Structural Changes Associated with the Temporal Dispersion of Teams: Evidence from Open Source Software Projects

Jorge Colazo
Trinity University
jcolazo@trinity.edu

Abstract

This study relies on Media Synchronicity Theory and Social Network Analysis to analyze how the structure of collaboration networks change when collaborating teams become temporally dispersed. The empirical test of hypotheses using ordinary least squares with archival data from 230 Open Source Software projects shows that the collaboration structure networks of more temporally dispersed teams are sparser and more centralized, and these associations are stronger in those teams exhibiting higher relative performance.

1. Introduction

The structure of distributed teams has repeatedly been described under the social network paradigm [1], where members are the nodes of a network, and ties are present when two members share work in a specific part of the project [e.g. 2, 3, 4].

Graph-based metrics are used to characterize those networks using different structural characteristics such as network density or network centralization [1]. These features define the collaboration dynamics of the team and have been linked to various measures of performance such as productivity, quality and member permanence [4].

Also, in distributed teams members are dispersed both geographically and temporally, (i.e., they work from different places and times zones, at different times). The concept of temporal dispersion (TD) complements the traditional idea of geographical dispersion when studying team features and performance, and is receiving increased attention in academia and practice. For instance, many software development projects pursue a "Follow the Sun" approach where at the end of their day developers hand over work to colleagues in other time zones [5, 6].

Temporally dispersed teams seem to in general underperform synchronous teams [6], but there are some exceptions [5], and moderating factors that could explain the difference in performance across temporally dispersed teams have not yet been identified. It is plausible to consider that varying degrees of TD have correlates in the characteristics of the teams' collaboration structure, and that those teams that cope best with TD in part do so because their structure is better suited for such dispersed work. This paper investigates the characteristics of such collaboration structures associated with higher relative team performance.

On the one hand, neither theoretically nor empirically have researchers studied to date the possible associations between the degree of TD in teams and the structure of their collaboration networks. Our first research question is: *How does the collaboration structure of teams change with the team's TD?*

On the other hand, if there is a correspondence between TD and network structure, it is interesting to examine whether better performing teams have a different kind of association between TD and structure than worse performers. Thus the second research question is: *Are low and high performing teams different in terms of their association between TD and the team's structure?*

Answering these questions can help project managers to custom fit the team's collaboration policies to obtain the best possible performance given their unique TD pattern.

We tackle these questions through two concurrent theoretical lenses. Media Synchronicity Theory (MST) [7, 8] is used to theorize 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 derive how, to conserve synchronicity and support performance, structural changes should accompany changes in the capabilities mentioned by MST.

Hypotheses about the association between TD and structural parameters are tested using regression models with empirical data from archival sources belonging to 230 Open Source Software (OSS) projects.

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

2. Literature review and theoretical background

2.1 Temporal Dispersion

Team dispersion has had different conceptualizations depending on the aspect of the team that is disseminated; for instance dispersion in contributed effort [9], or in team member turnover [10]. However, the most often studied trait is the dispersion in distance between team members [11, 12]. Geographic dispersion often concurs with and sometimes masks the presence of TD [5].

TD is defined as the variation in working hours throughout members of a team [5]. TD is often associated with geographic dispersion, since geographically dispersed team members often will 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 many instances even members of collocated teams prefer the use of communication technologies to a totally feasible face to face contact [13].

Recent research suggests that the concept of geographic dispersion is becoming outdated and should perhaps be dropped from future definitions of team dispersion [14]. Kiesler et al. [15] 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.

Spatial dispersion has been related to a reduction in spontaneous communication frequency mainly because it decreases the likelihood of face-to-face communication [16]. TD, however, not only minimizes spontaneous communications but also reduces real-time problem solving because it fundamentally decreases the potential for any form of synchronous interaction [17].

The decreasing level of synchronous interaction as teams become more distant generates coordination difficulties which become more salient as the TD of a team grows [18]. In general, mixed evidence exists on the impact of TD on team performance and the mechanisms through which TD may exert influence, with a predominantly negative view on its effects on virtual team performance [19], with exceptions [5].

On the one hand, several studies suggest that TD detracts from team member coordination and degrades communication quality [20]. In asynchronous communication environments, coordinating temporal patterns of group behavior is a significant challenge because the transmission of verbal cues is hindered, feedback is delayed, and interruptions and long pauses in communication can occur [21]. In addition, long lapses between communication events -as is often the case in temporally dispersed situations- can result in disjointed and discontinuous discussions [22]. As such, research has found that temporally dispersed virtual teams (e.g., global software development teams) face specific problems, such as increased coordination costs [23], additional barriers to conflict management [24], difficulty in assimilating atypical work hours and in meeting deadlines [25], as well as a substantial decrease in the attainability and effectiveness of leadership control over a team [26].

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 [27]. 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 (SCCS), also called “Versioning Systems,” such as CVS (Concurrent Versioning System) or the newer Subversion, are tools specifically designed to allow developers asynchronously contribute to the code base [28] and to facilitate coordination [29].

2.2 Media synchronicity theory

MST [7, 8] 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 transmissions that can take place simultaneously. The *number of symbol sets* is the number of ways the medium allows the message to be decoded, e.g. 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 takes place.

MST posits that convergence processes are more effective with media with capabilities showing a high degree of synchronicity, while conveyance accepts but does not require synchronicity, though for both purposes asynchronic communications are slower and in general less efficient.

Media synchronicity has been linked to team performance [30, 31] 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 [32]. Of the two processes; convergence and conveyance, we are here interested only in the latter, since it is the only one related to the transmission of messages between actors.

2.3 Social network analysis

The theory behind Social Network Analysis (SNA) [33] sees features of social actors as the result of the actors' embeddedness in multiple networks. In these networks, actors are nodes and ties are present when there is a specified relationship between two actors [1]. Graph theory provides a rich collection of parameters that can be used to characterize 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 centralization [1].

Group network structure has been previously studied using SNA in relation to group features, for instance distribution of knowledge [34] and project success [2].

Two of the more commonly used network characteristics are density and centralization. Network density is the ratio between the number of ties present and the maximum possible number of such ties. Density represents the intensity of collaboration; or how "close knit" the team is. Zero density corresponds to a graph where all actors are isolates (they do not collaborate), while density of one corresponds to a team where everyone collaborates with everyone else.

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" , but there are other less common definition of actor indices [1]. While actor centrality represents the importance of an actor, network centralization represents the dispersion on the actors' centrality. In networks with low centralization, all actors have the same relative importance, while in a network with high centralization importances vary widely. In accordance with most extant research in SNA [1], density and centralization are the two focal structural traits to be investigated in their relation to TD.

3. Research model and hypotheses

In this section we theorize how different degrees of TD are expected to associate with different network structures, in terms of their density and centralization. We first 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. Then, looking into SNA, we analyze what kind of structural changes suit each of the drivers' expected modifications.

From the MST perspective, a key to understand the effect of TD is the necessary shift from fully synchronous media, since 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. Note that some media can be used synchronously but can also accommodate varying degrees of asynchronicity. For instance a text message can be responded to either immediately or after it has been received.

From the SNA perspective, with the increase of TD, we face an increasing importance of message transmission trough indirect paths due to the temporal unavailability of the intended end recipient of the message, with the consequent alteration of performance in terms of transmission efficiency and effectiveness. For instance, developer “A”, working in New York will post a modification to the source code that is analyzed and modified by developer “B” in Los Angeles, 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, configuring a typical indirect transmission process.

While MST was conceived considering dyads of actors (senders and recipients) networked structures comfortably consider the indirect transmission of information through nodes in the network that are in between the original sender and the recipient, i.e. developers can receive information indirectly as well as directly. Some SNA concepts such as information centralization were explicitly designed to account for the transmission of information through both direct and indirect paths, shortest (geodesic) or otherwise.

As per MST, transmission velocity reflects on the time elapsed between the sender releases the message and the message is received. As TD increases, the overlap between working hours of sender and recipient diminishes, rendering ineffective synchronous media such as direct telephone calls and videoconferencing. Even with the use 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 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, more ties are present respective to the maximum possible number of ties, and hence there is a higher direct tie proportion with a higher likelihood of a direct path between any given pair of actors. Since direct paths avoid the need for indirect transmission, one can conclude that denser networks are associated with faster transmission velocity.

Centralized networks, even at the same density, concentrate ties on one or few actors. These more central actors act as information brokers, and create an information transmission pattern where direct paths are relatively few. At low TD there is no need for information brokers since synchronicity is possible and available direct paths can be used. We can expect brokers to be more active when TD starts becoming an issue. This is congruent with the observation that more central developers in dispersed teams shift their working times more than the rest, and work longer hours [6]. We can then posit that as TD increases structures will become more centralized. All in all, *decreased transmission velocity will associate with higher TD and with sparser and more centralized networks*.

Parallelism is the number of simultaneous transmissions that can take place. Low TD allow the message to be received by several teammates 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, because any transmitter of information has multiple paths available to different recipients. Also, from the point of view of centralization, a less centralized structure exhibits more parallel paths to reach recipients of information whilst a centralized structure requires in most cases the information to go through a central broker. We can then expect that *the higher the parallelism, the denser and less centralized structures will be*.

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. Symbol sets affect message transmission from two points of view. First, different sets require different times to encode and decode. Second, different sets have varying degrees of precision and effectiveness to encode a given message. Third, the skill level of the recipient in decoding the symbol set influences the speed and precision of the decoding. 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, and an engineer is faster to interpret blueprints and will catch more details than a layperson.

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 centralization, if fewer actors are adept to code or decode information in some symbol sets, those fewer actors 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 centralization*.

Rehearsability is the ability to fine tune or edit a message at the time of encoding. Since it occurs before transmission, it should not influence the conveyance process and 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 for an opinion before committing the changes.

With no TD all team members are synchronously available for feedback on contributions. When TD increases some ties to synchronous communication with teammates are severed, thus we can expect that TD will be negatively associated with reprocessability.

From the SNA perspective, sparser networks hinder reprocessability because fewer counterparts are available to help modify the message. When comparing centralized vs. decentralized 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 centralization*.

Taking all the arguments together (Table 1), and with only reprocessability vs. centralization at odds, we can overall expect that as TD increases, networks will become sparser and more centralized, and our first two hypotheses can be laid out:

H1: TD is negatively associated with team structure density

H2: TD is positively associated with team structure centralization

Also, MST posits that the five media capabilities support the ideal level of synchronicity to "successfully support performance" [7, p.575] and hence we are indirectly assuming that the first two hypotheses apply particularly to "successful" information transmission processes. The hypothesized associations may be weaker or not significant for those teams with less successful information transmission processes. Since communication and collaboration among team members is a predictor of overall team performance, [35, 36] 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 centralization is moderated by the team's performance.

Table 1: TD vs. Structural Parameters

		Expected effect when increasing TD	
Media capability	Association with TD	Density	Centralization
Transmission velocity	-	-	+
Parallelism	-	-	+
Symbol sets	-	-	+
Rehearsability	Not applicable		
Reprocessability	-	-	-

4. Methods

4.1 Research Setting

OSS is software whose source code is freely available to the public, allowing the modification and redistribution of the product. While flagship examples of OSS include well known programs such as Linux and Apache, there are successful and widely used examples of OSS in most types of applications.

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 [37] and their temporal work patterns may change even from day to day. 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, with a constant variation of the team's time signature as membership changes.

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 [38].

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. 10, 39], this research uses data collected from OSS projects hosted in Source Forge (SF) (www.sourceforge.net).

4.2 Sampling

The project data were collected initially in early 2008, with updates to the database ran periodically throughout mid 2012. Among all OSS projects hosted in SF, the projects analyzed 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 [40], and "C" is a procedural language for which there are well established metrics that are relatively easier to collect and crosscheck.

In SF, an overwhelming majority of registered projects do not have any meaningful activity with no source code at all. Projects with only one developer are not representative of temporally dispersed project teams, and 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 [41]. Core team members [28] are those who have 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 the OSS project.

4.3 Measurement

In extant research, TD has been measured in different ways: Knoll [42] measured the mean and standard deviation of actual working hours of team members around the Greenwich Meridian Time (GMT). O'Leary et al. [19] developed an index based on time zone differences among team members. McDonough et al. [43] used an ordinal scale (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. [19]. Following these prescriptions and in accordance to recent research [5], 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 SCCS log files and time stamps recorded in the developers' e-mail lists. SCCS 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). SCCS logs and e-mail logs were then parsed for submission times with custom-made scripts written in Practical Extraction and Report Language (PERL).

As mentioned before, the collaboration structure in the team is characterized by two main parameters: density and centralization.

Density is defined as the ratio of the number of ties to the maximum possible number of ties. There are several measures of the centralization of a social structure. Information centralization is the only centralization measure that is specifically designed to portray the flow of information among actors [1]. Conceptually, information centralization considers that an actor's importance is related to how much information flows through his/her node.

The network data was obtained from the activity logs of each project's SCCS, 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 detail 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 SCCS 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” [44] to obtain the structural parameters (network density and network centralization).

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. 40].

Code quality was measured by the expected number of pre-test bugs per KSLOC [45], standardized per thousand lines of source code (B/KSLOC). This measure estimates the number of defects latent in the source code, and it has been validated [46] against actually found quality defects in the software testing stage. Note that a higher defect count corresponds to lower software quality.

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 SCCS logs.

5. 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.

H1 and H2 were supported. Results of the OLS regression, with density and centralization as dependent variables, are shown in Table 2. Regression coefficients show that TD is in fact negatively associated with network density ($p<0.001$) and positively with network centralization ($p<0.001$).

Table 2: OLS results, overall

	Density		Centralization	
Constant	.253		-1.970	***
TD	-1.934	***	.749	***
Project tenure	.033		-.062	*
Project size	.087		.581	***
Developers	-.006		.647	***
N	230		230	
F	42	***	95	***
R^2	0.21		0.36	

*** p <0.001
*   p< 0.05

Additional OLS models were run splitting the sample into groups by performance (productivity and quality). Results are presented in Tables 3 and 4 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 performing group. When the regression with centralization as DV is observed, the association between TD and centralization 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 in stronger for the higher quality group, while the coefficient for TD is significant for centralization 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.

Table 3: OLS results, sample split by productivity

		Density		Central.	
Low productivity	Constant	-.853	*	-.228	***
	TD	-.488	**	.612	*
	Project tenure	.097		-.060	
	Project size	.320	**	.639	***
	Developers	.383	**	.710	***
	N	89		89	
	F	7		48	
	R^2	.10		.42	
	Constant	.673	*	-.875	***
High productivity					
	TD	-1.383	***	.614	***
	Project tenure	.037		-.059	
	Project size	.014		.562	***
	Developers	-.073		.650	***
	N	111		111	
	F	14		42	
	R^2	0.11		0.29	

Table 4: OLS results, sample split by quality

		Density		Centralization	
Low quality	Constant	-.692		-2.908	***
	TD	-1.644	***	.349	
	Project tenure	.182		.076	
	Project size	.317	*	.800	***
	Developers	.264		.931	***
	N			98	
	F	12	***	22	***
	R^2	0.22		.380	
High quality	Constant	.560	*	-1.783	***
	TD	-2.027	***	.841	***
	Project tenure	-.035		-.111	**
	Project size	.020		.537	***
	Developers	-.113		.591	***
	N			132	
	F	44	***	78	***
	R^2	.28		0.37	

6. Conclusions, limitations and future research

From the theoretical perspective, this study offers several contributions.

First, it breaks ground in showing the relationship between varying degrees of TD in distributed teams and their associated collaboration structures. Based on MST and elements of SNA we show that the collaboration structure networks of those teams that are more temporally dispersed are sparser and more centralized, associations that are stronger in those teams exhibiting higher relative performance.

Second, this paper is the first to link MST's drivers of media synchronicity with the concept of TD, opening a new theoretical avenue that can be used in future research to study TD.

Third, this paper is also one of the very few to empirically implement the concept of information centralization [47] to define a collaboration structure.

Fourth, we observed that the expected associations between TD and structural parameters are sensitive to the general performance of the team; i.e. team performance can be seen as a moderator. The rate of change of structural density with TD is not sensitive to productivity but it is in terms of code quality. Centralization associates more strongly to TD in those teams that exhibit both better code quality and / or higher productivity.

From the practical point of view, managers of temporally distributed teams should inform their team building strategy noting the different behavior more successful teams show.

First, although as expected all teams seem to sever some of their collaboration ties as teams become temporally dispersed (dropping ties with members that become out of temporal synch), relatively successful teams sever those ties at a higher rate. Although MST does not offer insights as to which ties are severed first, we can speculate that those are the more inefficient from the point of view of quality, for instance contributing similar ideas or offering mostly redundant knowledge or technologies [4, 48]. It is plausible that higher quality teams do not need to "cling" to those temporally distant and hence inefficient ties because in their case skills are evenly distributed along the different time zones rather than lumped into one or few, possibly distant, time zones. This would prescribe, for higher quality teams, avoiding an extreme concentration of skills in any one site, providing a continuum of capabilities across all time zones spanned.

Second, 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 [6]. The role of central developers seems to be more important for those projects that emphasize quality more than coding productivity. High performance teams should evenly distribute this kind of team member so that collaboration can be effectively centralized across time zones.

Limitations to these conclusions should be considered. First, using SF data in OSS empirical studies presents sampling issues [49], which in this case were addressed by a manual review of the projects in the sample and checking static metrics with off-the-shelf analyzers [50, 51]. Second, the sample is non-probabilistic and thus does not represent the universe of OSS projects or teams working in other kinds of projects. Third, 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 rather through their purported effect 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.

References

[1] Wasserman, S., and Faust, K., Social Network Analysis: Methods and Applications, Cambridge University Press, 1st edn, Cambridge, U.K., 1994.

[2] Singh, P.V., Tang, Y., and Mookerjee, V., "Network Effects: The Influence of Structural Capital on Open Source Project Success ", MIS Q., 35(4), 2011, pp. 813-829.

[3] 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", Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, 2008, pp. 2-11.

[4] Colazo, J.A., "Collaboration Structure and Performance in New Software Development: Findings from the Study of Open Source Projects", International Journal of Innovation Management, 14(5), 2010,

[5] Colazo, J.A., and Fang, Y., "Following the Sun: Temporal Dispersion and Performance in Open Source Software Project Teams", Journal of the Association for Information Systems, 11(11), 2010, pp. 684-707.

[6] Carmel, E., and Espinosa, J.A., I'm Working While They Sleep, 2010.

[7] Dennis, A.R., and Valacich, J.S., "Media, Tasks and Communication Processes: A Theory of Media Synchronicity", MIS Quarterly, 32(3), 2008, pp. 575-600.

[8] Dennis, A.R., and Valacich, J.S., "Rethinking Media Richness: Towards a Theory of Media Synchronicity", in (Editor, 'ed.'^'eds.'): Book Rethinking Media Richness: Towards a Theory of Media Synchronicity, AIS, Maui, HI, 1999

[9] Olivera, F., Goodman, P.S., and Tan, S.S.L., "Contrinution Behaviors in Distributed Environments", MIS Quarterly, 32(1), 2008, pp. 23-42.

[10] Colazo, J.A., and Fang, Y., "Impact of License Choice of Open Source Software Development Activity", Journal of the American Society for Information Science and Technology, 60(5), 2009, pp. 997-1011.

[11] Cramton, C.D., "The Mutual Knowledge Problem and Its Consequences for Dispersed Collaboration", Organization Science, 12(3), 2001, pp. 346.

[12] Majchrzak, A., Rice, R.E., Malhotra, A., King, N., and Ba, S., "Technology Adaptation: The Case of a Computer-Supported Inter-Organizational Virtual Team", MIS Quarterly, 24(4), 2000, pp. 569.

[13] Griffith, T.L., Sawyer, J.E., and Neale, M.A., "Virtualness and Knowledge in Teams: Managing the Love Triangle of Organizations, Individuals and Information Technology ", MIS Quarterly, 27(2), 2003, pp. 265-287.

[14] Kirkman, B.L., and Mathieu, J.E., "The Dimensons and Antecedents of Team Virtuality", Journal of Management, 31(5), 2005, pp. 700-718.

[15] Kiesler, S., and Cummins, J.M., "What Do We Know About Proximity and Distance in Work Groups?", in (Hinds, P.J., and Kiesler, S., 'eds.'): Distributed Work, MIT Press, Cambridge, MA, 2002

[16] Te'eni, D., "Review: A Cognitive-Affective of Organizational Communication of Designing It", MIS Quarterly, 25(2), 2001, pp. 251-312.

[17] Burke, K., Aytes, K.J., Chidambaran, L., and Johnson, J.J., "A Study of Partially Distributed Groups: The Impact of Media, Location and Time on Perceptions and Performance", Small Group Research, 30(4), 1999, pp. 453-490.

[18] Espinosa, J., Delone, W., and Lee, G., "Global Boundaries, Task Processes and Is Project Success: A Field Study", Information Technology & People, 19(4), 2006, pp. 345-370.

[19] O'leary, M.B., and Cummins, J.M., "The Spatial, Temporal and Configurational Characteristics of Geographic Dispersion in Teams", MIS Quarterly, 31(3), 2007, pp. 433-452.

[20] Warkentin, M., Sayeed, L., and Hightower, R., "Virtual Teams Versus Face-to-Face Teams: An Exploratory Study of a Web-Based Conference System", Decision Sciences, 28(4), 1997, pp. 975-997.

[21] Mcgrath, J., "Time, Interaction and Performance (Tip): A Thoery of Groups", Small Group Research, 22(1991, pp. 147-174.

[22] Ocker, R., Hiltz, S.R., Turoff, M., and Fjermestad, J., "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), 1995, pp. 127.

[23] Espinosa, J.A., and Carmel, E., "The Impact of Time Separation on Coordination in Global Software Teams: A Conceptual Foundation", Software Process Improvement and Practice, 8(2003, pp. 249-266.

[24] Montoya-Weiss, M.M., Massey, A.P., and Song, M., "Getting It Together: Temporal Coordination and Conflict Management in Global Virtual Teams", Academy of Management Journal, 44(6), 2001, pp. 1251.

[25] Labianca, G., Moon, H., and Watt, I., "When Is an Hour Not 60 Minutes? Deadlines, Temporal Schemata, and Individual and Task Group Performance", Academy of Management Journal, 48(4), 2005, pp. 677-694.

[26] Jarvenpaa, S., L., Knoll, K., and Leidner, D.E., "Is Anybody out There? Antecedents of Trust in Global Virtual Teams", Journal of Management Information Systems, 14(4), 1998, pp. 29-64.

[27] Rasters, G., Vissers, G., and Dankbaar, B., "An inside Look: Rich Communication through Lean Media in a Virtual Research Team", Small Group Research, 33(2002, pp. 718-754.

[28] Mockus, A., Fielding, R.T., and Herbsleb, J., "Two Case Studies of Open Source Software Development:

Apache and Mozilla", ACM Transactions on Software Engineering and Methodology, 11(3), 2002, pp. 309-346.

[29] Grinter, R.E., "Workflow Systems: Occasions for Success and Failure", Computer Supported Cooperative Work, 9(2000, pp. 189-214.

[30] Cao, X., Vogel, D.R., Guo, X., Liu, H., and Gu, J., "Understanding the Influence of Social Media in the Workplace: An Integration of Media Synchronicity and Social Capital Theories", in (Editor, 'ed.'^'eds.'): Book Understanding the Influence of Social Media in the Workplace: An Integration of Media Synchronicity and Social Capital Theories, IEEE, 2012, pp. 3938-3947.

[31] Hassell, M., and Limayem, M., "A Portfolio of Media: Effects of Media Synchronicity on Communication Performance", 2011,

[32] Ou, C.X.J., Davison, R.M., Zhong, X., and Liang, Y., "Empowering Employees through Instant Messaging", Information Technology & People, 23(2), 2010, pp. 193-211.

[33] Freeman, L.C., "Centrality in Social Networks: I. Conceptual Clarification", Social Networks, 1(1979, pp. 215-239.

[34] Rulke, D.L., and Galaskiewicz, J., "Distribution of Knowledge, Group Network Structure, and Group Performance", Management Science, 46(5), 2000, pp. 612-625.

[35] Cummins, J.N., and Cross, R., "Structural Properties of Work Groups and Their Consequences for Performance", Social Networks, 25(3), 2003, pp. 197-210.

[36] Powell, A., Piccoli, G., and Ives, B., "Virtual Teams: A Review of Current Literature and Directions for Future Research", The DATA BASE for Advances in Information Systems, 35(1), 2004, pp. 6-36.

[37] Dempsey, B.J., "Who Is an Open Source Software Developer?", Communications of the ACM, 45(2), 2002, pp. 67-72.

[38] Duchenaut, N., "Socialization in an Open Source Software Community: A Socio-Technical Analysis", Computer Supported Cooperative Work, 14(4), 2005, pp. 323-368.

[39] Hahn, J., Moon, J.Y., and Zhang, C., "Emergence of New Project Teams from Open Source Software Developer Networks: Impact of Prior Collaboration Ties", Information Systems Research, 19(3), 2008, pp. 369-391.

[40] Jones, C., Programming Productivity, McGraw-Hill, New York, 1986.

[41] Crowston, K., and Howison, J., "The Social Structure of Free and Open Source Software Development", in (Editor, 'ed.'^'eds.'): Book The Social Structure of Free and Open Source Software Development, Seattle, WA, 2003

[42] Knoll, K., "Communication and Cohesiveness in Virtual Teams", Doctoral Dissertation, Austin, TX, 2000,

[43] Mcdonough, E.F., Kahn, K., and Barczak, G., "An Investigation of the Use of Global, Virtual and Colocated New Product Development Teams", Journal of Product Innovation Management, 18(2001, pp. 110-120.

[44] R Development Team, "R V2.15.0", in (Editor, 'ed.'^'eds.'): Book R V2.15.0, 2012

[45] Ottenstein, L., "Predicting Numbers of Errors Using Software Science", ACM SIGMETRICS Performance Evaluation Review, 10(1), 1981, pp. 157-167.

[46] Gremillion, L.L., "Determinants of Program Repair Maintenance Requirements", Communications of the ACM, 27(8), 1984, pp. 826-832.

[47] Stephenson, K., and Zelen, M., "Rethinking Centrality: Methods and Applications", Social Networks, 11(1989, pp. 1-37.

[48] Janis, I.L., Victims of Groupthink, Houghton Mifflin, Boston, MA, 1972.

[49] Howison, J., and Crowston, K., "The Perils and Pitfalls of Mining Sourceforge", Workshop on Mining Software at the International Conference on Software Engineering, 2004

[50] Testwell Oy, "Cmt++", in (Editor, 'ed.'^'eds.'): Book Cmt++, 4.2 edn., Testwell, Hermia, Finland, 2012, pp. Software Static Metrics Analyzer.

[51] Scitools, "Understand", in (Editor, 'ed.'^'eds.'): Book Understand, 2.0 edn., Scientific Toolworks Inc., 2012, pp. Static Metrics Analyzer.