6-2014

Does latitude hurt while longitude kills? Geographical and temporal separation in a large scale software development project

Patrick WAGSTROM

Subhajit DATTA
Singapore Management University, subhajitd@smu.edu.sg

Does Latitude Hurt while Longitude Kills? Geographical and Temporal Separation in a Large Scale Software Development Project

Patrick Wagstrom
IBM Watson Group
1101 Kitchawan Rd
Yorktown Heights, NY 10538 USA
pwagstro@us.ibm.com

Subhajit Datta
Singapore University of Technology and Design
20 Dover Drive, Singapore 138682
subhajit.datta@acm.org

ABSTRACT

Distributed software development allows firms to leverage cost advantages and place work near centers of competency. This distribution comes at a cost – distributed teams face challenges from differing cultures, skill levels, and a lack of shared working hours. In this paper we examine whether and how geographic and temporal separation in a large scale distributed software development influences developer interactions. We mine the work item trackers for a large commercial software project with a globally distributed development team. We examine both the time to respond and the propensity of individuals to respond and find that when taken together, geographic distance has little effect, while temporal separation has a significant negative impact on the time to respond. However, both have little impact on the social network of individuals in the organization. These results suggest that while temporally distributed teams do communicate, it is at a slower rate, and firms may wish to locate partner teams in similar time zones for maximal performance.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management—*Software Teams*; K.4.3 [**Computing Milieux**]: Computers and Society—*Computer-supported Cooperative Work*

General Terms

Management, Human Factors

Keywords

Collaboration, Agile Development, Distributed Software Development, Social Network Analysis, Exponential Random Graph Models

ICSE'14, May 31 – June 7, 2014, Hyderabad, India
ACM 978-1-4503-2756-5/14/05
http://dx.doi.org/10.1145/2568225.2568279

1. INTRODUCTION

Software engineering may be the finest example of distributed computer supported cooperative work. Tools for software engineers are natively designed to account for teams working around the world and around the clock with a variety of different levels of communication fidelity [37]. Processes in software engineering, even those which are designed to be intensively collaborative and co-located, such as agile software development and pair programming, have excellent tools to support coordination and collaboration across time and space [27]. These tools have led to an explosion of distributed software development in both the commercial and open source worlds.

In the modern business world distributed software engineering teams are not just common, they are almost the norm. Within large multi-national companies it's not uncommon to have teams located in the United States, Brazil, China, India, and Europe all working together on the same project [21, 20]. Businesses have flocked to outsourcing and distributed software engineering as a vital component to stay relevant, cost competitive, and evolve in the face of a shifting international marketplace. While early results from the field showed promising financial advantages for outsourcing, more recent work has made the advantages of outsourcing and distributed work considerably less clear [43].

A primary challenge of working in these spatio-temporally distributed teams is synchronizing the team such that employees share some degree of working hours. While companies and individuals often use a variety of strategies to cope with temporal separation [47], these strategies only address part of the issues. Fully addressing the challenge, if possible, requires a deeper understanding of how mature teams function in a distributed environment. Specifically, it requires knowledge of how repeated interactions between individuals both inside and outside the business impact the development process.

In this paper we dive into some of the questions surrounding globally distributed software development by examining a large distributed team from IBM. We examine the artifacts collected by their work item and task management system to understand how individuals communicate with one another in the presence of geographical and temporal separation. In the next section we develop and position our hypotheses in the context of previous literature on organizational structure, software development, and distributed collaboration.

This is followed by a description of how our data was collected and two different methods of analysis, multiple linear regression and exponential random graph modeling, to test our hypotheses. Finally, we close with a discussion of the implications of these findings.

2. BACKGROUND AND RELATED WORK

In this section, we outline the background of our study in the light of related work. Our hypotheses lie at the intersection of research trends around communication and product structure, global software development, agile software development, and broader research on distributed collaboration in software development and other fields.

2.1 The Dual Relation Between Communication Structure and Product Structure

The association between the structure of communication in a team, and structure of the product built by that team has long been an intriguing question. Conway conjectured that these structures typically reflect one another [10]. This idea has subsequently inspired the "mirroring hypothesis" [9], and the notion of "socio-technical congruence" [7]. Conway's conjecture – subsequently canonized as "Conway's Law" by Brooks [4] – has a strong intuitive appeal, particularly in the context of organizations with streamlined communication channels between different teams working on different parts of a product. A colloquial illustration of the phenomenon was provided by Eric Raymond who wrote, "If you have four groups working on a compiler, you'll get a 4-pass compiler" [39].

Empirical validation of the relationship between communication structure and product structure has attracted significant research interest in recent years. Cataldo et al. introduced a definition of socio-technical congruence in the context of software development [7]. They measured congruence using the difference between coordination requirements arising out of technical dependencies vis-a-vis actual coordination, and gave empirical evidence that congruence between developers' coordination patterns and coordination needs, relates to reduced resolution time of modification requests [7]. This work has been extended by Wagstrom et al. to make a distinction between general communication and that aligned with task dependencies [48]. The authors demonstrate that the latter has minor benefits, while the former relates to reduced bug resolution time. The effects of socio-technical congruence on software build success in the IBM Rational Team Concert project was later studied by Kwan et al. who report that surprisingly, high congruence is found to be related to lower build success rates in some situations, while many zero-congruence builds are successful [26]. These results seem to offer interesting counter-indications on the effects of socio-technical congruence.

MacCormack et al. explored the duality between product and organization structures in the software industry and find evidence that loosely-coupled organizations develop products that are significantly more modular than those developed by tightly-coupled organizations [29]. On a related theme, Nagappan et al. explored the impact of organizational structure on software quality and report that organizational metrics can effectively predict failure proneness [33]. Colfer et al. examined the lineage of the mirroring hypothesis across domains and validated it across 102 empirical studies, finding support for the hypothesis in 69% of the cases [9].

Understanding how communication structure relates to product structure has assumed an urgency of late, due to the increasing popularity of global software development.

2.2 Global Software Development

The world of software engineering is evolving. Since Conway's Law was conceived, software development in the large has undergone many changes. Today, software engineering teams are frequently made up of skilled individuals gathered from locations around the world; the combination working nearly round-the-clock [5] – a paradigm commonly referred to as "global software development". Communication is nearly always mediated through internet accessible systems that automatically record, publish, and archive all information for future reference. This style of teamwork allows teams to be high performing even when distributed across dozens of locations on multiple continents. For example, GitHub, which itself is a platform for supporting distributed collaboration and software development, has a globally distributed work-force, of which only a small portion is located close to their San Francisco office [24]. Through the use of the GitHub software, real-time and archived chat, instant messaging, and email, the GitHub team is able to manage more than 150 employees and deploy their software to production servers dozens of times a day.

Understanding the nuances of global software development has been the focus of much research in the past decade and half. Herbsleb et al. demonstrated how common but unanticipated events can influence the breakdown of project communication [22]. Herbsleb and Mockus found evidence of notable communication and task completion delays for modification requests involving work across locations [20]. The importance of intra and inter team communication and coordination in a global development setting was established by Cataldo et al. [7], along with the need for accruing a common knowledge pool [11], and for recognizing expertise [15]. Bird et al. studied the development ecosystem of Windows Vista to dispel the notion that distributed development in more risky than co-located development [3]. Ehrlich et al. study the negative impact of distance in [16], while evidence for informal hierarchies facilitating smoother coordination in distributed teams have been found reported in [23]. Nguyen et al. [34] examined the question "does distance still matter?" [35] in the context of a distributed project-wide Jazz team and report that barring some exceptions, evidence for major differences between communication structures of co-located and multi-site Jazz teams were not found.

As evident from this discussion, the outcome of global software development is variously impacted by the communication mores of team members. And communication is a central theme in the agile philosophy of software development.

2.3 Interactions in Agile Development

The first credo of the *Manifesto for Agile Software Development* proclaims "Individuals and *interactions* over processes and tools", and one of the 12 guiding principles is "... developers must *work together* daily throughout the project" [2] (italics added). Taken together, these imply a significant emphasis on the role of interactions in the context of project work. Thus agile development is inherently *interaction centric*, with unfettered communication encouraged between various stakeholders, most notably, the devel-

opers. In *The Cathedral and the Bazaar*, Raymond uses the metaphor of a bazaar to highlight how the myriad of spontaneous and seemingly unconnected transactions can fulfill local objectives of individuals, as well as global objective of the marketplace [40]. Although originally invoked in the context of open source development, the bazaar metaphor aptly captures the spirit of interaction in large scale agile development. Many of the concepts behind the agile way of software development were first popularized by open source projects which have long stressed the need to have a small "nugget" of a project such that a new user or potential developer is able to quickly ascertain the value of the project [17]. These open source teams typically eschewed long term planning in favor of short iterations and used computer mediated communication systems, usually email, a centralized code repository, and a shared bug tracker, to keep all of the developers connected and in sync. The adaptation of some of the open source development traits into the agile software development movement, with its emphasis on short development iterations, adaptive scheduling, and always having a product that compiles and can be automatically tested, is one of the most notable trends in the evolution of software engineering processes [31].

The advent of new development practices like Agile, Extreme Programming (XP) etc. has put renewed emphasis on the role of team interactions in the success of a software development endeavor, as explored by Chong, Paasivaara, and others [8, 36].

It's not uncommon for large-scale projects that adopt agile methodologies to also imbibe additional traits from open source software development. One increasingly common practice is to open up a portion of the software development process to the wider community of users and customers. Most frequently this is done by providing a forum or blog to discuss development directly with developers. Sometimes projects allow developers to have direct to project work trackers, as is the case with IBM's Rational Team Concert, which not only is a tool for developers to collaborate on software development, but allows end users to comment and look directly into the project work item trackers through the Jazz project.

2.4 Jazz and Extended Development Teams

Jazz seeks to transform software development "... by making it more collaborative, productive and transparent, through integration of information and tasks across the phases of the software lifecycle" [1]. Jazz facilitates the definition and monitoring of units of work through developer discussions around them.

Development on the Jazz platform is informed by the *Eclipse way of development*, which adapts the spirit of agile methodologies for large scale distributed development [18]. Jazz supports the introduction of non-team members to core project infrastructure, such as the work item tracker, and consequently has the potential to radically change the nature of interactions within the team. Prior to the introduction of these non-team members, the team was primarily responsible for maintaining interactions and responding to additional queries amongst team members. The non-team members greatly increase the number of possible interactions required. This introduces additional cognitive load on the team members as they attempt to address all the concerns. One common observation about these environments is that repeated interactions between individuals encourages the building of common ground which then leads to more willingness to collaborate in the future [49].

In recent years several studies have analyzed data from projects on the Jazz platform. In addition to some of the papers cited earlier, we briefly mention few more interesting results. Wolf et al. develop a predictive model from a set of communication structure measurements to indicate whether an integration build is likely to fail [50]. McIntosh et al. establish that that build maintenance yields up to a 27% overhead on source code development and a 44% overhead on test development when studying a group of 10 projects, including one on the Jazz platform [32]. Ehrlich and Cataldo examine how centrality and closure affect individual performance [14]. Datta et al. study the development effort of a major product on the Jazz platform across 19 months of development activity, including 17,000+ work items and 61,000+ comments made by more than 190 developers in 35 locations [13]. They find evidence that the number of defects owned by a developer is influenced by the number of other developers he communicates to, his interpersonal influence in the network of work dependencies, the number of work items he comments on, and the number work items he owns.

As evident from the preceding discussion, studies of Jazz projects have found interactions between project stakeholders to play a central role in influencing project outcome.

2.5 Interaction Across Latitudes and Longitudes

Today, large scale software development is at an interesting juncture. Increased awareness about the effects of sociotechnical congruence are encouraging organizations to search for the most effective orientation of development teams. To best leverage the benefits of a distributed workforce and 24 hour "follow the sun" development cycles [6] organizations are deploying global software teams. Additionally, the predominance of interaction centric methodologies such as agile and the availability of collaborative development environments such as Jazz have contributed to a truly global pool of developers working together. These trends contribute to a situation where members of a software development team can be located anywhere on the globe.

The Agile Manifesto places special emphasis on face-to-face communication [2]. When team members are distributed across locations, the best alternative to face to face communication is connecting synchronously over chat, telephone, or video-conferencing, supplemented by occasional travel. Synchronous communication has the basic requirement for all participants to be awake and working at the same time. Locations which are far apart in latitude will be separated by long distances. But their time zones may not be commensurately apart if their longitudes are near one another. But places far apart in longitude will definitely have very different times zones, no matter how long it takes to travel from one place to the other.

Thus, within a distributed team latitude and longitude are expected to have very different influences. Dispersion of team members across latitudes leads to some degree of difficulty due to longer travel distances (should the need for travel arise). However team members distributed in different longitudes may make it necessary for a deeper readjustment of nearly everyone's daily work schedule, and in extreme

[1] `https://jazz.net/`

cases can also interfere with natural sleep-wake cycles [47]. On an ongoing basis, the effect of longitudinal separation appears to be more severe than being dispersed across the latitudes. This contingency is well recognized in agile circles dealing with large distributed teams. It has been pithily captured in the adage-like phrase *latitude hurts, longitude kills*[2].

But like all aphorisms, the statement "latitude hurts, longitude kills" elides some subtleties, which are worth exploring. In today's milieu, the co-workers of a typical software developer are commonly spread out over distances and time zones. Thus, investigating how developer interaction is affected by latitude and longitude is important for organizational behavior, project governance, as well as individual work profiles.

In this context, we introduce the following hypotheses to guide our investigations:

HYPOTHESIS 1. *Repeated interactions between individuals are associated with a decreased time to respond in future interactions.*

Consistent with previous research [35], it is likely that individuals who are not co-located will have a reduced chance of interacting with one another. This is partially due to the lack of visibility between individuals at different locations and also due to the way that the product teams are structured. If teams tend to be geographically centralized and there is an effort to minimize communications between team members then individuals at different locations will have less of a reason to interact with one another. This lack of communication between geographically distributed individuals may result in a decreased motivation to quickly respond to requests and leads to our next two hypotheses:

HYPOTHESIS 2. *Geographic distance is associated with a decreased chance to reciprocate interactions.*

HYPOTHESIS 3. *Geographic distance is associated with a increased time between interactions.*

Similar to the previous two hypotheses that examined geographic distance, we can also examine how temporal offset, the number of hours difference between two individuals, plays a similar role. While these two metrics are often correlated, it is becoming quite common to have teams with shared work days that may be separated by a distance of thousands of kilometers while having little temporal separation [1]. For example, Damian et. al. studied the collaboration of a team spread between unnamed locations in the United States and Brazil [12]. If we take the cities to be New York City and Sao Paulo, these cities are 7680 kilometers apart, greater than the 7504 kilometer distance from New York City to Moscow. Despite their greater distance, they are typically separated by only one hour as opposed to the eight hour time difference between New York and Moscow. It seems likely that a team distributed between New York and Moscow would have greater difficulties maintaining a collaborative atmosphere than one separated by the one hour difference between New York and Sao Paulo.

This prompts hypotheses 4 and 5, which isolate temporal offset between two individuals as a factor in the propensity to reciprocate interactions and also in the time for those reciprocations to occur.

[2]http://cf.agilealliance.org/program/files/11379.pdf

HYPOTHESIS 4. *Temporal offset is associated with a decreased chance to reciprocate interactions.*

HYPOTHESIS 5. *Temporal offset is associated with a increased time between interactions.*

Finally, it's likely that there is a relationship between geographic distance, temporal offset, the propensity to reciprocate interactions, and the time for interactions to take place. It it our belief that, given the prevalence of modern tools for computer mediated communication, the dominant factor related to the interaction between individuals will be the temporal offset and not geographic distance between the pair of individuals.

HYPOTHESIS 6. *After controlling for temporal offset, geographic distance will have little relation to the propensity to reciprocate interactions.*

HYPOTHESIS 7. *After controlling for temporal offset, geographic distance will have little relation to the time between interactions.*

3. DATA COLLECTION

The data were collected from a long lasting IBM project in the field of software development. This is considered to be a very important product for IBM and it is developed in such a way that customers have access to information from the development team, including project plans, work items, build logs, and reporting dashboards. The team uses agile programming methodologies and is generally considered to be very high performing. At the time of this writing the project is a little more than seven years old.

The team uses Rational Team Concert to support their development. This allows the team to consolidate their project planning, work item management, defect management, source code control, and many other important software engineering tasks in a single tool that provides excellent traceability and auditability for the software development process. We used publicly available OSLC APIs[3] on the project's Rational Team Concert server to download the complete discussion history on all of the work items from the project for a period of five years. The first discussion on a work item is from January 2006 and the last discussion in our dataset is from March 2011. A discussion typically begins with an individual explaining the reason for the creation of the work item, such as describing a bug or an idea for a new feature, and then proceeds with a linear non-threaded discussion system similar to mechanisms present in Bugzilla and other common work item tracking systems [44]. Each discussion item had the following pieces of identification that were used in this research:

- Work Item Identification Number
- Identity of Comment Author
- Timestamp of Comment

Other fields such as the text of the comment and work item specific fields such as the type of the work item (e.g. enhancement, defect, etc) or priority were available but were not needed for this research.

[3]http://open-services.net/

After obtaining the conversation history surrounding work items the data were augmented with two additional pieces of information: the physical location of the individual and the time zone of each individual. For employees of IBM the location of an individual was taken to be the physical location of the individual as recorded in the IBM corporate directory. For other individuals the location of the individual was obtained through the customer account registration information provided to IBM. We assumed that non-IBM individuals were truthful in sharing their location information and that IBM employees worked in the location recorded in the employee directory. This provided location information for 1636 of the 1673 total individuals registered on the site. Additionally, the IBM employee directory allowed us to further segment IBM employees into two groups, those that worked for Rational, the brand developing the product, and general IBM employees.

The time zone offset was obtained by examining the standard time zone for the user's location and was recorded as an offset from UTC. For locations that used Daylight Savings Time or Summer Time, the "Standard" time zone was used, which was usually one hour later than Summer Time. No attempt was made to deal with locations that have switched time zones during the course of the analysis. A summary of the data appears in Table 1.

Table 1: Data set summary statistics

	Rational Only	All IBM	Non-IBM	Total
Individuals	438	1306	367	1673
With Location	438	1283	353	1636
Unique Locations	12	160	204	304
Unique Timezones	3	17	16	17

Using this information we were able to create a matrix with both the geographic distance and temporal offset between each pair of individuals in the organization. Temporal offset was normalized to the range of +12 to -12 hours and then the absolute value of this distance was used. Geographic distance was calculated using the spherical law of cosines method to provide for great circle paths between two points on the globe.

For the purposes of this work, as we are most concerned with the interactions between individuals when taking into account the geographic and temporal location of the individuals, we filtered the complete data set to include only instances of interactions when location information was present for the individuals generating the comments. The resulted in an examination of 13399 different work items with 65222 comments on those work items.

The number of comments added to work items over the project life span is shown in figure 1. Within the data most of the comments were created between June 2008 and March 2011, the end of our dataset. Prior to June 2008 the team was using multiple different systems for managing work items and comments and therefore activity was sparse, with fewer than 100 comments being entered during this time period. This is most obvious from the fact that there were only 25 comments added to work items prior to March 2007, most of which were test comments as the team experimented with early versions Rational Team Concert. It was only after the first major release Rational Team Concert in June 2008 that there was a major effort to consolidate all work into the team's Rational Team Concert server and the

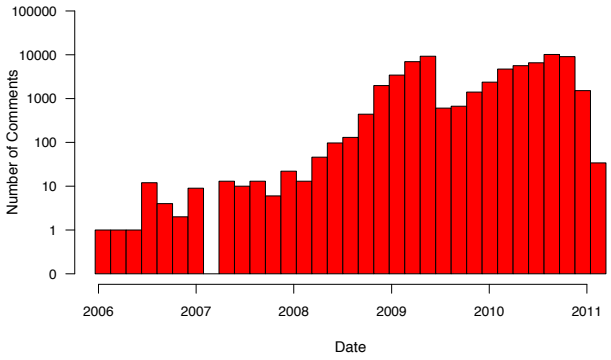

Figure 1: Number of comments added to work item tracker by date

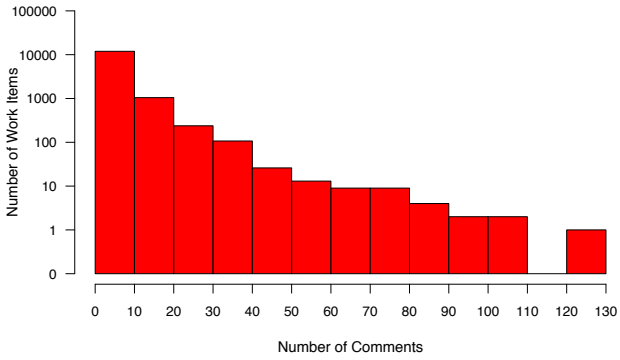

Figure 2: Histogram of number of comments per work item in dataset

number of comments available for analysis began to rapidly increase.

Most work items attracted very little discussion, getting only one or two comments. A steep drop off was seen in the number of comments added to each work item, with slightly more than 1,000 work items having between 10-20 comments, and fewer than 400 work items having more than 20 comments, as shown in figure 2.

As would be expected from a community of this size, there was no single individual that interacted with all other participants and the distribution of the frequency of interaction between pairs of individuals varied wildly. Figure 3 shows a distribution of the number of times various pairings of individuals interacted with one another in the work item tracker. In most cases there were few interactions between pairings, but there are substantial number of cases where the number of repeated interactions goes into the hundreds or even thousands, indicating that this was a collaborative community with frequent repeated interactions.

4. DATA ANALYSIS

Our analysis takes two different forms. For the hypotheses that address the temporal delay in responses – hypothesis 1, 3, 5, and 7 – we used linear regression models to to understand relationships to geographic distance, temporal offsite, and the time to respond to comments. However, such methods are not appropriate when attempting to understand the propensity for links to form between individuals in a net-

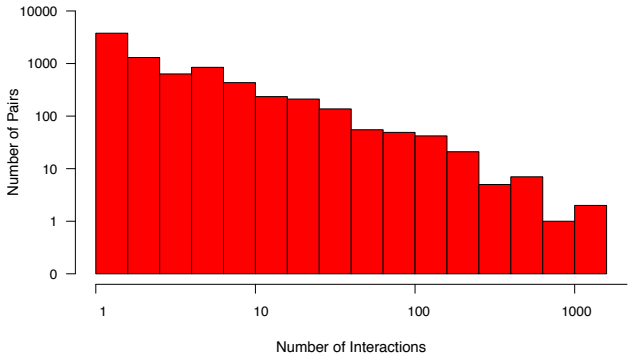

Figure 3: Histogram of number of interactions between pairs of individuals

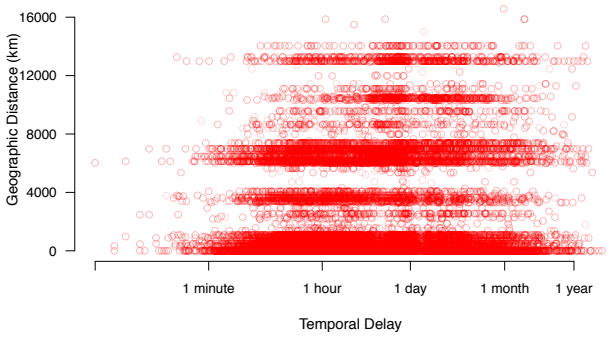

Figure 4: Relationship between temporal delay of responses on work item tracker and geographic distance

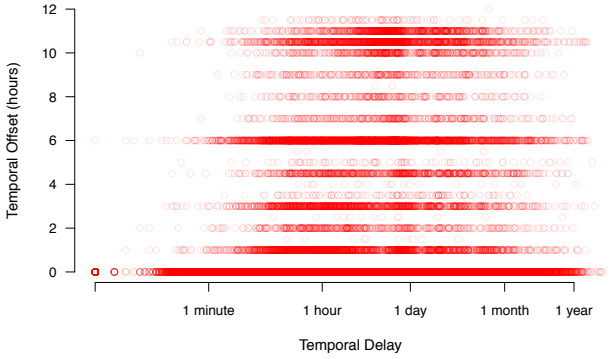

Figure 5: Relationship between temporal delay of responses on work item tracker and temporal offset

worked context. For hypothesis 2, 4, and 6 we used an advanced form of statistical network modeling, exponential random graph models (ERGM), to build a probabilistic network model and identify the factors related to communication between pairs of individuals.

4.1 Temporal Delay in Responses

To understand the temporal delay in responses it was necessary to filter the data to remove those comments which had not received a response by the end of our data set, a total of 13399 items. In addition, because we are concerned about the time it takes to respond to comment from someone else, we also removed all comments in which an individual responded to their own comment, an additional 18614 items. This resulted in a total of 33209 comments from 7633 work items that had sufficient usable data to examine the temporal delay in responses.

Before advancing into the examination of response time we conducted appropriate statistical tests to determine the similarity of response times across the different populations - Rational employees (the brand of IBM developing the software we examined), IBM employees, and non-IBM employees. As there are a variety of reasons why there may be extremely long tails in the model, we compared the median amount of time until a work item comment received a response on the basis of what population the respondent belonged to. A Kolmogorov-Smirnov test [46], a non-parametric test for analyzing similarities in the underlying distributions of sample populations when the underlying distribution is unknown, was unable to ascertain that the populations were drawn from different samples, and this allowed simplification of our work by allowing us to consider the population as a whole.

We then plotted the response temporal delay against the geographic and temporal offset between the original comment poster and the respondent as shown in figure 4 and 5. In both cases there were no clear trends that emerged from this analysis aside from the observation that most messages were responded to within a period of about two weeks and that most cases that involved very quick (under 1 minute) responses were between individuals with little temporal offset.

After gaining an overview of the data, we created a simple model to examine the relationship between the time it takes for a response and the number of previous interactions between the two parties in the discussion. We also control for the amount of time that the previous message in the discussion took, regardless of what two parties it was between. Because there is a diversity of several orders of magnitude for the values, we used the natural logarithm of the values in our regression model. The output of this first model can be seen in table 2.

As is widely practiced, we use multiple linear regression models to understand the influence of independent variables on the dependent variable (in this case, response time). In the following tables, we report the estimates of the coefficients of the independent variables along with their standard errors. The magnitude and sign ("+" or "-") of the coefficients indicate the level and directionality of influence of the corresponding independent variable on the dependent variable. The t value is the ratio of each coefficient to its standard error. Using this t value and the Student's t-distribution the p value is calculated. If the p value is less than the *level of significance*, the corresponding result is taken to be statistically significant. The level of significance is usually taken to be 0.05 - indicating a 1 in 20 chance of an effect being observed purely by chance. DF denotes the degrees of freedom. F is the Fisher F-statistic – the ratio of the variance in the data explained by the linear model divided by the variance unexplained by the model. R^2 is the coefficient of determination – the ratio of the regression sum of squares to the total sum of squares, indicating the goodness of fit of the regression model in terms of the proportion

of variability in the data set that is accounted for by the model. To compensate for over-fitting a model, adjusted R^2 adjusts the R^2 value for the number of explanatory terms in the model. Further details about regression models can be found in books such as [46].

Table 2: Multiple linear regression model showing response time based on response time for the previous message and repetition

	Estimate	Std. Error	t value	p-value
intercept	8.7798	0.0475	185.0319	<0.0001
log(previous delay)	0.1503	0.0045	33.2378	<0.0001
log(repetition)	-0.1469	0.0108	-13.5837	<0.0001
DF=27609, F=649.5, Adj $R^2 = 0.0449$				

In this simple model we also control for previous delay in discussion time between comments because some discussions naturally proceed more slowly because of organizations or technical issues, as shown by *previous delay*, which reduced the number of data points to 27619. This model shows that the more times a pair of individuals interacted, as shown by *repetition*, the less time we can expect future interactions between those individuals to take. Therefore, this provides support for hypothesis 1.

The next step was to construct a model that incorporated the geographic distance between the pair of individuals in addition to the repetition. We expect that based on the problems associated with geographically distributed teams we will find individuals who are further apart physically will be less likely to promptly respond to messages. However, this is not the case with the regression model that is shown in table 3. Rather we see a very slight, but statistically insignificant factor that indicates that an increase in distance may reduce the amount of time necessary to respond. Therefore, we do not find support for hypothesis 3.

Table 3: Multiple linear regression model showing response time based on response time for the previous message, repetition, and geographic distance

	Estimate	Std. Error	t value	p-value
intercept	8.7985	0.0559	157.4400	<0.0001
log(previous delay)	0.1501	0.0045	33.1799	<0.0001
log(repetition)	-0.1486	0.0112	-13.3151	<0.0001
log(geographic distance)	-0.0027	0.0043	-0.6342	0.5259
DF=27608, F=433.1, Adj $R^2 = 0.0449$				

Knowing that within our data there is little relation between geographic separation and the time to respond to messages on the work item tracker, a model was constructed that included the temporal offset between the individuals, as shown in table 4. Note, that because temporal offset is bounded between 0 and 12, we do not perform a logarithmic transform on it. Here we see a strong response that indicates that a larger temporal offset between individuals is associated with a slower response time to comments on the work item tracker, providing support for hypothesis 5.

Finally, we propose a model that includes both geographic distance and temporal offset, shown in table 5. When the model is completely assembled we find that temporal offset has a very large effect on the time to respond to comments on the work item tracker – on average individuals who are six time zones apart will take more than twice as long to respond ($e^{6*0.1352} = 2.25$), while individuals who are geographically distributed will have a slightly faster time to respond. This provides strong support for hypothesis 7

Table 4: Multiple linear regression model showing response time based on response time for the previous message, repetition, and temporal offset

	Estimate	Std. Error	t value	p-value
intercept	8.5419	0.0493	173.3900	<0.0001
log(previous delay)	0.1494	0.0045	33.2219	<0.0001
log(repetition)	-0.1167	0.0109	-10.7059	<0.0001
temporal offset	0.0728	0.0043	16.8929	<0.0001
DF=27608, F=532.6, Adj $R^2 = 0.0547$				

Table 5: Multiple linear regression model showing response time based on response time for the previous message, repetition, geographic distance, and temporal offset

	Estimate	Std. Error	t value	p-value
intercept	8.9729	0.0558	160.6941	<0.0001
log(previous delay)	0.1445	0.0045	32.2159	<0.0001
log(repetition)	-0.1506	0.0111	-13.6220	<0.0001
log(geographic distance)	-0.0919	0.0057	-16.1364	<0.0001
temporal offset	0.1352	0.0058	23.4099	<0.0001
DF=27607, F=468.3, Adj $R^2 = 0.0635$				

4.2 Link Formation

Using the data collected from the team's Rational Team Concert work item tracker we built a network of the collaborations present as a property graph. A property graph is a collection of nodes and edges and each node or edge may have multiple properties associated with it. In addition, nodes and edges may be of multiple types: for example, we can represent the geographic distance between two individuals as an edge between their corresponding nodes in the network. These edges have a numeric property that indicates the geographic distance between the people represented by the nodes. Likewise we can use edges to represent temporal offset, the number of previous collaborations, and nearly any other dyadic property. As an example, a sociogram showing the collaboration network obtained from examining comment replies from the work item tracker is visible in figure 6. This network shows that the communication network that emerged was quite dense and there was not a clear core-periphery structure between employees of Rational, IBM, and other users of the system.

The construction of a network allowed us to construct an exponential random graph model (ERGM) to examine the propensity of network links to form over time on the basis of overall network density, network structure, vertex properties, and edge properties [41]. Similar to a logistic regression, the output of an ERGM is a model that can be used to evaluate the probability of an edge being present given the factors of the model. Rather than providing estimates of coefficient multipliers relative to the outcome variable, as is done with a regression, in an ERGM estimates are given as factors of the inverse logit function $\left(logit^{-1}(\alpha) = e^{\alpha}/(e^{\alpha}+1)\right)$ that can then be used to calculate the odds ratio for the probability of an edge to exist in the network. For example, ERGMs can be used to model the likelihood the friendship links will be reciprocated in online social networks [28] and to model interactions in real world offices [42]. The `ergm` package [25, 19] for R [38] was used to model these networks.

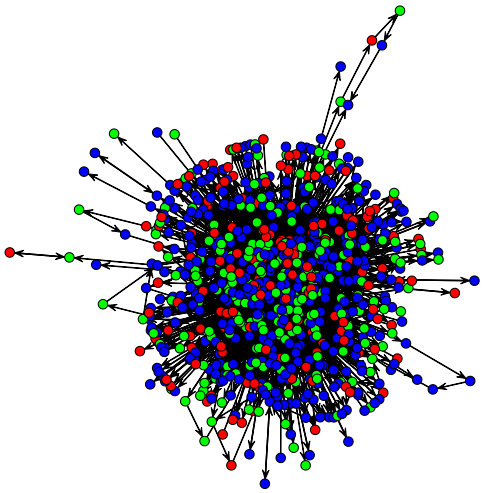

Figure 6: Sociogram of data showing edges between nodes if individuals communicated over the work item tracker. Non-IBM affiliated individuals are red, general IBM employees are blue, and employees of IBM's rational brand are green.

At the most simple level, the network state and density, represented by the *edges* parameter in an ERGM, can be used to model the probability of an edge existing based on dyadic properties such as previous collaboration, geographic distance, and temporal offset. In addition ERGMs allow edges to be formed on the basis of network related properties such as *reciprocity* – the tendency of a network given that there is a directed edge between the pair of nodes (A, B), the reciprocal directed edge (B, A) is also in the network. In a friendship network this is similar to saying that Alice says she is friends with Bob then Bob will often reciprocate and say he is friends with Alice. The complete list of all variables used in our models is shown in table 6.

The most basic version of this exponential random graph model, is shown in table 7. It shows the logit probability for an edge to be present given the overall density of the network, *edges*, and the tendency for ties to be reciprocated, *reciprocity*. The coefficient of -5.3885 on edges indicates that the probability of existence for any given is $e^{-5.3885}/\left(e^{-5.3885}+1\right) = 0.0046$ – this value is also roughly the density of the network obtained from the discussion data. This is compensated by a strong tendency for reciprocity in the network. In fact, 99.75% of all edges in the network $(e^{6.0178}/\left(e^{6.0178}+1\right) = 0.9975)$ are expected to be reciprocated based on this model. This indicates a very strong preference for an individuals to reply to each other in the network.

A slightly more advanced model, shown in table 8, introduces two new parameters that help to test hypothesis 2. The first parameter, *is_rational*, is a binary node property that indicates whether or not the individual represented by the node in the model was an employee of Rational, IBM's brand that developed the project. It's positive estimated value indicates that individuals are slightly more likely to respond to comments from employees of Rational. The second parameter is the log of the geographic distance between each pair of individuals. We find that introducing a geographic distance parameter between a pair of individuals has no statistically significant effect on the likelihood of those two individuals collaborating. Therefore, we do not find support for hypothesis 2 which states that geographic distance will result in a decreased chance of two individuals communicating.

Table 6: Summary of all ERGM variables used in this paper

Variable	Description
edges	Controls for the overall density of the network.
reciprocity	The tendency of nodes to reciprocate connections. Represented by the *mutual* parameter within the R ERGM package.
is_rational	A binary node factor that is *true* if the node represents an employee of the Rational brand within IBM and *false* if not. Represented by a *nodefactor* construction within the R ERGM package.
temporal distance	The absolute difference in hours between any two people in the network. This is represented by an *edgecov* construction with the R ERGM package.
log(geographical distance)	The log of the physical distance in kilometers between any two people in the network. This is represented by an *edgecov* construction with the R ERGM package. For cases where two individuals have the same location and the log of their geographical distance is undefined it is set to 0.

Table 7: Basic ERGM model controlling for density and reciprocity

	Estimate	Std. Error	p-value
edges	-5.3885	0.0226	<0.0001
reciprocity	6.0178	0.0566	<0.0001

Table 8: ERGM controlling for geographic distance

	Estimate	Std. Error	p-value
edges	-5.3392	0.4956	<0.0001
reciprocity	6.1226	0.0947	<0.0001
is_rational	0.4675	0.1183	0.0001
log(geographic distance)	-0.0435	0.0508	0.3920

To test hypothesis 4 we replace the geographic distance with the temporal offset between each pair of individuals, shown in table 9. We find no statistically significant relation between the chance of two individuals collaborating with one another and their temporal offset from one another. Therefore, we do not find support for hypothesis 4.

The final ERGM model, shown in table 10, takes into account both geographic distance and temporal offset between individuals. In this case the combination of the two effects creates a very complicated situation, increasing the significance and estimated effect of both factors. As before, *geographic distance* is not statistically significant. The effect for the *temporal offset* is statistically significant and in the positive direction, although the effect is fairly weak. In this case, and in contrast to the previous regression models, the positive direction results in an increase to the odds ratio of

Table 9: ERGM controlling for temporal offset

	Estimate	Std. Error	p-value
edges	-5.6135	0.0757	<0.0001
reciprocity	5.8606	0.1977	<0.0001
is_rational	0.4506	0.0361	<0.0001
temporal distance	0.0011	0.0092	0.9032

an edge being present in the network. Thus, the model indicates that if two individuals are temporally separated they are slightly *more* likely to be connected to one another. This is in complete contrast to our expectations of hypothesis 6 and is a very surprising result.

Table 10: ERGM model controlling for geographic distance and temporal offset

	Estimate	Std. Error	p-value
edges	-5.5220	0.3640	<0.0001
reciprocity	6.1465	0.0791	<0.0001
is_rational	0.5718	0.1032	<0.0001
log(geographic distance)	-0.0744	0.0413	0.0715
temporal offset	0.0372	0.0084	<0.0001

5. DISCUSSION

Attempting to predict social behaviors, such as two people communicating, is always a difficult process. This is magnified even more so when examining just a small sliver of the communication between those individuals, such as that provided in a project work item trackers within a collaborative development environment, such as Rational Team Concert. However, these tools can still provide a rich lens to understand and study the behavior of complex projects.

Based on existing literature we built and explored seven different hypotheses in this work, four were based on the time to respond to comments on the project work item tracker (hypotheses 1, 3, 5, and 7) and three were based on the propensity of two individuals to communicate (hypotheses 2, 4, and 6). Surprisingly, while there are many ways that our findings complement previous literature, they also contrast and pose a challenge to some aspects of elements of existing literature.

At the most basic level, we found that repeated interactions between a pair of individuals resulted in reduced future response times, supporting hypothesis 1. This agrees with prior research on team building [30] and suggests that organizations should support periodic team-building interactions for all team members (e.g. face-to-face meetings for the entire team, introductory one-on-one phone calls for new project members, shared video conferences over meals) to promote a higher sense of connection between team members, leading to lower response times in future interactions.

When it came to strictly looking at the effect of geographic distance on the propensity to reciprocate interactions (hypothesis 2) and the time it takes to respond (hypothesis 3) we found that there was no statistically significant relationship. While this is in contrast with the previous findings of Olson and Olson [35], it is consistent with the findings of of Nguyen et. al. [34]. Part of this is likely due to the fact that tools for supporting collaboration across geographic distances have significantly improved since Olson and Olson's original work. In particular, Rational Team Concert was designed with distributed teams in mind.

Next we examined the role of temporal offset in reciprocation of interactions (hypothesis 4) and time to to respond to comments (hypothesis 5). Here we found no indication that temporal separation resulted in a decreased chance of reciprocation, but we did find that temporal separation resulted in an increased average time for interactions. This seems to indicate that the primary issue is not that team members in different times zones do not want to collaborate, it's just that the difference in working hours make it so they can not respond immediately. This finding seems to support the findings of Tang et. al. who found that a common coping strategy for globally distributed was to partially time shift an employees schedule, for example by allowing an employee with a young child to work from 10am – 3pm and again from 8pm – 11pm after the child had gone to sleep [47]. This tactic of shifting a schedule also results in a greater probability of times overlapping for radically distributed teams.

Our final analyses, which take into account the effects of both geographic and temporal distance on the reciprocation of communication (hypothesis 6) and the time to respond to comments on the issue tracker (hypothesis 7) had the most nuanced results. First we found counter-intuitive results regarding reciprocity. When accounting for both factors geographic and temporal distance it was found that geographic distance was not a statistically significant factor in reciprocity while temporal distance actually made people more likely to reciprocate interactions. It is not clear how this finding lines up with existing theory and it's presence suggests that future research may be needed to see if this is an abnormality in our data or a more common phenomena.

In contrast, hypothesis 7, was consistent with both the predictions of existing theory and the oft repeated statement "latitude hurts, longitude kills" that indicates that it is not merely the presence of geographic distance that results in slower interactions, but the presence of temporal distance that causes the greatest challenge to response time in distributed teams. This has significant implications for distributed teams and in particular suggests that teams may be significantly better off by searching for development partners in a similar time zone.

From a practical perspective, this research was not designed to say if globally distributed software engineering teams were a positive or negative asset to the software industry. Rather it was designed to shed light on the process of communication in a globally distributed team. It shows that while there may be cost savings related to globally distributed development, there is still much to understand about optimal placement. Although the strategy of using development teams in China and India is quite popular, the time distance from teams in the United States may suggest that partner teams in South America would create an environment with a faster response time.

6. THREATS TO VALIDITY

There are some clear drawbacks and potential threats to validity for this work. It is extremely hard to predict social behaviors in any context, much less in a globally distributed environment such as this community. The adjusted R^2 of the regression models were small, indicating that these effects highlight only a tiny amount of the wide variance in the time to respond to comments on work items. Because of the complex nature of the items discussed on the tracker – external feature requests, bug reports, and planning for new

features are all common – it is difficult to tell to what degree these results are in line with prior research.

The use of exponential random graph models holds significant promise for the future of modeling these behaviors. In particular, this work took the network as a static entity, rather than a dynamically evolving entity that grows and morphs as a project progresses. Basic exponential random graph models cannot account for this level of dynamism in the social graph, however there are advanced models being developed that may be able to account for these shifting networks [45]. Archival communication data obtained from computer mediated communication systems, such as work item trackers, is an ideal data source to use to generate these evolutionary social networks and may prove for a very fruitful research path.

Finally, we've only studied a single, albeit very large and open, project for this research. The team behind the project is one of the best teams inside of IBM and is very experienced with distributed collaboration – both in terms of using tools to support distributed collaboration and also in terms of managing a distributed agile development process. However, being a single project, this may amplify issues with the data used for this work. For example, we made no effort to account for transitions to daylight savings time/summer time. Additionally, although we were fortunate with this model, a team that works in extreme northern parts of the world would have a very different model because of the fact that the shortest geographic distance between two individuals would cross the polar regions.

In addition to the Rational Team Concert server employed by the team, the team also makes heavy use of additional web based collaborative development tools, instant messengers, and of course, email. As the team matured the use of non-automated email decreased significantly, while the use of the tracker skyrocketed, which may have an impact on the final model. Although we did find that temporal offset resulted in slower responses, we counterintuitively found that geographic distance resulted in faster responses. It is possible that this team was an extreme outlier in these regards and that examination of other teams may result in dramatically different findings.

7. CONCLUSION

This work attempted to put some hard data behind the oft-repeated mantra of agile and distributed software development, "latitude hurts, longitude kills", which is taken to mean that geographical separation between members of a development team may cause some difficulty in interaction, whereas temporal separation is likely to make things significantly more difficult. We evaluated the relative pain in two different ways, the amount of time that it takes to get a response from another individual and the probability of ever having communicated with other individuals on the team. We found that indeed, when it comes to the amount of time required to respond to a comment on a work item that longitude kills. In contrast we found very little evidence of latitude hurting. On a social basis, when a statistical network model was created the opposite effect was found. Individuals who were temporally distributed were more likely via the project work item tracker, perhaps using it as a substitute for synchronous communication.

We hope that this work has put some academic rigor behind "Latitude Hurts, Longitude Kills". In reality, like most aphorisms, the reality is significantly more complicated. Organizations using global software development strategies may want to consider finding ways to reduce response time of geographically distributed teams by minimizing temporal separation while strengthening the support for computer mediated communication tools that foster communication across time and space.

8. REFERENCES

[1] Arora, A., and Gambardella, A. *From underdogs to tigers: the rise and growth of the software industry in Brazil, China, India, Ireland, and Israel*. Oxford University Press, Oxford, 2006.

[2] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. Manifesto for agile software development. Last visited: May 15, 2013. http://agilemanifesto.org/principles.html, 2001.

[3] Bird, C., Nagappan, N., Devanbu, P., Gall, H., and Murphy, B. Does distributed development affect software quality?: an empirical case study of windows vista. *Communications of the ACM 52*, 8 (Aug. 2009), 85–93.

[4] Brooks, F. P. *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition*, 2 ed. Addison-Wesley Professional, Boston, MA, USA, Aug. 1995.

[5] Carmel, E. *Global Software Teams: Collaborating Across Borders and Time Zones*. Prentice Hall, Upper Saddle River, NJ, 1999.

[6] Carmel, E., Espinosa, J., and Dubinsky, Y. "Follow the sun" workflow in global software development. *Journal of Management Information Systems 27*, 1 (July 2010), 17–38.

[7] Cataldo, M., Herbsleb, J. D., and Carley, K. M. Socio-technical congruence: a framework for assessing the impact of technical and work dependencies on software development productivity. In *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ACM (Kaiserslautern, Germany, 2008), 2–11.

[8] Chong, J. Social behaviors on XP and non-XP teams: a comparative study. In *Proceedings of the 2005 Conference on Agile Software Development* (Denver, CO, USA, July 2005), 39–48.

[9] Colfer, L., and Baldwin, C. Y. The mirroring hypothesis: Theory, evidence and exceptions. Working Paper ID 1539592, Harvard University Working Paper, June 2010.

[10] Conway, M. How do communities invent? *Datamation 14*, 5 (Apr. 1968), 28–31.

[11] Cramton, C. D. The mutual knowledge problem and its consequences for dispersed collaboration. *Organization Science 12*, 3 (June 2001), 346–371.

[12] Damian, D., Marczak, S., and Kwan, I. Collaboration patterns and the impact of distance on awareness in requirements-centred social networks. In *15th IEEE International Conference on Requirements Engineering Conference*, IEEE (Dehli, India, 2007), 59–68.

[13] Datta, S., Sindhgatta, R., and Sengupta, B. Talk versus work: characteristics of developer collaboration on the jazz platform. In *Proceedings of the ACM International Conference on Object Oriented Programming Systems Languages and Applications*, OOPSLA '12, ACM (Tucson, AZ, USA, 2012), 655–668.

[14] Ehrlich, K., and Cataldo, M. All-for-one and one-for-all?: a multi-level analysis of communication patterns and individual performance in geographically distributed software development. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, CSCW '12, ACM (New York, NY, USA, 2012), 945–954.

[15] Ehrlich, K., and Chang, K. Leveraging expertise in global software teams: Going outside boundaries. In *2006 International Conference on Global Software Engineering* (Florianopolis, Brazil, Oct. 2006), 149–158.

[16] Ehrlich, K., Valetto, G., and Helander, M. Seeing inside: Using social network analysis to understand patterns of collaboration and coordination in global software teams. In *Second IEEE International Conference on Global Software Engineering, 2007. ICGSE 2007* (2007), 297–298.

[17] Fogel, K. *Producing Open Source Software*. O'Reilly & Associates, Sebastapol, CA, 2005.

[18] Frost, R. Jazz and the eclipse way of collaboration. *IEEE Software 24*, 6 (2007), 114–117.

[19] Handcock, M. S., Hunter, D. R., Butts, C. T., Goodreau, S. M., and Morris, M. Statnet: Software tools for the statistical modeling of network data. version 3.0., Mar. 2012. Last visited: May 30, 2013. http://statnet.org/.

[20] Herbsleb, J., and Mockus, A. An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on Software Engineering 29*, 6 (June 2003), 1–14.

[21] Herbsleb, J., and Moitra, D. Global software development. *IEEE Software 18*, 2 (2001), 16–20.

[22] Herbsleb, J. D., and Grinter, R. E. Architectures, coordination, and distance: Conway's law and beyond. *IEEE Software 16*, 5 (1999), 63–70.

[23] Hinds, P., and Mcgrath, C. Structures that work: Social structure, work structure and coordination ease in geographically distributed teams. In *Proceedings of the 2006 20th anniversary conference on Computer Supported Cooperative Work*, ACM Press (Banff, Alberta, Canada, Nov. 2006), 343–352.

[24] Holman, Z. How GitHub works, Aug. 2011. Last visited: May 15, 2013. http://zachholman.com/posts/how-github-works/.

[25] Hunter, D. R., Handcock, M. S., Butts, C. T., Goodreau, S. M., and Morris, M. ergm: A package to fit, simulate and diagnose exponential-family models for networks. *Journal of Statistical Software 24*, 3 (May 2008), 1–29.

[26] Kwan, I., Schroter, A., and Damian, D. Does socio-technical congruence have an effect on software build success? a study of coordination in a software project. *IEEE Transactions on Software Engineering 37*, 3 (2011), 307–324.

[27] Lanubile, F., Ebert, C., Prikladnicki, R., and Vizcaino, A. Collaboration tools for global software engineering. *IEEE Software 27*, 2 (Apr. 2010), 52–55.

[28] Lu, Z., Savas, B., Tang, W., and Dhillon, I. Supervised link prediction using multiple sources. In *10th International Conference on Data Mining*, IEEE (Sydney, Australia, 2010), 923–928.

[29] MacCormack, A., Baldwin, C., and Rusnak, J. Exploring the duality between product and organizational architectures: A test of the "mirroring" hypothesis. *Research Policy 41*, 8 (Oct. 2012), 1309–1324.

[30] Marlow, J., and Dabbish, L. Designing interventions to reduce psychological distance in globally distributed teams. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion*, CSCW '12, ACM (Seattle, WA, USA, 2012), 163–166.

[31] Martin, R. C. *Agile Software Development: Principles, Patterns, and Practices*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2003.

[32] McIntosh, S., Adams, B., Nguyen, T. H., Kamei, Y., and Hassan, A. E. An empirical study of build maintenance effort. In *Proceedings of the 33rd International Conference on Software Engineering*, ICSE '11, ACM (Honolulu, HI, USA, 2011), 141–150.

[33] Nagappan, N., Murphy, B., and Basili, V. The influence of organizational structure on software quality: an empirical case study. In *Proceedings of the 30th International Conference on Software Engineering*, ICSE '08, ACM (Leipzig, Germany, 2008), 521–530.

[34] Nguyen, T., Wolf, T., and Damian, D. Global software development and delay: Does distance still matter? In *Proceedings of the 2008 International Conference on Global Software Engineering*, IEEE (Bangalore, India, Aug. 2008), 45–54.

[35] Olson, G. M., and Olson, J. S. Distance matters. *Human Computer Interaction 15*, 2&3 (2000), 139–178.

[36] Paasivaara, M., Durasiewicz, S., and Lassenius, C. Using scrum in a globally distributed project: a case study. *Software Process: Improvement and Practice 13*, 6 (2008), 527–544.

[37] Pickering, J. M., and Grinter, R. E. Software engineering and CSCW: a common research ground. In *Software Engineering and Human-Computer Interaction*, R. N. Taylor and J. Coutaz, Eds., no. 896 in Lecture Notes in Computer Science. Springer Berlin Heidelberg, Jan. 1995, 241–250.

[38] R Core Team. *R Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012.

[39] Raymond, E. S. *The New Hacker's Dictionary*. MIT Press, Cambridge, MA, 1996.

[40] Raymond, E. S. *The Cathedral and the Bazaar*. O'Reilly & Associates, Sebastapol, CA, Oct. 1999.

[41] Robins, G., Pattison, P., Kalish, Y., and Lusher, D. An introduction to exponential random graph (p*) models for social networks. *Social Networks 29*, 2 (May 2007), 173–191.

[42] Sailer, K., and McCulloh, I. Social networks and spatial configuration - how office layouts drive social

interaction. *Social Networks 34*, 1 (Jan. 2012), 47–58.

[43] Sakthivel, S. Managing risk in offshore systems development. *Communications of the ACM 50*, 4 (Apr. 2007), 69–75.

[44] Serrano, N., and Ciordia, I. Bugzilla, ITracker, and other bug trackers. *IEEE Software 22*, 2 (2005), 11–13.

[45] Snijders, T. A., van de Bunt, G. G., and Steglich, C. E. Introduction to stochastic actor-based models for network dynamics. *Social Networks 32*, 1 (Jan. 2010), 44–60.

[46] Tabachnick, B. G., and Fidell, L. S. *Using multivariate statistics*. Pearson Education, Boston, June 2012.

[47] Tang, J. C., Zhao, C., Cao, X., and Inkpen, K. Your time zone or mine?: A study of globally time zone-shifted collaboration. In *Proceedings of the ACM 2011 conference on Computer Supported Cooperative Work*, CSCW '11, ACM (Hangzhou, China, Mar. 2011), 235–244.

[48] Wagstrom, P., Herbsleb, J., and Carley, K. Communication, team performance, and the individual: Bridging technical dependencies. In *Proceedings of the 2010 Academy of Management Annual Meeting* (Montreal, Canada, Aug. 2010).

[49] Wang, Y., and Redmiles, D. Understanding cheap talk and the emergence of trust in global software engineering: An evolutionary game theory perspective. In *Proceedings of the 6th International Workshop on Cooperative and Human Aspects of Software Engineering* (San Francisco, CA, May 2013).

[50] Wolf, T., Schroeter, A., Damian, D., and Nguyen, T. Predicting build failures using social network analysis on developer communication. In *Proceedings of the 2009 IEEE 31st International Conference on Software Engineering*, IEEE Computer Society (Vancouver, BC, May 2009), 1–11.