Web 2.0 tools, explore further potential uses, investigate the implications of socially open software development, and explore implications for development processes and methods. Understanding the implications of Web 2.0 in software development requires studying a wide variety of projects and contexts.

At the same time we see that there is still a gap between the Web 2.0 community and the software engineering community. To fully realize the potential of Web 2.0 mechanisms for software development and to close this gap, we also need to understand how Web 2.0 is used by the Web 2.0 community. This will allow us to transfer even more ideas to software development.

[11] http://www.cincomsmalltalk.com/main/products/webvelocity/
[12] http://developer.palm.com/
[13] http://heroku.com/
[14] https://bespin.mozillalabs.com/
[15] http://eyeos.org/
[16] http://280atlas.com/

Acknowledgments

We thank all workshop participants and authors for their valuable contributions and presentations at the workshop.

We also thank the members of our program committee for providing timely and relevant reviews: Jorge Aranda (University of Toronto, Canada), Andrew Begel (Microsoft Research, USA), Li-Te Cheng (IBM TJ Watson Research Center, USA), Uri Dekel (Carnegie Mellon University, USA), Jean-Marie Favre (OneTree Technologies, Luxembourg), Harald Gall (University of Zurich, Switzerland), Michele Lanza (University of Lugano, Switzerland), Frank Maurer (University of Calgary, Canada), Jonathan Sillito (University of Calgary, Canada), Markus Strohmaier (Graz University of Technology, Austria), Gina Venolia (Microsoft Research, USA) and Thomas Zimmermann (Microsoft Research, USA).

References

[BHB10] Sue Black, Rachel Harrison, and Mark Baldwin. A survey of social media use in software systems development. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 1–5, New York, NY, USA, 2010. ACM.

[BJ10] Sue Black and Joanne Jacobs. Using web 2.0 to improve software quality. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 6–11, New York, NY, USA, 2010. ACM.

[BPZ10] Andrew Begel, Khoo Yit Phang, and Thomas Zimmermann. Codebook: discovering and exploiting relationships in software repositories. In *ICSE '10: Proc. of the 32nd ACM/IEEE Intl. Conf. on Software Engineering*, pages 125–134, New York, NY, USA, 2010. ACM.

[BZ10] Andrew Begel and Thomas Zimmermann. Keeping up with your friends: function foo, library bar.dll, and work item 24. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 20–23, New York, NY, USA, 2010. ACM.

[DLR10] Marco D'Ambros, Michele Lanza, and Romain Robbes. Commit 2.0. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 14–19, New York, NY, USA, 2010. ACM.

[GTS10] Lars Grammel, Christoph Treude, and Margaret-Anne Storey. Mashup environments in software engineering. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 24–25, New York, NY, USA, 2010. ACM.

[LAHX10] Peng Liang, Paris Avgeriou, Keqing He, and Lai Xu. From collective knowledge to intelligence: pre-requirements analysis of large and complex systems. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 26–30, New York, NY, USA, 2010. ACM.

[LC01] Bo Leuf and Ward Cunningham. *The Wiki way: quick collaboration on the Web*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[Lou06] Panagiotis Louridas. Using wikis in software development. *IEEE Software*, 23(2):88–91, 2006.

[MFK06] David R. Millen, Jonathan Feinberg, and Bernard Kerr. Dogear: Social bookmarking in the enterprise. In *CHI '06: Proc. of the Conf. on Human Factors in computing systems*, pages 111–120, New York, NY, USA, 2006. ACM.

[Mur07] S. Murugesan. Understanding Web 2.0. *IT Professional*, 9(4):34–41, 2007.

[O'R05] Tim O'Reilly. What is Web 2.0: Design patterns and business models for the next generation of software, 2005. http://oreilly.com/web2/archive/what-is-web-20.html.

[PM09] Shelly Park and Frank Maurer. The role of blogging in generating a software product vision. In *CHASE '09: Proc. of the ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, pages 74–77, Washington, DC, USA, 2009. IEEE.

[SRS+09] Margaret-A. Storey, Jody Ryall, Janice Singer, Del Myers, Li-Te Cheng, and Michael Muller. How software developers use tagging to support reminding and refinding. *IEEE Trans. on Software Engineering*, 35(4):470–483, 2009.

[TS09] Christoph Treude and Margaret-Anne Storey. How tagging helps bridge the gap between social and technical aspects in software development. In *ICSE '09: Proc. of the 31st Intl. Conf. on Software Engineering*, pages 12–22, Washington, DC, USA, 2009. IEEE.

[TS10a] Brendan Tansey and Eleni Stroulia. Annoki: a mediawiki-based collaboration platform. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 31–36, New York, NY, USA, 2010. ACM.

[TS10b] Christoph Treude and Margaret-Anne Storey. Awareness 2.0: staying aware of projects, developers and tasks using dashboards and feeds. In *ICSE '10: Proc. of the 32nd ACM/IEEE Intl. Conf. on Software Engineering*, pages 365–374, New York, NY, USA, 2010. ACM.

[TS10c] Christoph Treude and Margaret-Anne Storey. The implications of how we tag software artifacts: exploring different schemata and metadata for tags. In *Web2SE '10: Proc. of the 1st Workshop on Web 2.0 for Software Engineering*, pages 12–13, New York, NY, USA, 2010. ACM.

[vDMC+10] Arie van Deursen, Ali Mesbah, Bas Cornelissen, Andy Zaidman, Martin Pinzger, and Anja Guzzi. Adinda: a knowledgeable, browser-based ide. In *ICSE '10: Proc. of the 32nd ACM/IEEE Intl. Conf. on Software Engineering*, pages 203–206, New York, NY, USA, 2010. ACM.