Trinity University
Digital Commons @ Trinity

School of Business Faculty Research | School of Business

5-2014

Structural Changes Associated with Temporal Dispersion in Software Development Teams: Evidence from Open Source Software Project Teams

Jorge A. Colazo
Trinity University, jcolazo@trinity.edu

Follow this and additional works at: https://digitalcommons.trinity.edu/busadmin_faculty

 Part of the Finance and Financial Management Commons

Repository Citation

Colazo, J. (2014). Structural changes associated with temporal dispersion in software development teams: Evidence from open source software project teams. *International Journal of Innovation Management, 18*(5), 1-21. doi: 10.1142/S1363919614500303

This Article is brought to you for free and open access by the School of Business at Digital Commons @ Trinity. It has been accepted for inclusion in School of Business Faculty Research by an authorized administrator of Digital Commons @ Trinity. For more information, please contact jcostanz@trinity.edu.

International Journal of Innovation Management
Vol. 18, No. 5 (October 2014) 1450030 (21 pages)
© Imperial College Press
DOI: 10.1142/S1363919614500303

Imperial College Press
www.icpress.co.uk

STRUCTURAL CHANGES ASSOCIATED WITH TEMPORAL DISPERSION IN SOFTWARE DEVELOPMENT TEAMS: EVIDENCE FROM OPEN SOURCE SOFTWARE PROJECT TEAMS

JORGE COLAZO

Department of Finance and Decision Sciences
Trinity University, 1 Trinity Pl. TX 78212
jcolazo@trinity.edu

Published 22 May 2014

Collaboration structure and temporal dispersion (TD) in teams have been studied independently so far. This study uses Media Synchronicity Theory (MST) to derive hypotheses positing that the structure of collaboration networks in distributed teams changes when those teams are more temporally dispersed. The empirical test of hypotheses using ordinary least squares with archival data from 230 open source software (OSS) projects shows that the collaboration structure networks of those OSS teams that are more temporally dispersed are sparser and more centralised, and these associations are stronger in those teams exhibiting higher relative performance. Theoretical and practical consequences are discussed.

Keywords: Temporal dispersion; distributed teams; collaboration structure; open source software.

Introduction

Open source software (OSS) is a software whose source code is freely available to the public, allowing any skilled individual to modify and redistribute the product. While flagship examples of OSS include well-known products such as Linux, Apache, and MySQL, there are successful and widely used examples of OSS in most types of applications, from messaging suites, to e-commerce, to media players. The importance of OSS for the private sector, government, and education is nowadays beyond question, and Sourceforge.net, the largest repository of OSS, counts with more than half a million projects in different stages.

OSS is generally developed by distributed teams of developers. The structure of OSS teams has been described as a loosely connected network of developers (Crowston *et al.*, 2007), who make extensive use of electronic communications and less frequently of other communication media such as phone calls or face-to-face meetings (Yamauchi *et al.*, 2000). Many developers work in several projects at any given time (Colazo, 2010), and most of them are volunteers, although some projects with corporate sponsorship include paid professional developers (Fitzgerald, 2006).

The internal structure of OSS teams has repeatedly been described under the social network paradigm (Wasserman and Faust, 1994), where developers are the nodes of a network, and links are present when two developers share work in a specific part of the project (e.g., Colazo, 2010; Singh *et al.*, 2011). The mathematics of graph theory allows characterising those networks using different structural characteristics such as network density or network centralisation (Wasserman and Faust, 1994). Features of those networks define the collaboration dynamics of the team and have been linked to various measures of performance such as productivity, quality and developer permanence (Colazo, 2010).

Not only are OSS teams connected in networked structures, but also they are a prime example of dispersed virtual teams. OSS developers are dispersed both geographically and temporally, (i.e., they work from different places and times zones, at different times). Temporal dispersion (TD) is defined as the variation in working hours throughout members of a team (Colazo and Fang, 2010). The concept of TD complements the traditional idea of geographical dispersion when studying team features and performance, and is receiving increased attention in academia and practice. In fact, many software development projects pursue a "Follow the Sun" approach where at the end of their day developers hand over work to collaborators in other time zones (Carmel and Espinosa, 2010; Colazo and Fang, 2010).

Although to date OSS team structure and TD have been researched independently, it is plausible to consider that varying degrees of TD have correlates in the characteristics of the teams' collaboration structure. Neither theoretically nor empirically have researchers studied to date, the possible associations between the degree of TD in OSS teams and the structure of their collaboration networks. Such gap in the literature is remedied in this study. Our first research question is: *How does the collaboration structure of OSS development teams correlate with the team's TD?*

If there is a logical correspondence between TD and network structure, it is substantively interesting to examine whether better performing teams have a different kind of association between TD and structure than worse performers. The underlying idea is that after identifying how structure should change with TD,

those teams with a better fit between TD and structure would outperform others. Thus the second research question is: *Does team performance moderate the direction or strength of association between TD and team structure?*

We tackle these questions through Media Synchronicity Theory (MST) (Dennis and Valacich, 1999, 2008), which is used to theorise how TD should affect media synchronicity in functioning teams through the effect of TD on the five media capabilities that, as per this theory, are key to the performance of communication processes. Social Network Analysis (SNA) is tapped into to describe the collaboration structure.

Hypotheses about the association between TD and structural parameters are tested using regression models with empirical data from archival sources belonging to 230 OSS projects, including general data on project characteristics and metrics related to their source code.

In the remainder of this paper, Sec. 2 lays out the literature review and theoretical background. Section 3 includes the hypotheses development, while Sec. 4 explains the analytical methods used to test the hypotheses and Sec. 5 the results obtained. Section 6 includes the conclusions, limitations, and implications of this work for future research.

Literature Review and Theoretical Background

Temporal dispersion

Team dispersion has had different conceptualisations depending on the aspect of the team that is disseminated; for instance dispersion in contributed effort (Olivera *et al.*, 2008), or in team member turnover (Colazo and Fang, 2009). However, the most often studied trait is the dispersion in distance between team members, or geographic dispersion (Majchrzak *et al.*, 2000; Cramton, 2001).

Geographic dispersion often concurs with and sometimes masks the presence of TD. The variation in working hours throughout members of a team is often associated with geographic dispersion, since geographically dispersed team members often work from different time zones. With the use of technologies that allow remote collaboration, the importance of face-to-face contact has diminished, even for teams working at close physical range. In fact, in many instances even collocated team members preferred the use of communication technologies instead of resorting to a totally feasible face-to-face contact (Griffith *et al.*, 2003).

Recent research suggests that the concept of geographic dispersion is becoming outdated and should perhaps be dropped from future definitions of team dispersion (Kirkman and Mathieu, 2005). Kiesler *et al.*, (2002) when discussing highly dispersed teams operating in technologically intensive contexts, suggested that

geographic proximity might be becoming largely irrelevant and will be superseded by the concept of "virtual proximity", which is enabled and altered by the use of different communication technologies.

The concept of TD applies particularly well to OSS. Most OSS developers are volunteers who not only are geographically dispersed but also have very flexible work hours (Dempsey, 2002). They do not normally adhere to regular "office" hours and even their temporal work patterns may change from day to day or over longer periods of time. Thus, even those who sit in the same time zone may still work at different times, showing a substantive degree of TD. Additionally, OSS team membership is highly fluid, and thus produces a constant renewal and variation of the team's work time signature as project membership changes.

Research on TD is relatively nascent, and presents a predominantly negative view on the effects of TD and spatial dispersion on virtual team performance (O'Leary and Cummins, 2007), with some exceptions (Colazo and Fang, 2010). Spatial dispersion has been related to a reduction in spontaneous communication frequency mainly because it decreases the likelihood of face-to-face communication (Allen, 1977; Te'eni, 2001). TD, however, not only minimises spontaneous communications but also reduces real-time problem solving because it fundamentally decreases the potential for any form of synchronous interaction (Burke *et al.*, 1999).

In a typical OSS project, software is developed by a spatially-dispersed group that manages interdependencies by coordinating its efforts through computer-mediated channels with limited face-to-face interaction (Duchenaut, 2005). The decreasing level of synchronous interaction as teams become more distant generates coordination difficulties and becomes more salient as the TD of a team grows (Espinosa *et al.*, 2006). Mixed evidence exists on the impact of TD on team performance and the mechanisms through which TD may exert influence.

On the one hand, several studies suggest that TD detracts from team member coordination and degrades communication quality (Warkentin *et al.*, 1997). In asynchronous communication environments, coordinating temporal patterns of group behaviour is a significant challenge because the transmission of verbal cues is hindered, feedback is delayed, and interruptions and long pauses in communication can occur (McGrath, 1991). In addition, long-time lapses between communication events, as is often the case in temporally dispersed situations, can result in disjointed and discontinuous discussions (Ocker *et al.*, 1995). As such, research has found that temporally dispersed virtual teams (e.g., global software development teams) face specific problems, such as increased coordination costs (Espinosa and Carmel, 2003), additional barriers to conflict management (Montoya-Weiss *et al.*, 2001), difficulty in assimilating atypical work hours and in meeting deadlines

(Labianca *et al.*, 2005), as well as a substantial decrease in the attainability and effectiveness of leadership control over a team (Jarvenpaa *et al.*, 1998).

On the other hand, some research postulates that TD can have positive effects on the effectiveness of asynchronous communication. First, asynchronous communication, by definition, eliminates time and space constraints on the act of communicating. Second, asynchronous communication allows members to take time to consider more carefully both the received information and the responses that should follow. Third, it can also allow members to consult other resources, internal or external to the team, for improved problem solving (Rasters *et al.*, 2002). Fourth, an array of IT tools has recently been made available to facilitate and coordinate tasks that would otherwise be difficult to manage in asynchronous communication. For instance, in the context of software development, source code control systems, also called "Versioning Systems," such as Concurrent Versioning System (CVS) or the newer Subversion, are tools specifically designed to allow developers asynchronously contribute to the code base (Mockus *et al.*, 2002) and to facilitate coordination (Grinter, 2000). OSS project teams also coordinate their work using other lean communication media, such as mailing lists (Yamauchi *et al.*, 2000).

Media synchronicity theory

MST (Dennis and Valacich, 1999, 2008) argues that communication supporting task fulfillment can be understood in terms of two processes: conveyance of information and convergence of meaning. In order to perform those processes, actors engage in information transmission and information processing, where the locus of activity is among actors for the former, and within actors for the latter.

Both conveyance and convergence rest on a series of media capabilities: *Transmission velocity* is the speed at which messages can be transmitted. *Parallelism* is the number of simultaneous transmissions from multiple senders that can take place simultaneously. The *number of symbol sets* is the number of ways the medium allows the message to be decoded. These sets include written symbols, natural languages, programming languages, etc. *Rehearsability* is the extent to which the message can be fine tuned before transmitting it to the recipient. *Reprocessability* is the extent to which the media permits the message to be processed again, while and after the initial transmission.

MST posits that convergence processes are more effective with media showing a high degree of synchronicity, while conveyance accepts but does not require synchronicity, though asynchronic communications are slower and in general less efficient. Media synchronicity has been linked to virtual team performance

(Hassell and Limayem, 2011; Cao *et al.*, 2012) and MST has been used in the context of asynchronous electronic communication tools such as instant messaging, and it has been implied, though not proven, that team network structure relates to media synchronicity drivers (Ou *et al.*, 2010). Of the two processes; convergence and conveyance, we are here interested only in the latter, since it is the only one involved in the interaction between actors.

Social network analysis

Researchers adhering to SNA (Freeman, 1979) see features of social actors as the result of the actors' embededness in multiple networks. Analytically, graph theory is used to characterise these networks, where actors are nodes and links are present when a certain relationship between dyads of actors is observed (Wasserman and Faust, 1994).

Graph theory provides a rich collection of parameters that can be used to characterise and differentiate particular networks. There are actor-level parameters such as actor degree or actor centrality, and also network level parameters such as network density or network centralisation (Wasserman and Faust, 1994).

Group network structure has been previously studied using SNA in relation to group features, for instance distribution of knowledge (Rulke and Galaskiewicz, 2000), and in relation to group success, e.g., project success (Singh *et al.*, 2011), new product development performance (Colazo, 2007) and other success metrics (Colazo, 2010; Singh *et al.*, 2011).

Two of the more commonly used network characteristics are density and centralisation. Network density is the ratio between the number of links present and the maximum possible number of such links. Density represents the cohesion of the network; or how "close-knit" the team is. In a collaboration network, zero density corresponds to an empty graph, where nobody collaborates and all actors are isolates, while density of one corresponds to a team where everyone collaborates with everyone else. Density can then be construed as a measure of the intensity of collaboration that goes on within the team.

Actors in a network have different relative importance, i.e., centrality. Actor centrality indexes are calculated considering a certain definition of importance, for instance the number of ties they maintain, or "degree" (Snijders, 1981), but there are other less common definition of actor indices (Wasserman and Faust, 1994). While actor centrality represents the importance of an actor, network centralisation represents the dispersion on the actors' centrality. In networks with low centralisation, all actors have the same relative importance, while in a network with high centralisation one actor is relatively more important than the rest.

In accordance with most extant research in SNA (Wasserman and Faust, 1994), density and centralisation are the two focal structural traits to be investigated in their relation to TD.

Research Model and Hypotheses

Theoretical background

In this section, we theorise how different degrees of TD are expected to associate with different network structures. The two structural traits we are going to consider are network density and network centralisation. We consider, through the lens of MST, how the five drivers of media synchronicity, focusing on the conveyance process, are affected when teams become temporally dispersed.

From the MST perspective, a key to understand the potential effect of TD is the necessary shift from fully synchronous media, possible with TD $= 0$, to the use of media that accepts some degree of asynchronicity when teams start becoming temporally dispersed. With no TD, all team members' working hours are the same, and fully synchronous communication is possible, whereas when TD increases, team members' working hours are under increasing variance, making necessary the use of communication tools with asynchronous capability such as email, internet relay chat and instant messaging.

Some media can be used synchronously but also can accommodate varying degrees of asynchronicity, for instance a text message can be responded to either immediately or sometime after it was received. It can be expected that as TD increases, media will be shifted to engage in asynchronicity or alternatively, the asynchronous capabilities of the same media are going to be tapped into.

With the increase of TD, we face an increasing importance of message transmission trough indirect paths. For instance, developer "A", working in the U.S. East Coast will post a modification to the source code that is analysed and modified by developer "B" on the U.S. West-Coast, and later modified again by developer "C" working in Singapore. The first developer, when starting a new day of work, would see feedback for his work from "C" that has been influenced by "B", all in a serial fashion.

While MST was derived considering dyads of actors (senders and recipients) networked structures comfortably consider the indirect transmission of information. Some SNA concepts such as information centralisation were explicitly designed to account for both direct and indirect paths, shortest (geodesic) or otherwise.

Hypotheses

To start analysing how TD associates with the drivers of media synchronicity, we begin by looking at transmission velocity. Transmission velocity refers to the

speed at which a message can be delivered to a recipient. Transmission velocity reflects on the time between the sender releases the message and the message is effectively received. As TD increases, the overlap between working hours of sender and recipient diminishes, which renders ineffective synchronous media such as direct telephone calls and videoconferencing, needing to replace those by asynchronous media. Also, even with the use of the same kind of asynchronous media, since in general there will be a difference in working times, the time for effective delivery of the message is increased, and then transmission velocity decreased. The effect is compounded when indirect transmission is considered, since retransmission is necessary and delays are multiplied. We can then expect a negative association between TD and transmission velocity.

Looking at transmission velocity from the SNA perspective, when networks are denser there is a higher direct link proportion and then more direct paths between any pair of actors. Since direct paths avoid the need for indirect transmission, one can conclude that denser networks are associated with faster transmission velocity.

Centralised networks, even at the same density, concentrate links on one or few actors, and create an information transmission patterns with fewer direct paths. At low TD, direct paths can be used, but when TD starts becoming an issue, indirect paths are necessary and direct ones are stretched, which is congruent with the observation that more central developers in dispersed teams shift their working times more than the rest, and work longer hours (Carmel and Espinosa, 2010). We can then posit that as TD increases structures will become more centralised.

Parallelism is the number of simultaneous transmissions that can take place. Low TD allow the message to be sent to several team mates simultaneously, since many of them will share the same working hours, but when TD increases, the capacity of transmitting messages in parallel goes down, then supporting a negative association between TD and parallelism.

From the SNA perspective, the kind of structure that allows bigger parallelism is a denser structure, where any transmitter of information has multiple paths available to different recipients. Also, from the point of view of centralisation, a less centralised structure exhibits more parallel paths. We can then expect that higher parallelism associates with denser and less centralised structures, or as TD increases, parallelism decreases, and structures become sparser and more centralised.

Symbol sets are the number of ways in which a message can be encoded for transmission, for example verbal, visual, physical. Natural languages are also symbol sets, as they are programming languages, since for instance the same algorithm can be implemented in different programming languages. Symbol sets affect message transmission from two points of view. First, different sets require

different times to encode and decode, and second, different sets have varying degrees of precision and effectiveness to encode a given message, for instance a visual set such as blueprint is more effective to convey the meaning of a design than a verbal explanation of the same. In OSS development, observed symbol sets include written messages in natural languages, programming languages, IRC or real time instant messaging in natural languages and to a much lesser degree verbal communications, both synchronous and asynchronous.

While when TD increases and media become more asynchronous, those symbol sets that require synchronicity cease to be used (e.g., natural language in a phone conversation), while those that permit it such as e-mail may become more prevalent. It can then be expected that the variety of symbol sets will decrease with TD, exhibiting a negative association.

From the SNA perspective, the use of fewer symbol sets can be associated with fewer direct connections between actors since some actors are going to not be versed in the use of all symbol sets, or prefer all symbol sets equally, and some of the ties present in the network will disappear then producing a sparser network. Looking at centralisation, if fewer actors are adept to code or decode information in some symbol sets, those fewer actors that are expected to be more central as symbol sets are fewer. In summary, when examining symbol sets, we can expect that TD will have a negative association with number of symbol sets and density and a positive association with centralisation.

Rehearsability is the ability to fine tune or edit a message at the time of encoding, and before transmission. Since it occurs at the time of message encoding and before transmission, it is not considered here.

Reprocessability is the extent to which a medium enables the reexamination or editing of a message within or after the communication event. The ability to provide feedback can be understood as a form of reprocessability. For instance, this is what happens when developers with higher administrative standing vet other developers' contributions and often require changes or send the contributions to third parties before committing the changes to the source base.

With no TD all team members are synchronously available for feedback on contributions. When TD increases some links to synchronous communication with teammates are severed and then we can expect that TD is negatively associated with reprocessability.

From the SNA perspective, sparser networks hinder reprocessability because fewer counterparts are available to help modify the message. When comparing centralised versus decentralised networks, the latter allow less reprocessability since the message has to go through specific central actors stemming the variety of actors who can potentially give feedback. We can then expect that reprocessability is negatively associated with both density and centralisation. When TD increases,

Table 1. TD versus structural parameters.

		Expected effect when increasing TD	
Media capability	Association with TD	Density	Centralisation
Transmission velocity	−	−	+
Parallelism	−	−	+
Symbol sets	−	−	+
Rehearsability		Not considered	
Reprocessability	−	−	−

we should expect, from the perspective of reprocessability, sparser and decentralised structures.

Taking all the arguments together (Table 1), and with reprocessability versus centralisation being the oddity, when we consider the expected evolution of the media capabilities involved in the conveyance of information, we can overall expect that as TD increases, networks will become sparser and more centralised, and then our first two hypotheses can be laid out:

H1: *TD is negatively associated with collaboration structure density.*
H2: *TD is positively associated with collaboration structure centralisation.*

Also, MST posits that the five media capabilities support the ideal-level of synchronicity to "successfully support performance" (Dennis and Valacich, 2008, p. 575) and hence we are indirectly assuming that the first two hypotheses apply particularly to "successful" information transmission processes. The hypothesised associations may be weaker or even not be valid for those teams with less successful information transmission processes. Since communication and collaboration among team members is a predictor of overall team performance, (Guzzo and Dickson, 1996; Cummins and Cross, 2003; Powell and Piccoli, 2004) we can expect performance to moderate the relationship between TD and structure:

H3: *The association between TD and structure density is moderated by the team's performance.*
H4: *The association between TD and structure centralisation is moderated by the team's performance.*

Methods

Research setting

The public availability of data on OSS, including the product itself (source code) and other artifacts such as mailing lists, change logs, and so on, poses a great

opportunity for empirical research. OSS project team information is hosted in web-based repositories that have been used repeatedly as sources of archival data for empirical studies. In accordance with the majority of previous OSS empirical studies (e.g., Hahn *et al.*, 2008; Colazo and Fang, 2009), this research uses data collected from OSS projects hosted in Source Forge (SF) (www.sourceforge.net).

Sampling

The OSS project data were collected initially in early 2008, with updates to the database ran throughout mid 2011. Among all OSS projects hosted in SF, the projects analysed were restricted to those using "C" as their programming language. The use of a single programming language is strongly preferred when using code-based metrics (Jones, 1986), and "C" is a procedural language for which there are well-established metrics that are relatively easier to collect and crosscheck than if an object-oriented language had been used, in which case other aspects such as the need to consider coupling and cohesion would make data collection and analysis much more resource intensive. A program was considered written in "C" when at least 90% of their source code files were exclusively in that language.

In SF, an overwhelming majority of registered projects do not have any meaningful activity with no source code at all. Moreover, projects with only one developer are not representative of temporally dispersed project teams or even of projects of interest for business. Larger teams are more likely representative of projects that can develop popular, widely used software. Consequently and following previous empirical research on OSS, projects with six or more core team members were selected (Crowston and Howison, 2003). In accordance with the previous research (Mockus *et al.*, 2002), core team members are defined here as project members who have administrative rights to write source code in the repository.

In the SF repository, there were 587 software projects being developed by six or more core team members and using "C". These projects constituted the sampling frame. The final sample was reduced to 230 projects (39% of the sampling frame) because only that number had archived full data on all development activities. In spite of the non-probabilistic nature of the sample (see limitations), it contained projects with varied types of applications, sizes, and degrees of complexity. Data were cross-sectional, and the unit of analysis was the OSS project.

Measurement

Temporal dispersion

In extant research, TD has been measured in different ways: Knoll (2000) measured the mean and standard deviation of actual working hours of team members around

the Greenwich Meridian Time (GMT). O'Leary *et al.*, (2007) developed an index based on time zone differences among team members. McDonough *et al.* (2001) used an ordinal measure, categorising teams into co-located, virtual, and global.

Our data allowed us to measure the variation of actual work hours rather having to use time zone differences, the former being a better alternative as suggested by O'Leary *et al.*, (2007).

Following these prescriptions and in accordance to recent research (Colazo and Fang, 2010), TD was measured using the variance in the team members' starting work time, with such time expressed in a location-independent time unit: Universal Time Coordinated (UTC). For every day in a given time window, the time when each and every developer submitted his/her first contribution was recorded and the variance of those starting times calculated. The mentioned time window was set as a month, but results do not change in significance if the time window is set at a quarter or at a semester.

Developer activity time was observed from two different sources: time stamps in the Subversion log files and time stamps recorded in the developers' e-mail lists. Subversion log information was already in UTC regardless of where the code changes came from. E-mail time stamps were not in UTC, but they were transformed into UTC by noting the time zone recorded in the e-mail exchange log (e.g., 10:45 PM UTC+5 was recoded into 5:45 PM UTC). Subversion logs and e-email logs were then parsed for submission times with custom-made scripts written in Practical Extraction and Report Language (PERL).

Structure

Collaboration structure was measured following metrics common in social network analysis (Wasserman and Faust, 1994). As mentioned before, the collaboration structure in the team is characterised by two main parameters: density and centralisation.

Density

A graph is defined as a collection of nodes and arcs linking them. In a graph, density is defined as the ratio of the number of arcs to the maximum possible number of arcs. In a valued graph (one in which arcs have a value or "strength") this concept is extended by assigning to each arc its value and to each missing arc the value of 0, and density's value is exactly the average strength of the arcs.

Centralization

There are several measures of the centralisation of a social structure, such as degree centralisation, betweenness centralisation, etc. (Wasserman and Faust,

1994). Information centralisation is the only centralisation measure that is designed to accommodate valued networks, and to conceptually portray the flow of information among actors (Wasserman and Faust, 1994).

This metric is agreeable with this paper's theoretical approach, grounded on a theory involving the transmission of information. Conceptually, information centralisation represents how much information flows through each of the nodes in a network. It is a variation of betweenness centralisation that considers not only geodesics, but all possible paths deriving from actors.

In order to calculate the information centralisation for a team, we need to first do so with information centrality for the individual actors of the network. While details of the calculation of information centrality are tedious and can be found elsewhere (Stephenson and Zelen, 1989), in a nutshell, all paths containing the specific actor are given a value representing the strength of the information signal flowing through that path. The information value for a specific actor is the harmonic average of all signals containing information flowing through that actor, excluding actors that are isolates.

In turn, the network's information centralisation can be obtained as either the variance of the individual information centralities or alternatively the ratio between the network's average centrality and the maximum possible theoretical average centrality given the specific topology of the network (Freeman, 1977). The latter is the definition we used here.

The network data was obtained from the activity logs of each project's Subversion, a standard tool that registers each modification a team member makes to any file belonging to the project; it is used to prevent programming conflicts in multi-member projects. The log files contain, among other information, the name of the file modified, the name of the team member making the modification, the kind of modification made, the number of lines added, lines deleted, and lines modified.

A custom-made script downloaded these logs and extracted the details of who worked on each file. The number of team members was calculated from the logs as the number of different individuals who made changes to the project's files.

Another script translated the Subversion log information into associative data relating each possible team member dyad to the files they worked on in common, if any. These network data were fed to "R" (R Development Team, 2012) to obtain the structural parametres (network density and network centralisation).

Team performance

Two measure of performance were selected: productivity and quality. Productivity was measured by the number of lines of code written per developer per month, a popular and established measure for development productivity (cf. Jones, 1986).

Code quality was measured by the expected number of pre-test bugs per KSLOC (Ottenstein, 1981), standardised per thousand lines of source code (B/KSLOC). This measure estimates the expected number of defects latent in the source code, and it has been validated (Gremillion, 1984) against actually found quality defects in the software testing stage. Note that a higher defect count corresponds to lower software quality.

Other variables/controls

The sampling design controlled for programming language to eliminate potential inconsistencies in the metrics based on source code. Project tenure was measured by counting the number of days between the first known date of activity in the source code repository and the date the measurements were taken. Project size was measured in total lines of source code. The number of developers was taken from the Subversion logs.

Results

The hypotheses were tested using ordinary least squares (OLS) regression. All variables were log-transformed to increase linearity, except for project tenure, which was inverse-transformed. Bivariate correlations are shown in Table 2.

H1 and H2 were supported. Results of the OLS regression, with density and centralisation as dependent variables, are shown in Table 3. Regression coefficients show that temporal dispersion is in fact negatively associated with network

Table 2. Correlations.

	TD	Density	KSLOC/ Dev/month	Centralisation	Project tenure	Size	Developers	B/KSLOC
TD	1	−0.453***	−0.047*	0.455***	−0.121***	0.265***	0.625***	0.163***
Density			0.075*	0.468***	0.119***	0.296***	0.223***	0.248***
KSLOC/ Dev/month				−0.012	0.103***	0.474***	−0.060*	0.042
Centralization					−0.007	0.243***	0.529***	0.154***
Project tenure						0.336***	−0.084**	0.241***
Size						1	0.310***	0.615***
Developers							1	0.214***
B/KSLOC								1

Note: All variables log transformed, project tenure inverse transformed.
***$p < 0.001$.
**$p < 0.01$.
*$p < 0.1$

Table 3. OLS results, overall.

	Density	Centralisation
Constant	0.253	−1.970***
TD	−1.934***	0.749***
Project tenure	0.033	−0.062*
Project size	0.087	0.581***
Developers	−0.006	0.647***
N	230	230
F	42***	95***
R^2	0.21	0.36

***$p < 0.001$.
*$p < 0.05$.

density ($p < 0.001$) and positively associated with network centralisation ($p < 0.001$).

Additional OLS models were run splitting the sample into groups by performance. The two performance measures selected were productivity and quality. Results are presented in Tables 4 and 5 for subsamples obtained by dividing the main sample at the mean value of either productivity or quality.

H3 and H4 were partially supported. Looking at the OLS results for the sample split at low and high productivities, while the association between TD and density remained at a similar strength, the significance level is higher for the higher

Table 4. OLS results, sample split by productivity.

		Density	Centralisation
Lower productivity	Constant	−0.853*	−2.228***
KSLOC/DEV/month < 6	TD	−1.488**	0.612*
	Project tenure	0.097	−0.060
	Project size	0.320**	0.639***
	Developers	0.383**	0.710***
	N	89	
	F	7	48
	R^2	0.10	0.420
Higher productivity	Constant	0.673*	−1.875***
KSLOC/DEV/month >= 6	TD	−1.383***	0.614***
	Project tenure	0.037	−0.059
	Project size	0.014	0.562***
	Developers	−0.073	0.650***
	N	111	
	F	14	42
	R^2	0.11	0.29

Table 5. OLS results, sample split by quality.

		Density		Centralization	
Lower quality	Constant	−0.692	0.296	−2.908	0.000
B/KSLOC > = 63	TD	−1.644***		0.349	
	Project tenure	0.182		0.076	
	Project size	0.317*		0.800***	
	Developers	0.264		0.931***	
	N			98	
	F	12***		22***	
	R^2	0.22		0.380	
Higher quality	Constant	0.560*		−1.783***	
B/KSLOC < 63	TD	−2.027***		0.841***	
	Project tenure	−0.035		−.111**	
	Project size	0.020		0.537***	
	Developers	−0.113		0.591***	
	N			132	
	F	44***		78***	
	R^2	0.28		0.37	

performing group. When the regression with centralisation as DV is observed, the association between TD and centralisation is only significant for the higher productivity group.

Looking at the results with the sample split into two groups by quality, the effect of TD on density is stronger for the higher quality group, while the coefficient for TD is significant for centralisation only in the higher quality group. Although not uniformly strong, there is then support for the moderating effect of performance in the association between TD and structural parameters.

Conclusions, Limitations and Future Research

This study is the first to support the relationship between varying degrees of TD in distributed development teams and their collaboration structures. Based on MST and elements of SNA we show that the collaboration structure networks of those OSS teams that are more temporally dispersed are sparser and more centralised, associations that are stronger in those teams exhibiting higher relative performance.

This paper is also the first to link MST's drivers of media synchronicity with the concept of TD, by examining how synchronicity and the associated media capabilities should correlate when teams become more temporally dispersed.

This paper is also one of the few to empirically implement the use of the concept of information centralisation (Stephenson and Zelen, 1989) to describe a collaboration structure, making use of a metric that is more specialised that others such as degree centralisation or betweenness centralisation but is particularly well adapted to the study's research framework.

The observed support for H1 and H2 suggests that as teams become temporally dispersed ties between members are severed. We can speculate that those ties that are cut first are those corresponding to developers who work shorter hours, are more specialised or have a smaller skill set in terms of programming languages or access to asynchronous media.

Support for H3 and H4 show that the expected associations between TD and structural parameters predicted by the changes foreseen in media capabilities, are sensitive to the general performance of the team; i.e., team performance moderates the associations. More successful teams in terms of both quality and productivity, seem to rely more on "linchpin developers" that become more active as the team's TD grows and serve as "bridges" between time zones. MST suggests that these more central developers are those who can effectively use a wide array of both synchronous and asynchronous media, as well as are versed in many symbol sets in use for the development of code, such as multiple programming languages. These central developers will likely have to work more flexible hours and shift their working hours to a greater extent than the rest of the team members (Carmel and Espinosa, 2010). The role of central developers seems to be more important for those projects that emphasise quality more than coding productivity.

From the practical point of view, managers of temporally distributed teams should assure that as TD increases in their teams, a solid group of central developers are kept in the loop of code changes. MST suggests that these more central developers should be selected with an eye on their skills for effectively using an array of both synchronous and asynchronous media, as well as versed in most symbol sets in use for the development of code, such as multiple programming languages. These central developers will likely have to work more flexible hours and shift their working hours to a greater extent than the rest of the team members. The role of central developers seems to be more important for those projects that emphasise quality more than coding productivity.

While a major strength of this study is the "hard" nature of the data and metrics, limitations should be considered. First, using SF data in OSS empirical studies presents three major issues: the lack of integrity of the downloaded data, the need to cross-check data with other available sources, and the need to clean the data (Howison and Crowston, 2004). All these problems were addressed by a manual review of the projects in the sample. Samples of the source code were manually inspected to confirm the programming language and line counts. We always used

only the main branch of Subversion, and a small number of projects were built from source and compiled to confirm that they represented somewhat complete software and no major blocks of code were missing from our analysis. We also double-checked all static metrics with one or two off-the-shelf analysers, depending on the availability of the metrics in such packages (Scitools, 2012; Testwell Oy, 2012).

Second, the sample is non-probabilistic and thus does not represent the universe of OSS projects. However, it is not the intention of this study to generalise findings to small projects or to those that never progressed to the coding phase.

Third, our sample ignores programs written in programming languages other than "C", and in particular the results should be cross checked with object oriented programming languages such as Java or C++, where coupling and cohesiveness become more salient than traditional static metrics (Chidamber and Kemerer, 1994).

A theoretical limitation is that we considered the MST's five media capabilities in the hypotheses development but did not measure these capabilities directly, but only indirectly through the foreseeable effect of those media capabilities on structure.

Future research should replicate these findings with not only software written in other programming languages, but also in proprietary software development environments, and even in temporally dispersed teams working in realms different from software development.

Structural models with additional variables, and perhaps measuring the media capabilities directly, may yield further insights into the complex relationships among structure and TD. Additionally, the moderating effect of performance should be investigated using alternative measures and other data sources such as surveying dispersed teams.

References

Allen, TJ (1977). *Managing the Flow of Technology*. Cambridge, MA: MIT Press.

Burke, K, KJ Aytes, Chidambaram and JJ Johnson (1999). A study of partially distributed groups: The impact of media, location and time on perceptions and performance. *Small Group Research*, 30(4), 453–490.

Cao, X, DR Vogel, X Guo, H Liu and J Gu (2012). *Understanding the Influence of Social Media in the Workplace: An Integration of Media Synchronicity and Social Capital Theories*. System Science (HICSS), IEEE.

Carmel, E and JA Espinosa (2010). *I'm Working While They Sleep: Time Zone Separation Challenges and Solutions*. Washington, DC: Nedder Stream Press.

Chidamber, SR and CF Kemerer (1994). A metrics suite for object-oriented design. *IEEE Transactions on Software Engineering*, 20, 476–493.

Colazo, JA (2007). Innovation success: An empirical study of software development projects in the context of the open source paradigm. The University of Western Ontario, Canada.

Colazo, JA (2010). Collaboration structure and performance in new software development: Findings from the study of open source projects. *International Journal of Innovation Management*, 14(5), 735.

Colazo, JA and Y Fang (2009). Impact of license choice of open source software development activity. *Journal of the American Society for Information Science and Technology*, 60(5), 997–1011.

Colazo, JA and Y Fang (2010). Following the sun: Temporal dispersion and performance in open source software project teams. *Journal of the Association for Information Systems*, 11(11), 684–707.

Cramton, CD (2001). The mutual knowledge problem and its consequences for dispersed collaboration. *Organization Science*, 12(3), 346.

Crowston, K and J Howison (2003). *The Social Structure of Free and Open Source Software Development.* 24th Int. Conf. Information Systems, Seattle, WA.

Crowston, K, Q Li, K Wei, UY Eseryel and J Howison (2007). Self-organization of teams for free/libre opensource software development. *Information and Software Technology*, 49(6), 564–575.

Cummins, JN and R Cross (2003). Structural properties of work groups and their consequences for performance. *Social Networks*, 25(3), 197–210.

Dempsey, BJ (2002). Who is an open source software developer? *Communications of the ACM*, 45(2), 67–72.

Dennis, AR and JS Valacich (1999). *Rethinking Media Richness: Towards a Theory of Media Synchronicity.* 32nd Hawaii Int. Conf. System Sciences, Maui, HI, AIS.

Dennis, AR and JS Valacich (2008). Media, tasks and communication processes: A theory of media synchronicity. *MIS Quarterly*, 32(3), 575–600.

Duchenaut, N (2005). Socialization in an open source software community: A socio-technical analysis. *Computer Supported Cooperative Work*, 14(4), 323–368.

Espinosa, J, W DeLone and G Lee (2006). Global boundaries, task processes and IS project success: A field study. *Information Technology & People*, 19(4), 345–370.

Espinosa, JA and E Carmel (2003). The impact of time separation on coordination in global software teams: A conceptual foundation. *Software Process Improvement and Practice*, 8, 249–266.

Fitzgerald, B (2006). The transformation of open source software. *MIS Quarterly*, 30(3), 587–598.

Freeman, LC (1977). A set of measures of centrality based on betweenness. *Sociometry*, 40(1), 35–41.

Freeman, LC (1979). Centrality in social networks: I. Conceptual clarification. *Social Networks*, 1, 215–239.

Gremillion, LL (1984). Determinants of program repair maintenance requirements. *Communications of the ACM*, 27(8), 826–832.

Griffith, TL, JE Sawyer and MA Neale (2003). Virtualness and knowledge in teams: Managing the love triangle of organizations, individuals and information technology. *MIS Quarterly*, 27(2), 265–287.

Grinter, RE (2000). Workflow systems: Occasions for success and failure. *Computer Supported Cooperative Work*, 9, 189–214.

Guzzo, RA and MW Dickson (1996). Teams in organizations: Recent research on performance and effectiveness. *Annual Review of Psychology*, 47, 307–338.

Hahn, J, JY Moon and C Zhang (2008). Emergence of new project teams from open source software developer networks: Impact of prior collaboration ties. *Information Systems Research*, 19(3), 369–391.

Hassell, M and M Limayem (2011). A portfolio of media: Effects of media synchronicity on communication performance.

Howison, J and K Crowston (2004). *The Perils and Pitfalls of Mining SourceForge*. Workshop on Mining Software at the International Conference on Software Engineering, Edinburgh, Scotland, U.K., ICSE.

Jarvenpaa, SL, K Knoll and DE Leidner (1998). Is anybody out there? Antecedents of trust in global virtual teams. *Journal of Management Information Systems*, 14(4), 29–64.

Jones, C (1986). *Programming Productivity*. New York, NY: McGraw-Hill.

Kiesler, S and JM Cummins (2002). What do we know about proximity and distance in work groups? In *Distributed Work*, PJ Hinds and S Kiesler (Eds.). Cambridge, MA: MIT Press.

Kirkman, BL and JE Mathieu (2005). The dimensons and antecedents of team virtuality. *Journal of Management*, 31(5), 700–718.

Knoll, K. (2000). Communication and cohesiveness in virtual teams. *Doctoral Dissertation*, Austin, TX.

Labianca, G, H Moon and I Watt (2005). When is an hour not 60 minutes? Deadlines, temporal schemata, and individual and task group performance. *Academy of Management Journal*, 48(4), 677–694.

Majchrzak, A, RE Rice, A Malhotra, N King and S Ba (2000). Technology adaptation: The case of a computer-supported inter-organizational virtual team. *MIS Quarterly*, 24(4), 569.

McDonough, EF, K Kahn and G Barczaka (2001). An investigation of the use of global, virtual and colocated new product development teams. *Journal of Product Innovation Management*, 18, 110–120.

McGrath, J (1991). Time, interaction and performance (TIP): A thoery of groups. *Small Group Research*, 22, 147–174.

Mockus, A, RT Fielding and JD Herbsleb (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, 11(3), 309–346.

Montoya-Weiss, MM, AP Massey and M Song (2001). Getting it together: Temporal coordination and conflict management in global virtual teams. *Academy of Management Journal*, 44(6), 1251–1262.

O'Leary, MB and JM Cummins (2007). The spatial, temporal and configurational characteristics of geographic dispersion in teams. *MIS Quarterly*, 31(3), 433–452.

Ocker, R, SR Hiltz and SR Turoff and J Fjermestad (1995). The effects of distributed group support and process structuring on software requirements development teams: Results on creativity and quality. *Journal of Management Information Systems*, 12(3), 127–153.

Olivera, F, PS Goodman and SS-L Tan (2008). Contrinution behaviors in distributed environments. *MIS Quarterly*, 32(1), 23–42.

Ottenstein, L (1981). Predicting numbers of errors using software science. *ACM SIGMETRICS Performance Evaluation Review*, 10(1), 157–167.

Ou, CXJ, RM Davison M Robert, X Zhong and L Yi (2010). Empowering employees through instant messaging. *Information Technology & People*, 23(2), 193–211.

Powell, A, G Piccoli (2004). Virtual teams: A review of current literature and directions for future research. *The DATA BASE for Advances in Information Systems*, 35(1), 6–36.

R Development Team (2012). R v2.15.0.

Rasters, G, G Vissers and B Dankbaar (2002). An inside look: Rich communication through lean media in a virtual research team. *Small Group Research*, 33, 718–754.

Scitools (2012). Understand, Scientific Toolworks Inc.: Static Metrics Analyzer.

Singh, PV, Y Tang and V Mookerjee (2011). Network effects: The influence of structural capital on open source project success. *MIS Quarterly*, 35(4), 813–829.

Snijders, T (1981). The degree variance: An index of graph heterogeneity. *Social Networks*, 3, 163–174.

Stephenson, K and M Zelen (1989). Rethinking centrality: Methods and applications. *Social Networks*, 11, 1–37.

Te'eni, D (2001). Review: A cognitive-affective of organizational communication of designing IT. *MIS Quarterly*, 25(2), 251–312.

Testwell Oy (2012). CMT++. Hermia, Finland, Testwell: Software Static Metrics Analyzer.

Warkentin, M, L Sayeed and R Hightower (1997). Virtual teams versus face-to-face teams: An exploratory study of a web-based conference system. *Decision Sciences*, 28(4): 975–997.

Wasserman, S and K Faust (1994). *Social Network Analysis: Methods and Applications*. Cambridge, UK: Cambridge University Press.

Yamauchi, Y, M Yokozawa and T Shinohara and T Ishida (2000). *Collaboration with Lean Media: How Open-Source Software Succeeds*. Computer-Supported Collaborative Work, Philadephia, PA.