Affective Trust as a Predictor of Successful Collaboration in Distributed Software Projects

Fabio Calefato
University of Bari
Taranto, Italy
fabio.calefato@uniba.it

Filippo Lanubile
University of Bari
Bari, Italy
filippo.lanubile@uniba.it

ABSTRACT
Building trust among remote developers is challenging because trust typically grows through close face-to-face interaction. In this paper, we present the preparatory design of an empirical study aimed to assess whether affective trust, established through social communication between developers, is a predictor of successful collaboration in distributed projects. Specifically, we intend to measure affective trust through sentiment analysis of pull-request comments.

CCS Concepts
•Human-centered computing •Collaborative and social computing •Empirical studies in collaborative and social computing.

Keywords
Affective trust; sentiment analysis; human factors; distributed software projects.

1. INTRODUCTION
Affective trust has been widely studied in several research domains, including software engineering [1],[15]. Other definitions of trust distinguish between *cognitive* (or rational) and *affective* (or social) perspectives. Wilson *et al.* [17] define affective trust in terms of reciprocal emotional ties, concerns, and care between the trustee and the trustor, which push the latter to do something for the former because it is perceived as moral duty. Conversely, cognitive trust is defined in terms of expectations about others' competence and reliability in performing important actions that the trustor cannot monitor. Overall the trustor's decision is based on both cognitive and affective appraisal of existing information about the trustee derived from the observation of the trustee's behavior.

Trust is fundamental in software projects, especially if distributed. Reduced trust has been reported to (a) aggravate the feeling of being separate teams with conflicting goals, (b) decrease the willingness to share information and cooperate to solve problems, and (c) affect goodwill toward others in case of objections and disagreements [2]. Trust among project members typically grows through close face-to-face (F2F) interaction, since it represents the most effective way to establish connections with others and gain awareness of both technical and personal aspects [6],[11].

SEmotion'16, May 17 2016, Austin, TX, USA
 ISBN 978-1-4503-4169-1/16/05…$15.00
DOI: http://dx.doi.org/10.1145/2897000.2897001

Unfortunately, F2F interaction is also the very activity that is largely reduced in distributed software projects or even completely unavailable, as in the case of open-source software (OSS) projects. In our previous work, we argued that collecting personal information from social media and then linking it to work artifacts can augment social awareness and affective trust [4],[5]. Prior empirical research has also focused on how to build and strengthen trust among members of distributed teams who have few or no chances to meet. Wang & Redmiles [16] studied two large OSS projects and observed that engaging in non-work-related communication with remote collaborators has trust-building effect on distant developers. A similar study was conducted by Guzzi *et al.* [7] on social interactions occurring by emails on the mailing lists of one large OSS project.

One common limitation identified in prior empirical-research findings is that there is no explicit measure of 'how much' improving trust contributes to a project performance. Here we present the preparatory design of an empirical study aimed to assess whether affective trust established between developers is a predictor of successful collaboration in distributed projects. Specifically, we will perform a sentiment analysis of pull-request comments, as pull-based code reviews are complex activities that have been found to involve both technical and social factors [7].

2. RESEARCH MODEL
Existing literature has proposed several approaches for fostering trust [1],[3],[9]. The underlying idea of these studies is that the process of trust building mainly develops along several dimensions called *trust antecedents* [14], i.e., the properties of the trustee that trigger the trustor's appraisal when assessing trustworthiness.

According to Schumann *et al*. [15], the personal properties of a trustee that facilitate the establishment of the affective trust relationship with a trustor are *benevolence* (e.g., courtesy, availability) and *integrity* (e.g., faithfulness, adherence to moral norms). Typically, information elements about these antecedents of affective trust are exchanged throughout social communication [14]. Besides, Jarvenpaa & Leidner [10] observed trust evolution in global teams interacting only through computer-mediated communication. Their analysis indicates that teams with low level of initial trust lacked in social communication at the beginning of projects. Conversely, teams with a high level of trust at the end of projects had an initial focus in social communication, which later diminished to make room for procedural and task-focused interactions. This result is consistent with the findings from the study by Wang & Redmiles [16], who analyzed the IRC chat logs of two large OSS projects and observed that the amount of social interactions between developers tend to decrease and even disappears once trust and cooperation are fully developed.

Therefore, building on the model of antecedents of affective trust, we hypothesize that (see Figure 1):

H_1 – *In distributed projects, the amount of social communication decreases over time, as affective trust mutually develops between developers throughout collaboration.*

Treinen & Miller-Frost [12] observed that the development of mutual trust between distant sites at the beginning of a project turned out to be more important than resolving technical issues. In fact, increased trust allowed remote sites to anticipate future issues from afar (e.g., conference call), thus resulting in increased overall efficiency. However, to the best of our knowledge, no previous study provided direct evidence that connects affective trust to project performance. Establishing a cause/effect relationship between developers' trust and project performance is a challenging task as many other confounding factors, such as project type or individual skills may interfere along the process. In this study, we intend to overcome this limitation by approximating the overall performance of a project (e.g., requirements completion, productivity, duration) with the history of successful collaborations occurring between project developers. By *successful collaborations* we indicate situations where (at least) two developers work together and their cooperation is successful because it yields project advancement (e.g., by fixing bugs or adding new features). Through such a fine-grain unit of analysis, we aim to measure 'more directly' how affective trust facilitates cooperation.

Consequently, we hypothesize that (see Figure 1):

H_2 – *In distributed projects, there is a positive correlation between the amount of affective trust mutually established between developers and the probability of observing a successful collaboration among them.*

These hypotheses are meant to be investigated by means of an empirical study, the design of which is detailed in the next section.

3. EMPIRICAL STUDY

Based on the theory of trust antecedents, we intend to perform a study that will provide a measurement of how affective trust can improve performance in a distributed project. In particular, we intend to look at the content of the interactions occurring in large open-source projects to identify, through sentiment analysis, the information elements on which affective trusts grows. As a first step, we intend to focus on studying affective trust development between pairs of developers rather than groups. This decision is motivated by a recent quantitative study [13], which found that two developers discover an optimal number of defects during code reviews.

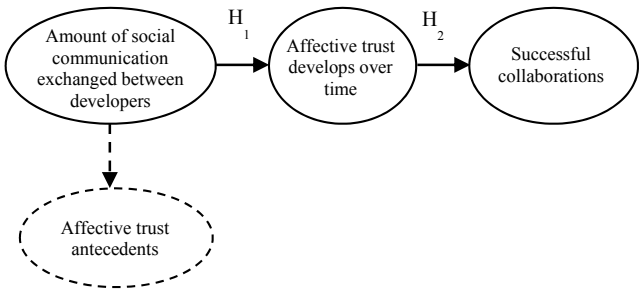

Figure 1. The proposed research model.

3.1 Construct Variables

Here we describe the constructs to conceptualize the elements in our hypotheses. The constructs described in the following relate to the concept of *pull request*, enabled by Git and popularized by collaborative coding platforms, such as GitHub and Bitbucket.

Modern distributed software projects support their workflow and coordinate remote work through version control systems. A pull request is one the preferred way of submitting contributions to a project using a distributed version control system such as Git. Pull-based development is a distributed software development model with a distinct way of collaborating. In this model, the project's main repository is not shared among developers. Instead, developers contribute by forking (i.e., cloning) the repository and making their changes independently from each other. When a set of changes is ready to be submitted to the main repository, a potential *contributor* creates a pull request. Then, an *integration manager*, one of the core developers, is assigned responsible to inspect the changes and integrate them into the project's main development line. The role of the integration manager is crucial to ensure project quality. After a contribution has been received, the integrators must *close* a pull request deciding whether it is suitable for the project – i.e., the pull request is *accepted* and changes are merged into the project's main repository – or incorrect – i.e., the pull request is *declined* and changes are rejected. Closed pull requests, whether accepted or declined, require that a consensus is reached through discussion. OSS projects typically rely on collaborative development platforms such as GitHub and Bitbucket, which make it easier for developers to collaborate through pull requests as they provide a user-friendly web interface for discussing proposed changes before integrating them into the project source code.

Previous research has typically relied on self-reported data, typically questionnaires, to measure trust on a given scale [1], [3],[14],[15]. Conversely, for the sake of providing a more reliable measure, here we define the construct of *amount of affective lexicon* (e.g., statements conveying affective states such as *'I hate …'* or '*I like that…'*), in the comments exchanged between developers during pull requests, as an approximation of the level of the existing level of affective trust.

Thus, H_1 is rephrased as follows:

H_1' – *In distributed projects, the amount of affective lexicon in pull request comments decreases over time, as affective trust mutually develops between developers.*

Regarding the second hypothesis, we intend to collect evidence that affective trust is an antecedent of successful contributions between project developers. Recent work identified that and 'social distance' between developers can be used to predict whether a contribution will be accepted or not [14]. Therefore, we define the construct of *accepted pull requests* to represent a successful collaboration between the pull request contributor and the integration manager. In other words, we argue that affective trust acts as a catalyst for accepted pull requests that, in turn, boost project performance by contributing new features or resolving bugs.

Therefore, H_2 is rephrased as follows:

H_2' – *In distributed projects, the larger the amount of affective lexicon in prior pull-request comments, the larger the chances for the current pull request to be accepted.*

3.2 Measurements

In this section, we illustrate how we intend to measure the amount of affective trust between any two pair of developers in a project, starting from the amount of affective lexicon exchanged in pull-request comments.

The amount of affective trust will be computed using SentiStrength,[1] a sentiment-analysis tool capable of dealing with short written informal text, including abbreviations, intensifiers and emoticons. Because a sentence can convey mixed sentiment, the tool provides both positive and negative overall sentiment scores for an input text, computed as the maximum positive and negative scores assigned to each sentence. In our case, given a thread of comments in a pull request, for all the comments exchanged between any identified developer pairs, SentiStrength will output two scores in the interval $[\pm 1, \pm 5]$, which will be summed up and considered as a measure of the affective lexicon in the pull request.

Thus, the overall level of affective trust between the pull-request contributor and the integrator will be given by summing the affective lexicon between the same pair of developers achieved through all the previous pull requests.

4. FUTURE WORK

Our analysis will focus on large OSS projects that are rich in pull request-related discussions among contributors and integration managers. For example, large successful projects like Ruby on Rails[2] and Node.js[3] typically have several hundred open pull requests being reviewed in parallel.

The selected projects data will be obtained through GHTorrent,[4] a project that offers both an on-line queriable and offline downloadable mirror of data offered through the GitHub REST API.

One problem with using the SentiStrength or any other sentiment analyzers, is that so far not tool has been trained on software engineering documents and language. As such, using the tool unmodified might lead to misestimating emotions or the incorrect identification of whether emotion exists at all in the communication.

5. REFERENCES

[1] Al-Ani B. & Redmiles D. (2009). In Strangers We Trust? Findings of an Empirical Experiment of Distributed Teams. In: *Proc. of 4th IEEE Int'l Conf. on Global Software Engineering (ICGSE '09)*, pp. 121-130. doi:10.1109/ICGSE.2009.20.

[2] Al-Ani B., Redmiles D., de Souza C.R.B., Prikladnicki R., Marczak S., Lanubile F. & Calefato F. (2013) "Trust in Virtual Teams: Theory and Tools." In *Proc. of Int'l Conf. on Computer-Supported Cooperative Work (CSCW'13)*, pp. 301-306. doi:10.1145/2441955.2442029.

[3] Calefato F., Lanubile F. & Novielli N. (2015) The Role of Social Media in Affective Trust Building in Customer-Supplier Relationships. *Electronic Commerce Research*, 15(4), 453-482, doi:10.1007/s10660-015-9194-3.

[4] Calefato F. & Lanubile F. (2103) SocialCDE: A Social Awareness Tool for Global Software Teams. *Proc. of 9th ESEC/FSE'13*, 587-590, doi:10.1145/2491411.2494592.

[5] Calefato F. & Lanubile F. (2012) Augmenting Social Awareness in a Collaborative Development Environment." *5th Int'l Workshop on Cooperative and Human Aspects of Software Engineering (CHASE'12)*, 12-14, doi:10.1109/CHASE.2012.6223009.

[6] Calefato F., Lanubile F., Sanitate N. & Santoro G. (2011) Augmenting Social Awareness in a Collaborative Development Environment." *4th. Int'l Workshop on Social Software Engineering (SSE'11)*, 39-42, doi:10.1145/2024645.2024656.

[7] Gousios G, Zaidman A, Storey M.-A & van Deursen A. (2015) Work Practices and Challenges in Pull-Based Development: The Integrator's Perspective," *Proc. 37th Int'l Conf. Software Engineering (ICSE'15),* 358-368, doi:10.1109/ICSE.2015.55.

[8] Guzzi A, Bacchelli A, Lanza M, Pinzger M & van Deursen A (2013) Communication in open source software development mailing lists. *Proc. of the 11th Working Conference on Mining Software* Repositories *(MSR 2013)*, 277–286,

[9] Hacker, S. K., Israel, J. T. & Couturier, L. (1999). Building Trust in Key Customer-Supplier Relationships. *The Performance Center and Satisfaction Strategies.* http://www.satisfactionstrategies.com/paper4.pdf. Accessed 12 Jan. 2016.

[10] Jarvenpa S.L. & Leidner D.E. (1999) Communication and Trust in Global Virtual Teams. *Journal of Organization Science*, 10(6), 791-815.

[11] Lanubile F., Calefato F. Ebert C. (2013) Group Awareness in Global Software Engineering. *IEEE Softw.* 30(2), 18-23. doi:dx.doi.org/10.1109/MS.2013.30

[12] Treinen J.J. & Miller-Frost S.L. (2006) Following the sun: case studies in global software development, *IBM Syst. J.*, 45(4), 773-783.

[13] Tsay J., Dabbish L. & Herbsleb J. (2014) Influence of social and technical factors for evaluating contribution in GitHub. *Proc. of 36th Int'l Conf. on Software Engineering* (ICSE 2014), 356-366, doi:10.1145/2568225.2568315.

[14] Rusman E., van Bruggen J., Sloep P. & Koper R. (2010). Fostering trust in virtual project teams: Towards a design framework grounded in a TrustWorthiness Antecedents (TWAN) schema. *International Journal of Human-Computer Studies,* 68(11), 834-850. doi:10.1016/j.ijhcs.2010.07.003.

[15] Schumann J., Shih P., Redmiles D. & Horton G. (2012). Supporting Initial Trust in Distributed Idea Generation and Evaluation. In: *Proceedings International ACM SIGGROUP Conference on Supporting Group Work (GROUP '12)*, pp. 199-208, doi:10.1145/2389176.2389207.

[16] Wang Y, & Redmiles D. (2015). Cheap talk, cooperation, and trust in global software engineering: An evolutionary game theory model with empirical support. *Empirical Software Engineering*,1-35, doi:10.1007/s10664-015-9407-3.

[17] Wilson J.M., Strausb, S.G. & McEvily, B. (2006). All in due time: The development of trust in computer-mediated and face-to-face teams. *Organizational Behavior and Human Decision Processes*, 99(1), 16–33. doi:10.1016/j.obhdp.2005.08.001.

[1] http://sentistrength.wlv.ac.uk

[2] https://github.com/rails/rails/pulls

[3] https://github.com/nodejs/node/pulls

[4] http://ghtorrent.org