Estimating Trust in Virtual Teams
A Framework based on Sentiment Analysis

Guilherme A. Maldonado da Cruz, Elisa Hatsue Moriya Huzita and Valéria D. Feltrim
Informatics Department, State University of Maringá, Maringá, Paraná, Brazil

Keywords: Trust, Versioning System, Sentiment Analysis, Virtual Teams, Global Software Development.

Abstract: The advance in technology has enabled the emergence of virtual teams. In these teams, people are in different places and possibly over different time zones, making use of computer mediated communication. At the same time distribution brings benefits, there are some challenges as the difficulty to develop trust, which is essential for efficiency in these teams. In this scenario, trust information could be used to allocate members in a new team and/or, to monitor them during the project execution. In this paper we present an automatic framework for detecting trust between members of global software development teams using sentiment analysis from comments and profile data available in versioning systems. Besides the framework description, we also present its implementation for the GitHub versioning system.

1 INTRODUCTION

Software development using virtual teams characterizes distributed software development or global software development (GSD) when the distance between members comprises continents. It aims at providing benefits such as: low costs, proximity to the market, innovation and, access to skilled labor (O'Conchuir et al., 2006). However, geographic distribution and cultural differences bring some challenges as well, mainly in communication, which depends mostly on computer mediated communication (CMC).

One of the challenges faced by virtual teams and therefore GSD is the trust among team members. There are several studies that show the importance of trust for GSD teams (Kuo and Yu, 2009; Al-Ani et al., 2011; Pangil and Chan, 2014). Trust is related to the efficiency of the team, since high-trust teams can achieve their goals with less effort than low-trust teams. So, in this context, information about trust among people can be used for team recommendation and/or to monitor the relationship among members.

Some models have been proposed to estimate trust among people based on trust evidences, such as number of interactions, success of these interactions and similarity among people (Skopik et al., 2009; Li et al., 2010). We consider trust evidence something that indicates the existence of trust or that happens when there is trust among people.

Information used by trust models can be extracted, for example, from social networks. In general, it refers to the amount of interactions, evaluation of these interactions and their success. However, in a working environment people may not feel free to provide assessments of co-workers. Besides that, when the number of interactions is high, people may start to provide incorrect ratings, leading to incorrect trust estimation. Skopik et al. (2009) developed a set of metrics to analyze the success of an interaction. These metrics eliminates the need for feedback, however, they are domain dependent and ignore subjectivity, which is one of the characteristics of trust.

In this paper we present a framework to estimate trust among members of GSD teams. It extracts trust evidences observed in member interactions inside versioning systems, without human intervention and using sentiment analysis.

Sections 2, 3 and 4 present the concepts of GSD, trust and sentiment analysis, used in the development of the proposed framework. Section 5 describes the framework and Section 6 presents conclusions and also directions for future works.

2 GLOBAL SOFTWARE DEVELOPMENT

According to O'Conchuir et al. (2006) GSD is a collaborative activity, which can be characterized by

Cruz, G., Huzita, E. and Feltrim, V.
Estimating Trust in Virtual Teams - A Framework based on Sentiment Analysis.
In *Proceedings of the 18th International Conference on Enterprise Information Systems (ICEIS 2016) - Volume 1*, pages 464-471
ISBN: 978-989-758-187-8

having members from different cultures and organizations, separated by time and space using CMC to collaborate. This team organization aims at providing benefits, such as: reduced development costs, follow-the-sun development, modularization of labor, access to skilled labor, innovation, best practices and knowledge sharing and proximity to the market.

Despite its benefits, GSD also brings challenges that add to those already existing in virtual teams, such as: strategic problems, cultural problems, inadequate communication, knowledge management, processes management and technical problems. Among these challenges is trust (Khan, 2012).

In this context, trust is important, mainly in GSD, due to members' inability to check what other members are doing by just watching (Jarvenpaa et al., 1998). Thus, trust reduces the risk and cost of monitoring (Striukova and Rayna, 2008). It also impacts in information sharing (Pangil and Chan, 2014), cohesion (Kuo and Yu, 2009) and cooperation (Striukova and Rayna, 2008).

3 TRUST

Trust has been studied in many fields, such as psychology, philosophy and economics. Based on definitions of different areas, Rusman et al. (2010, p.836) defined trust as:

> a positive psychological state (cognitive and emotional) of a trustor (person who can trust/distrust) towards a trustee (person who can be trusted/distrusted) comprising of trustors positive expectations of the intentions and future behavior of the trustee, leading to a willingness to display trusting behavior in a specific context.

This definition presents one of the trust properties, which is context specificity. Trust is also dynamic, non-transitive, propagative, composable, subjective, asymmetrical, events sensitive and self-reinforcing (Sherchan et al., 2013).

Before deciding to trust, a person evaluates the trustworthiness of the person to be trusted and the risk involved, so that if she chooses to trust, she became vulnerable positively and negatively to the trusted person, assuming the risk (Rusman et al., 2010). Therefore, the higher the trustworthiness, the higher the chance to be trusted. Rusman et al. (2010) define trustworthiness antecedents as attributes used to evaluate trustworthiness and divided them into five categories: (i) communality, (ii) ability, (iii) benevolence, (iv) internalized norms and (v) accountability.

3.1 Trust Evidence

Through a literature review we could not find an exact formula to determine whether there is trust among team members. However, some studies indicate behaviours and characteristics that serves as evidence of trust existence. For example, Jarvenpaa et al. (1998), conducted a qualitative study based on observations of teams with a high level of trust and teams with low level of trust. The authors observed common characteristics to high-trust teams that did not appear in low-trust teams, enumerated as: proactivity, task oriented communication, positive tone, rotating leadership, task goal clarity, roles division, time management, feedback and intensive communication.

Besides Jarvenpaa et al.'s (1998) work, we found other studies identifing teams characteristics which serve as evidence of trust. The list below sums up the trust evidences found in our literature review:

- Initiation and response: initiations are defined as questions or statements that lead the receiver to provide a relevant response. Iacono and Weisband (1997) used this characteristic to measure trust.
- Motivation: According to (Paul and He, 2012) motivation and trust are highly correlated.
- Knowledge sharing: Paul and He (2012) showed that the greater the trust among people, the greater is information sharing between them.
- Knowledge acceptance: people tend to accept knowledge of who they trust (Al-Ani et al., 2011).
- Trustworthiness: trust and trustworthiness are highly correlated (Jarvenpaa et al., 1998).
- Proactivity: high-trust team members are proactive, volunteering for roles and showing initiative (Jarvenpaa et al., 1998).
- Task oriented communication: in high-trust teams most conversations are about tasks to be completed (Jarvenpaa et al., 1998).
- Positive tone: high-trust teams tend to show enthusiasm in their conversations, praising and encouraging each other (Jarvenpaa et al., 1998).
- Task goal clarity: high-trust teams tend to discuss their goals, and when in doubt, they seek coordinators for clarification instead of making assumptions (Jarvenpaa et al., 1998).
- Rotating leadership: many members show leadership traits, and according to project needs, they assume the leadership as necessary (Jarvenpaa et al., 1998).
- Role division: team members assume roles in their project and show results of their work so others can provide feedback (Jarvenpaa et al., 1998).

- Time management: high-trust teams discuss deadlines, establish milestones and care to fulfill them (Jarvenpaa et al., 1998).
- Feedback and intense communication: high-trust teams display intense communication and feedback about team members' work (Jarvenpaa et al., 1998; Rusman et al., 2010; Kuo and Yu, 2009).
- High Performance: trust is positively related with team cohesion (Kuo and Yu, 2009), commitment, satisfaction and performance (Mitchell and Zigurs, 2009).
- Output quality: trust is positively related with output quality (Khan, 2012).
- Common vocabulary: when there is trust between people they tend to share a common vocabulary in CMC (Scissors et al., 2008).

Besides the evidences described above, Khan (2012) considered authority delegation, enthusiasm and high quantum of work as signs of trust. To Rusman et al. (2010), resources sharing, task division and delegation also occur when there is trust among team members.

3.2 Trust Models

In our literature review we found two models to estimate trust among people. The framework proposed by Skopik et al. (2009) aims at determining trust automatically, without the need for explicit feedbacks. The framework generates a graph in which nodes represent both services and people, and edges represent the trust value between them. Trust values are derived from the number of successful interactions relative to the total number interactions. Successful interactions are computed by a set of metrics, such as occurrence of errors in services.

The downside of Skopik et al.'s (2009) work is that, by relying on metrics, it ignores subjectivity that is intrinsic to trust by treating all people equally. For instance, if a service takes up to 30 seconds to respond an interaction, one person may consider it a success, while some other person may consider it a failure, even if it spends 10 seconds. Thus, this type of metric fails to capture such subjectivity.

The trust model proposed by Li et al. (2010) aims at assisting users of E-commerce in choosing best sellers. In their work, trust is estimated based on assessments made by users after interactions, and in the absence of interactions, on the similarity between assessments provided by them. It generates a user graph in which edges and weights represent the trust among them.

4 SENTIMENT ANALYSIS

Sentiment analysis have gained a lot of attention by the research community in the last decade, and have found its application in almost every business and social domain (Liu, 2012).

One of the tasks focused by sentiment analysis systems is to determine for a given text the polarity of the sentiment expressed: if it is positive, neutral or negative. The text unity used in the analysis determines its level, which usually falls into one of the three: (i) document level, (ii) sentence level and (iii) entity-aspect level (Liu, 2012).

Sentiment analysis has been also applied in the context of software development research. Guzman (2013) used sentiment analysis to capture emotion during diferent software development phases and to provide emotional climate awareness. Borbora et al. (2013) considered the sentiment expressed in communications as an indicator of the presence/absence of trust among stakeholders. Zhang et al. (2009) suggested the use of sentiment analysis to get a better understanding of trust among users. In fact, as presented in Section 3.1, one of the trust evidences is positive tone in communication, which can be directly captured by sentiment analysis tools.

5 THE FRAMEWORK

As previously discussed, virtual teams need trust among members in order to achieve their goals, since trust affects team's efficiency (Pangil and Chan, 2014). Trust models can be used to monitor trust among team members. However, some models require users to provide evaluation of others, or assume that there are means of informing if interactions were positive or not. The problem is that, in GSD teams, members can not feel free to evaluate co-workers. Even if it were not an issue, there would be a lot of interactions and members could end up getting tired of evaluating each interaction, providing nonsense evaluations that compromise the outcome of the models (Li et al., 2010).

In this context, we propose a framework to automatically estimate trust among GSD team members. The framework collects trust evidences observed in member interactions inside versioning systems, without human intervention and using sentiment analysis. Figure 1 presents the framework in terms of its inputs, used techniques, trust evidences considered and its output. The remaining of this section describes the framework focusing on its characteristics, how it works, its and design and implementation.

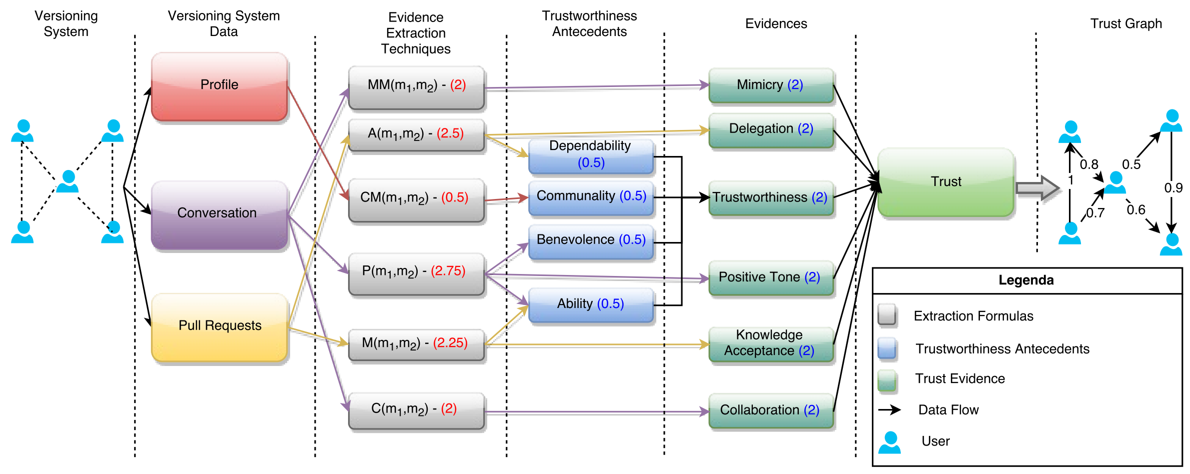

Figure 1: Proposed framework to estimate trust existence.

5.1 Characteristics

The main characteristics of the framework are:

a) It uses versioning systems as data source. Versioning systems are tools heavily used in software development and therefore in GSD (Robbes and Lanza, 2005). Particularly, the versioning systems that interest us are the ones that allow their users to perform commits and comment other's commits.

b) It uses trust evidences extracted from versioning system data (comments, commit state and user profile) to estimate trust among members of a project. As we could not extract all the trust evidences described in Section 3.1, the ones considered in the framework are: mimicry (common vocabulary), delegation, trustworthiness, positive tone, knowledge acceptance and collaboration. To extract the trustworthiness of a member we estimate four of the five trustworthiness antecedents presented in Rusman et al. (2010): dependability, communality, benevolence and ability.

c) It is automatic. Once it is set, the framework retrieves data without human intervention, estimates the existence of trust according to the temporal window and update estimated values according to the update frequency.

d) It preserves the subjectivity inherent to trust by using sentiment analysis and considering mimicry as a trust evidence. Sentiment analysis values and mimicry are extracted from how members write their comments, which is personal for each member. Thus, the framework takes into account subjectivity as it infers trust evidences from personal data. With sentiment analysis we can infer positive tone directly. Benevolence can be inferred since it was defined in Rusman et al. (2010) as the positive attitude and courtesy displayed by the trustee. Ability can also be inferred from sentiment analysis while the polarity of every comment may be seen as a feedback about the commit, and the commit in turn is the result of a member's ability to solve a problem. Therefore, we use the polarity of comments as a feedback about members' ability.

e) It updates evidences and trust values over time. It considers a time window to perform the extractions, and from time to time it moves this window, discarding old data, retrieving new ones and updating the values according to the data in the current time window.

f) It generates an initial graph of relations. This graph tells in which pull requests team members interact. The initial graph has an edge between a pair of members if they interact in a pull request. During execution, we add partial and final edges to the initial graph of relations. Partial edges keep partial values that are used to calculate final edges. One final edge is added for each evidence extraction technique. For instance, a pair of nodes may have many partial edges keeping the polarity of one comnent each, and one final edge keeping the rate of positive comments.

g) It generates a trust graph with its estimative of trust existence between each pair of members that interacted in at least one pull request. In this graph, nodes represent members of a project and edges represent the existence of trust between them. The weight of the edge, ranging from 0 to 1, displays the probability that there is trust between two members. The closer to 1 the edge value is, the higher is the chance of existing trust between those two members.

Comparing our framework with the works presented in Section 3.2, we also use interactions like them both (Skopik et al., 2009; Li et al., 2010). Unlike Li et al. (2010), we removed the need for assessments, but kept the time factor by using a temporal window. We wanted it to be automatic like in Skopik et al. (2009), but we did not use metrics. Instead, we used mimicry and sentiment analysis in order to preserve subjectivity. We also added more trust evidences found in the literature in order to enrich the information used to estimate trust.

5.2 Design and Implementation

We designed and implemented an instance of our framework[1] to work with the GitHub versioning system. As presented in Figure 2, the framework is composed of four components:

Graph This component provides the framework with graphs that will be used as the initial graph of relations and the trust graph. By changing this component, we are able to alter how the framework keeps its data. The default implementation uses graphs.

VS Data Extractor This component extracts data from the versioning system. It extracts profile information from members in the project, pull requests information and conversations. In our implementation for GitHub we used the GitHub Java API[2]. Besides extracting data, this component also generates the initial graph of relations.

Evidence Analyzer This component provides classes implementing the *EvidenceAnalyser* interface representing an evidence extraction technique. Each one of these classes will analyze data extracted from versioning system and generate a value that is stored in the graph of relations and used to estimate trust existence. We provided six evidence extraction techniques: mimicry, assignments, communality, polarity, merges and collaboration. These evidence extraction techniques are formulas described further in this section. The addition of more evidences to the framework requires only a new implementation of *EvidenceAnalyser* interface to a new evidence extraction technique.

[1]Available at https://github.com/Tulivick/Trust-Framework

[2]https://github.com/eclipse/egit-github/tree/master/org.eclipse.egit.github.core

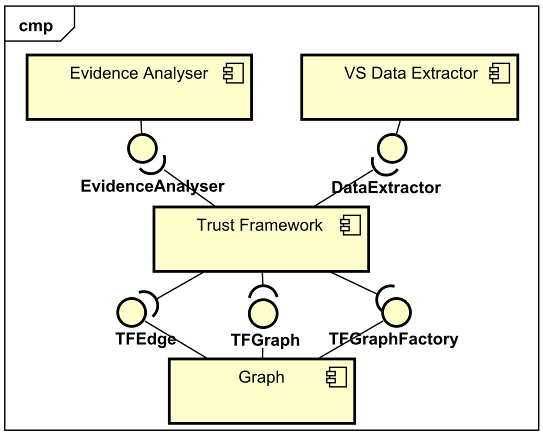

Figure 2: Component diagram for trust framework.

Trust framework This is the main component. It provides ways to configure and use the framework. This component is responsible to get data from `VS Data Extractor`, and transmit it to `Evidence Analyzer` with the graph of relations, so the graph can be updated. From the graph of relations, this component generates the trust graph using the formula described further in this section.

Note that by providing new implementations for `VS Data Extractor` we are able to extend the framework to other versioning systems. However, as each versioning system may have different data, the `Evidence Analyzer` is bound to the `VS Data Extractor` component. If one wants to provide support to another versioning system, it may be necessary to replace `Evidence Analyzer` by one that supports the new versioning system. It is also possible to extend the set of considered evidences by implementing other classes that extend *EvidenceAnalyser* interface.

As mentioned before, in order to extract evidence values, we used a set of formulas that we named evidence extraction techniques on Figure 1. To extract conversations mimicry, we calculate how similar are the vocabularies used in conversations for a pull request. We calculate the similarity of two comments vocabularies by using cosine similarity on word frequency. In Equation 1, $SC(c_1,c_2)$ is the cosine similarity between two comments, and FV is the words frequency array for each comment.

$$SC(c_1,c_2) = \frac{FV_1 \cdot FV_2}{\|FV_1\|\|FV_2\|} \quad (1)$$

The member m_1 mimicry value for the member m_2, $MM(m_1,m_2)$ is the average of the similarities between m_1's comments and m_2's comments that precede in the same pull request. In Equation 2 PR_{12} is the set of pull requests where m_1 and m_2 interact, C_1

is the comments set of m_1 in pull request pr_i and C_{2j} is the comments set of m_2 preceding the comment c_j.

$$MM(m_1,m_2) = \frac{\sum_{pr_i=1}^{|PR_{12}|}\sum_{c_j=1}^{|C_1|}\sum_{c_k=1}^{|C_{2j}|} SC(c_j,c_k)}{\sum_{pr_i=1}^{|PR_{12}|}\sum_{c_j=1}^{|C_1|}\sum_{c_k=1}^{|C_{2j}|} 1} \quad (2)$$

Communality is calculated by averaging the similarity of three GitHub user profile attributes: (I) followed users, (II) watched projects, and (III) location. (I) and (II) are calculated using Jaccard similarity (Equation 3) where C_1 and C_2 are evaluated sets, in this case the followed users and watched projects for both members. (III) in turn uses a variant of the Euclidean distance (Dodd et al., 2013, Equation 4) on Geert Hofsted index values (Hofstede et al., 2010). Indexes of Geert Hofsted characterize the culture of a region using six indexes: power distance, individualism, masculinity, uncertainty avoidance, long-term guidance and indulgence. In Equation 4 L_i is a location, K is the amount of indexes, I_{ik} is the value of the index k to a location L_i and V_k is the variance of the index k. If the indexes do not exist for a particular location, we will consider biggest the distance possible between two locations.

$$JS(C_1,C_2) = \frac{C_1 \cap C_2}{C_1 \cup C_2} \quad (3)$$

$$ED(L_1,L_2) = \sqrt{\sum_{k=1}^{K} \frac{(I_{1k} - I_{2k})^2}{V_k}} \quad (4)$$

Therefore, the communality $CM(m_1,m_{,2})$ between members m_1 and m_2 is given by Equation 5 where F_i, W_i and L_i are respectively the set of followed users, watched projects and m_i's location.

$$CM(m_1,m_2) = \frac{JS(F_1,F_2) + JS(W_1,W_2) + \frac{1}{1+ED(L_1,L_2)}}{3} \quad (5)$$

We used comments polarities to estimate benevolence, ability and positive tone. The polarity value between two members is given by Equation 6, where $P(m_1,m_2)$ is member m_1 polarity value for the member m_2, C_{+12} is the amount of comments with positive polarity from m_1 to m_2 and C_{12} is the total amount of comments from m_1 to m_2. Comments from m_1 to m_2 are comments that m_1 wrote in m_2's pull request or comments where m_1 mentioned m_2. Note that there are pull requests created by m_1 where m_2 did not interact, however these are not considered.

$$P(m_1,m_2) = \frac{C_{+12}}{C_{12}} \quad (6)$$

Dependability and delegation can be observed through the assignments of a member by another in a pull request. The assignment value of a member m_1 to a member m_2 is 1 if m_1 assigned at least one pull request to m_2 or 0 otherwise. Equation 7 calculates the assignment value for m_1 to m_2, $A(m_1,m_2)$, based on the amount of pull requests assigned to m_2 by m_1, PRs_{12}.

$$A(m_1,m_2) = \begin{cases} 1, & \text{if } PRs_{12} \geq 1 \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

We infer values for knowledge acceptance and ability of a member from the amount of pull requests that were merged. In Equation 8, $M(m_1,m_2)$ is the proportion of pull requests created by m_2 in which m_1 and m_2 interacted, that were merged, PR_{a12} is the amount of pull request created by m_2 in which m_2 and m_1 interacted, that were merged, PR_{12} is the amount of pull request created by m_2 in which m_1 and m_2 interacted.

$$M(m_1,m_2) = \frac{PR_{a12}}{PR_{12}} \quad (8)$$

We estimate collaboration as the proportion of interactions as given by Equation 9. In this equation $C(m_1,m_2)$ is the proportion of interactions between m_1 and m_2 out of the total interactions of m_1. I_{12} is the number of interactions between m_1 and m_2, and I_1 is the amount of m_1 interactions with everyone.

$$C(m_1,m_2) = \frac{I_{12}}{I_1} \quad (9)$$

Finally, we estimate values of trust among members using Equation 10, which is the weighted average of the formulas listed above. The best way to set the weights α_i would be through the use of history data from previous projects to learn the weights. However, we are not aware of any database with trust information among members in versioning systems. Thus we defined weights based on the number of evidences calculated through the use of each formula presented above.

$$\begin{aligned} T(m_1,m_2) = \\ \frac{\alpha_1 MM(m_1,m_2) + \alpha_2 CM(m_1,m_2) + \alpha_3 P(m_1,m_2)}{\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6} \\ + \frac{\alpha_4 A(m_1,m_2) + \alpha_5 M(m_1,m_2) + \alpha_6 C(m_1,m_2)}{\alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5+\alpha_6} \end{aligned} \quad (10)$$

First we consider that every evidence has the same weight. We choose weight 2, because we will propagate it to the trustworthiness antecedents and evidence extraction techniques, and if it were lower we would obtain many decimal places.

We consider that each trustworthiness antecedent has the same weight to calculate trustworthiness, so by propagating the weight 2 from trustworthiness to its antecedents we obtain a weight of 0.5 for each.

By propagating the weights from antecedents and evidences we obtain the values $\alpha_1 = 2$, $\alpha_2 = 0.5$, $\alpha_3 = 2.75$, $\alpha_4 = 2.5$, $\alpha_5 = 2.25$ and $\alpha_6 = 2$ presented in front of each formula on Figure 1.

As an example, we will propagate the weights to $MM(m_1,m_2)$ and $A(m_1,m_2)$. $MM(m_1,m_2)$ infers only mimicry that has weight 2, thus when we propagate its weight to the formula, that gains weight 2 also. $A(m_1,m_2)$ in turn infers delegation with weight 2, and dependability with weight 0.5, thus by propagating this weights to the formula it gains weight 2.5. By doing this to each formula we obtained the α values.

In Figure 1, the numbers in parentheses represent the weights of each evidence and trustworthiness antecedents, except the ones appearing at the evidence extraction techniques, which are the α values obtained by propagating the weights from evidences and trustworthiness antecedents.

As mentioned at the beginning of this subsection, each formula is coded in a class implementing the `EvidenceAnalyser` interface in the `EvidenceAnalyser` component. In order to implement these classes, we used the following APIs: Lucene[3] to generate word frequency count for comments, Sentistrength[4] to retrieve the polarity for every comment, AlchemyApi[5] to extract comments targets and Google Maps Geocoding API[6] to discover from which country each user‘s address was.

5.3 How It Works

To start using the framework we need to configure it. This is done by informing a target project (owner/repository), the evidence extraction techniques to be used and their weights, the size of the temporal window, the update frequency, and a graph factory.

Once it starts running, the framework extracts project data from GitHub. These data are in turn processed using the formulas described in the previous section to extract the trust evidences. The data retrieved from GitHub are delivered to each instance of *EvidenceAnalyser* interface. This instances will add partial and final edges to the initial graph of relations. Combining final edges values through Equation 10 we estimate trust existence among members. This estimate is provided by means of a trust graph.

With the trust graph in hands, we can, for example, use trust values to suggest members to a team that has a higher chance of having a high level of trust. In addition, as the framework process the latest comments and automatically updates the values of trust, it enables us to monitor trust among members, so that the manager can take actions when negative changes in the teams' trust are perceived.

[3]https://lucene.apache.org/

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

[5]http://www.alchemyapi.com/

[6]https://developers.google.com/maps/

6 CONCLUSIONS

The efficiency of a GSD team is directly tied to trust among team members. The higher the trust is, the lower project costs are. Trust also increases communication and facilitate cooperation, coordination, knowledge and information sharing, which improve the quality of generated products.

Motivated by the importance of trust in these teams, this work presented an automatic framework to estimate trust existence among members of a GSD team. It uses versioning systems, a collaborative tool used in software development, as data source. To design the framework, we used trust evidences presented in the literature that can be extracted from versioning systems data. One of the main features of the proposed framework is the use of sentiment analysis to extract some of these evidences, for example, the positive tone of the conversations.

The main contribution of this paper is in the mapping of trust evidences and elements of trust models that can be captured using sentiment analysis. We also contributed with an implemented instance of the framework that works with GitHub. We expect our framework to provide a better estimative of trust existence than general automatic models in the literature since it uses sentiment analysis and a rich set of evidences. By employing sentiment analysis, we have added subjectivity to our estimative, which is an important characteristic of trust. GSD managers can benefit from our framework to create teams with higher trust levels. With our framework it is also possible to monitor trust level variations, so actions can be taken by the project manager when trust level decreases.

As we do not have a GitHub dataset annotated with trust information, we considered all weights the same. In order to better calibrate the weights we are conducting a survey with experienced people in GSD to aid us determine the weight of each evidence and validate the formulas we presented. The results obtained until now, are promising.

As future work we foresee: (i) the addition of other trust evidences to the framework, (ii) the conclusion and analyzes of the results for our survey, which may lead to an improvement of the framework and the conduction of a new survey, and (iii) the monitoring of a real project, allowing us to collect trust information about team members in order to compare with the results given by our framework.

REFERENCES

Al-Ani, B., Wilensky, H., Redmiles, D., and Simmons, E. (2011). An understanding of the role of trust in knowledge seeking and acceptance practices in distributed development teams. In *6th IEEE International Conference on Global Software Engineering (ICGSE)*, pages 25–34.

Borbora, Z. H., Ahmad, M. A., Oh, J., Haigh, K. Z., Srivastava, J., and Wen, Z. (2013). Robust features of trust in social networks. *Social Network Analysis and Mining*, 3(4):981–999.

Dodd, O., Frijns, B., and Gilbert, A. (2013). On the role of cultural distance in the decision to cross-list. *European Financial Management*, pages n/a–n/a.

Guzman, E. (2013). Visualizing emotions in software development projects. In *First IEEE Working Conference on Software Visualization (VISSOFT)*, pages 1–4.

Hofstede, G., Hofstede, G. J., and Minkov, M. (2010). *Cultures and Organizations: Software of the Mind*. McGraw-Hill USA, New York, 3 edition. An optional note.

Iacono, C. and Weisband, S. (1997). Developing trust in virtual teams. In *Proceedings of the Thirtieth Hawaii International Conference on System Sciences, 1997*, volume 2, pages 412–420 vol.2.

Jarvenpaa, S. L., Knoll, K., and Leidner, D. E. (1998). Is anybody out there? antecedents of trust in global virtual teams. *Journal of Management Information Systems*, 14(4):pp. 29–64.

Khan, M. S. (2012). Role of trust and relationships in geographically distributed teams: exploratory study on development sector. *International Journal of Networking and Virtual Organisations*, 10(1):40–58.

Kuo, F.-y. and Yu, C.-p. (2009). An exploratory study of trust dynamics in work-oriented virtual teams. *Journal of Computer-Mediated Communication*, 14(4):823–854.

Li, J., Zheng, X., Wu, Y., and Chen, D. (2010). A computational trust model in c2c e-commerce environment. In *Proceedings - IEEE International Conference on E-Business Engineering, ICEBE*, pages 244 – 249, Shanghai, China.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

Mitchell, A. and Zigurs, I. (2009). Trust in virtual teams: Solved or still a mystery? *ACM SIGMIS Database*, 40(3):61–83.

O'Conchuir, E., Holmstrom, H., Agerfalk, P., and Fitzgerald, B. (2006). Exploring the assumed benefits of global software development. In *International Conference on Global Software Engineering, 2006. ICGSE '06*, pages 159–168.

Pangil, F. and Chan, J. (2014). The mediating effect of knowledge sharing on the relationship between trust and virtual team effectiveness. *Journal of Knowledge Management*, 18(1):92–106.

Paul, S. and He, F. (2012). Time pressure, cultural diversity, psychological factors, and information sharing in short duration virtual teams. In *45th Hawaii International Conference on System Science (HICSS)*, pages 149–158.

Robbes, R. and Lanza, M. (2005). Versioning systems for evolution research. In *Eighth International Workshop on Principles of Software Evolution*, pages 155–164.

Rusman, E., van Bruggen, J., Sloep, P., and 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.

Scissors, L. E., Gill, A. J., and Gergle, D. (2008). Linguistic mimicry and trust in text-based cmc. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 277–280, New York, NY, USA. ACM.

Sherchan, W., Nepal, S., and Paris, C. (2013). A survey of trust in social networks. *ACM Computing Surveys (CSUR)*, 45(4):47:1–47:33.

Skopik, F., Truong, H. L., and Dustdar, S. (2009). Viete - enabling trust emergence in service-oriented collaborative environments. In *Proceedings of the Fifth International Conference on Web Information Systems and Technologies WEBIST*, pages 471–478.

Striukova, L. and Rayna, T. (2008). The role of social capital in virtual teams and organisations: corporate value creation. *International Journal of Networking and Virtual Organisations*, 5(1):103–119.

Zhang, Y., Chen, H.-J., Jiang, X.-H., Sheng, H., and Wu, Z.-H. (2009). RCCtrust: A combined trust model for electronic community. *Journal of Computer Science and Technology*, 24(5):883–892.