Augmenting Social Awareness in a Collaborative Development Environment

Fabio Calefato
Dipartimento di Informatica
Università degli Studi di Bari "A. Moro"
Bari, Italy
calefato@di.uniba.it

Filippo Lanubile
Dipartimento di Informatica
Università degli Studi di Bari "A. Moro"
Bari, Italy
lanubile@di.uniba.it

***Abstract*—Social awareness, that is information that a person maintains about others in a social or conversational context, can contribute to counteract the lack of teamness in global software development and strengthen trust among remote developers. We hypothesize that information shared on social media can work for distributed software teams as a surrogate of the social awareness gained during informal face to face chats. As a preliminary step we have developed a tool that extends a collaborative development environment by aggregating content from social networks and microblogs into the developer's workspace.**

Keywords-trust building; collaborative development environment; social awareness; social networks

I. INTRODUCTION

S*ocial awareness* is the information that a person maintains about others in a social or conversational context [9]. Although acknowledged only recently, social awareness can contribute to the success of globally distributed projects by strengthening trust [11] [12], more specifically, *affective trust*. From an affective perspective, trust is defined as the reciprocal emotional ties, concerns, and care that morally push the trustee to do something for the trustor [10] [15].

Treinen & Miller-Frost [14] reported on several case studies where the development of mutual trust between distant sites at the beginning of a project was more important than the resolution of technical issues. Al-Ani & Redmiles [2] identified both technical and socio-emotional leadership among the positive forces acting on trust building in large software organization. Costa *et al.* [6] observed that, on a monthly basis, 25% of new coordination requirements of large-scale distributed projects involve members who do not often work together and have insufficient time to establish social connections. DiMicco *et al.* [8] analyzed the professional use of a social network within IBM to find that people did not connect to proximate colleagues with whom they communicated on a regular basis, but rather with employees they did not know well, to build stronger ties. Consistently, Ali-Hassan *et al.* [1] found evidence that publishing personal information, photos, and so on in the workplace lead people to build new ties in their networks. Bradner & Mark [5] observed that the distance of a collaborating partner affects the willingness to initially cooperate, as well as the willingness to deceive and the ability to persuade partners. Such effects, however, were observed only when people *believed* that partners lived nearby since it is *feeling close*, rather than actually *being close*, that has a trust-building effect. Shami *et al.* [13] observed that, when seeking help, participations in social software and social closeness (i.e., being a friend of a friend) account more than technical skills, since people prefer to avoid cold calls and contact other people who are more likely to respond. Finally, Bougie *et al.* [4] found that microblogs are successfully used in software engineering projects because of the little costs of displaying and monitoring actions through "tweets", thanks to their short, fixed length.

The problem with trust building is that it typically grows through close interaction and face-to-face (F2F) chats. However, F2F interaction is also the very activity that global teams see reduced, due to distance. Therefore, to date the following research question still remains open*: How do we strengthen or build trust among developers of globally distributed teams who have few or no chances to meet?*

Previous research provides some initial evidence to support the hypothesis that information shared on social media can work for distributed software teams as a surrogate of the social awareness gained during informal face-to-face chats. Therefore, there is a need for tools that support sharing personal and contextual information to increase the likelihood of successful interactions.

II. SOCIALTFS

Collaborative Development Environments (CDEs), also known as Application Lifecycle Management (ALM) platforms, are project workspaces with a standardized toolset for software teams (e.g., tracker, version control, dashboard, and event notification). The name was first coined by Booch & Brown [1], who envisioned collaborative features to be available as extensions of the core toolset that would increase users' comfort and productivity.

Although to different extents, all the largest and most used CDEs available today, such as Google Code, Rational Team Concert, and Trac, support group-structural and informal awareness. Instead, social awareness is partially supported only by GitHub, which allows its users to directly follow the developer's connections like in a person-centered social network, such as Google+ or Twitter.

SocialTFS (Figure 1a) is a tool developed as an extension of Visual Studio and Team Foundation Server (TFS) to aggregate teammates' content from social software into the Microsoft CDE. SocialTFS includes three main components. The client component, which is realized as a Visual Studio plugin, handles the visualization of all the social content collected from the services enabled by a user. The server side component builds on the ServerObjectModel API of TFS and its main duty is notifying events and workspace changes

to the other components via web service protocols, such as REST or SOAP. The third component, called Social Proxy Server, is an aggregator that accesses the API of corporate microblogs and social networking websites. Being a proxy, it interacts both with the SocialTFS client and with TFS via the HTTP/REST protocol. It builds on the ClientObjectModel API of TFS and its main duties are retrieving information about registered users from social network services (SNSs) and about software projects from CDEs. To make this possible, the Social Proxy Server stores user credentials and caches posts on behalf of users, who give authorization on the first access through OAuth, an authorization protocol used by most social software services. Other than accessing SNSs, the proxy can also handle and store connection data for both TFS corporate installations and CodePlex, which is a public TFS installation where Microsoft hosts open source projects. Finally, the Social Proxy Server runs as a web site (it requires Internet Information Services) and comes with an administration panel, where an administrator handles all the configuration, such as what SNS to enable (e.g., Twitter, but not Facebook; the URL to access the corporate microblog installation) and where TFS and its components are deployed (e.g., the URL to access the SQLServer installation).

Figure 1b shows the services currently available in SocialTFS as of this writing. They include CodePlex and TFS as supported CDEs, Twitter, Yammer and StatusNet (both public and corporate) as microblogs, and finally Facebook and LinkedIn, with Google+ scheduled for the upcoming iteration. As for the SNSs, SocialTFS allows a user to specify what information can retrieve from the account. In Figure 1c, access rights for Facebook are shown. In particular, the Social Proxy Server component is allowed to retrieve and store the list of mutual friends that one has (i.e., both the followings and the followers) and the profile picture, but none of the posts shared. Access rights are specific for each service. For instance, in the case of LinkedIn a user is also asked to give access to his/her skills as reported on the site.

As mentioned before, social content is loaded and cached by the Social Proxy Server component. Then, all the information is requested by the SocialTFS client and presented to the end user in a view within the Visual Studio IDE. Such information is shown through three different timelines, namely *home*, *iteration*, and *interactive*.

The *home timeline* in SocialTFS resembles the timeline available in microblogging sites, such as Twitter or Yammer, as it gets populated by the posts from the current user and those from the his/her followings. To avoid cold start problems, SocialTFS incorporates a recommender system that suggests whom to follow. We call this type of followings *static*, because an explicit follow/unfollow action is required to add or remove someone from one's awareness network, that is, the set of people whose actions one monitors and to whom one's actions are displayed. However, de Souza & Redmiles [7] have found that an awareness network is *fluid* and changes over time, depending on task assignments or the software development phases. Therefore, other than visualizing the stream of static followings in the home timeline, we also designed a *dynamic* type of following.

Unlike static followings, dynamic followings do not require any explicit follow/unfollow action, as they are automatically added to and removed from a user's awareness network, depending on the two different conditions detailed below. The first condition relates to the changes occurring to users' assignments in the current iteration. If, for example, Fabio reported or commented on a work item assigned to Nicola, he will be able to see Fabio's posts in the so-called *iteration timeline*. The work items considered are only those in active or fixed state in the iteration at hand. The second condition relates to actions performed by a user within Visual Studio.

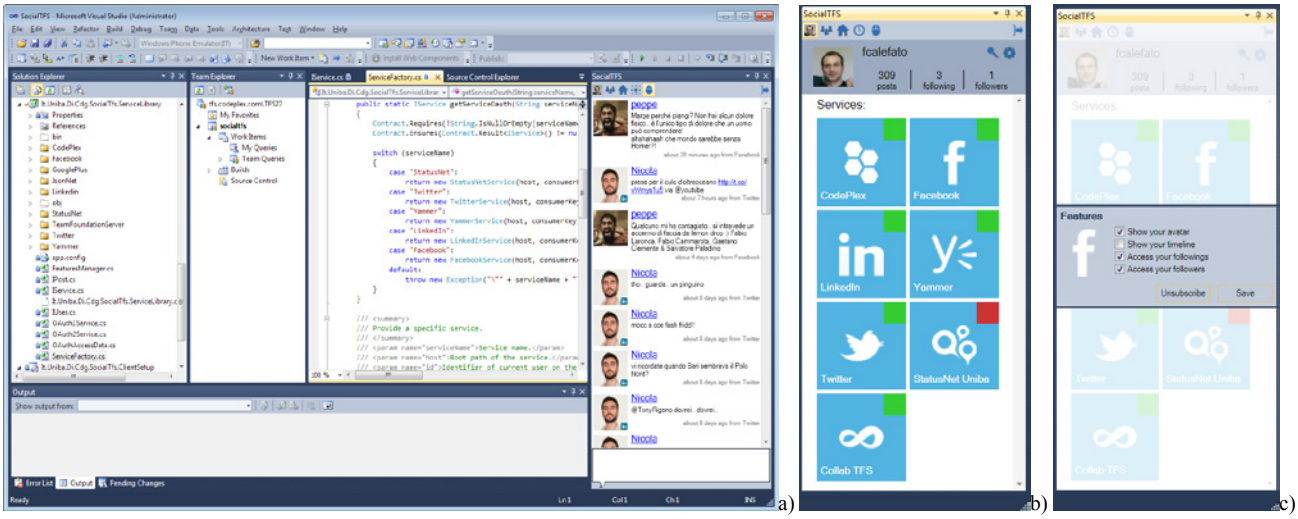

Figure 1. SocialTFS user interface (a), available SNS and microblog services (b). and customization of access rights (c).

In fact, the *interactive timeline* displays posts from dynamic followings "inferred" from the artifact (i.e., a work item or a source code file) shown in the focused tab of the main editor of the IDE. If, for example, Peppe is editing a file that has been committed by Filippo, he will appear as a dynamic following in Peppe's interactive timeline.

III. CONCLUSIONS & FUTURE WORK

In this paper we have presented SocialTFS, an extension that augments a collaborative development environment by aggregating content from multiple social media into the developer's workspace. The tool has been developed to support our hypothesis that information shared on SNSs and microblogs can work as a surrogate of the social awareness gained during informal chats, thus helping to build trust among members of global teams. As a future work, we intend to conduct case studies to empirically test our hypothesis in large scale industrial projects.

ACKNOWLEDGMENT

The work reported in this paper is currently funded by the European Territorial Cooperation Operational Programme "Greece-Italy 2007-2013" under the project Intersocial. SocialTFS has got the 2011 Software Engineering Innovation Foundation (SEIF) Award from Microsoft Research in the 2011 competition.

REFERENCES

[1] H. Ali-Hassan, D. Nevo, H.M. Kim, and S. Perelgut. "Organizational Social Computing and Employee Job Performance: The Knowledge Access Route," Proc .HICSS 2011, Hawaii, pp.1-10.

[2] B. Al-Ani and D. Redmiles, "In Strangers We Trust? Findings of an Empirical Study of Distributed Teams," Proc. ICGSE 2009, Limerick, Ireland, Jul. 13-16, pp. 121-130.

[3] G. Booch and A.W. Brown, "Collaborative Development Environments", Advances in Computers, 59, Academic Press, 2003.

[4] G. Bougie, J. Starke, M.A. Storey, and D.M. German, "Towards Understanding Twitter Use in Software Engineering: Preliminary Findings, Ongoing Challenges and Future Questions." Proc. Web2SE'11, 2011.

[5] M. Bradner and G. Mark, "Why Distance Matters: Effects on Cooperation, Persuasion and Deception. Proc. CSCW'02, 2002.

[6] J.M.R. Costa, M. Cataldo, and C.R.B. de Souza, "The Scale and Evolution of Coordination Needs in Large-Scale Distributed Projects: Implications for the Future Generation of Collaborative Tools", Proc. CHI'11, 2011.

[7] C.R.B. de Souza and D.F. Redmiles "The Awareness Network, To Whom Should I Display My Actions? And, Whose Actions Should I Monitor?" IEEE Trans. on Sw Eng, 37(3), 2011.

[8] J. DiMicco, D.R. Millen, W. Geyer, C. Dugan, B. Brownholtz, and M. Michael, "Motivations for Social Networking at Work", Proc. CSCW'08, 2008.

[9] Gutwin, C., Greenberg, S., Roseman, M.. Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation. HCI 1996

[10] S.L. Jarvenpaa, K. Knoll, and D.E. Leidner, "Is anybody out there?: antecedents of trust in global virtual teams," Journal of Manage. Inf. Syst. 14(4), pp. 29-64, 1998.

[11] J. Marlow and L. Dabbish, "Designing interventions to reduce psychological distance in globally distributed teams". In Int'l C*onf. Computer Supported Cooperative Work* (CSCW '12), Seattle, WA, USA, Feb. 11-15, 2012, pp. 163-166.

[12] H. Robinson and H. Sharp, "Organizational culture and XP: three case studies," Proc. Agile '05, 2005.

[13] N.S. Shami, K. Ehrlich, G. Gay, and J.T. Hancock, "Making Sense of Strangers' Expertise from Signals in Digital Artifacts," Proc. CHI'09, 2009.

[14] J.J. Treinen and S.L. Miller-Frost, "Following the sun: case studies in global software development", IBM Syst. J. 45(4), Oct. 2006, pp. 773-783.

[15] J.M. Wilson, S.G. Strausb, and B. McEvily, "All in due time: The development of trust in computer-mediated and face-to-face teams", Organizational Behavior and Human Decision Processes, 99(1), pp. 16–33, Jan. 2006.