Temporal Analysis of Dynamic Collaboration Graphs of Open Source Software Development: Forking

Amir Azarbakht
School of Electrical Engineering and Computer Science
Oregon State University
Corvallis, Oregon
azarbaam@eecs.oregonstate.edu

ABSTRACT

Social networks are a ubiquitous part of our lives, and the creation of online social communities has been a natural extension of this phenomena. Free and Open Source Software (FOSS) development efforts are prime examples of how communities can be leveraged in software development, where group are formed around communities of interest, and depend on continued interest and involvement. Forking, either as a violent split or a friendly divide, affects the community. Most existing research on forking is post-hoc. The run-up to fork is seldom studied. This leaves a number of questions unanswered. In this study, we propose to use longitudinal social network analysis to study the evolution and social dynamics of FOSS communities. With these techniques we aim to identify better measures for influence and the shift of influence, measures associated with unhealthy group dynamics, e.g. a simmering conflict, as well as early indicators of major events in the lifespan of a community. One set of dynamics we are especially interested in, are those that lead FOSS projects to fork. This will help predict formation of unhealthy dynamics, which gives the community a heads-up when they can still take action to ensure the sustainability of the project.

Categories and Subject Descriptors

H.5.m [**Information interfaces and presentation (e.g., HCI)**]: [Miscellaneous]

General Terms

Measurement, Reliability, Human Factors

Keywords

Free and Open Source Software, Social Dynamics, Temporal Analysis, Forking, Longitudinal Social Network Analysis, ERGM, Separable Temporal Exponential Family Random Graph Models, STERGM, FOSS.

1. INTRODUCTION

Social networks are a ubiquitous part of our social lives, and the creation of online social communities has been a natural extension of this phenomena. Social media plays an important role in software engineering, as software developers utilize them to communicate, learn, collaborate and coordinate with others [47]. Free and Open Source Software (FOSS) development efforts are prime examples of how community can be leveraged in software development, where groups are formed around communities of interest, and depend on continued interest and involvement in order to stay alive [34].

Although the bulk of collaboration and communication in FOSS communities occurs online and is publicly accessible, there are still many open questions about the social dynamics in FOSS communities. Projects might go through a metamorphosis when faced with an influx of new developers or the involvement of an outside organization. Conflicts between developers' divergent visions about the future of the project might lead to forking of the project and dilution of the community. Forking, either as a violent split when there is a conflict or as a friendly divide when new features are experimentally added both affect the community [9].

Previous research on forking ranges from the studies of Robles et al. [40] that identified all significant FOSS forks since 1990, to the works of Baishakhi et al. [7] on post-forking porting of new features or bug fixes from peer projects. It encompasses works of Nyman on developers' opinions about forking [36], developers motivations for performing forks [31], the necessity of code forking as tool for sustainability [35], and Syeed's work on socio-technical dependencies in the BSD projects family [48].

Most existing research on forking, however, is post-hoc. It looks at the forking events in retrospect and tries to find the outcome of the fork; what happened after the fork happened; what was the cause of forking, and such. The run-up to the forking events are seldom studied. This leaves a number of questions unanswered: Was it a long-term trend? Was the community polarized, before forking happened? Was there a shift of influence? Did the center of gravity of the community change? What was the tipping point? Was it predictable? Is it ever predictable? We are missing that context.

Additionally, studies of FOSS communities tend to suffer from an important limitation. They treat community as a static structure rather than a dynamic process. Longitudinal studies on open source forking are rare.

Our research proposes to use temporal social network analysis to study the evolution and social dynamics of FOSS communities. With these techniques we aim to identify better measures for influence, and the shift of influence, measures associated with unhealthy group dynamics, e.g. a simmering conflict, as well as early indicators of major events in the lifespan of a community. One set of dynamics we are especially interested in, are those that lead FOSS projects to fork.

This paper is organized as follows: We present related literature on open source social communities. We then present the gap in the literature, and discuss why the issue needs to be addressed. After that, in methodology, we describe research goals and research questions, how data gathering, statistical modeling, and qualitative analysis are proposed to be done. At the end, we present preliminary results of our initial study, conclusions, the timeline for the proposed research, and threats to validity.

2. RELATED WORK

The free and open source software development communities have been studied extensively. Researchers have studied the social structure and dynamics of team communications [10][23][24][19][30], identifying knowledge brokers and associated activities [44], project sustainability [35][30], forking [34], requirement satisfation [15], their topology [10], their demographic diversity [27], gender differences in the process of joining them [26], and the role of age and the core team in their communities [50][14][2][3]. Most of these studies have tended to look at community as a static structure rather than a dynamic process [13]. This makes it hard to determine cause and effect, or the exact impact of social changes.

Post-forking porting of new features or bug fixes from peer projects happens among forked projects [7]. A case study of the BSD family, i.e. FreeBSD, OpenBSD, and NetBSD, all of which evolved from the same codebase found that 10-15% of lines in BSD release patches consist of ported edits, and on average 26-58% of active developers participate in porting per release. They also found that over 50% of ported changes propagate to other projects within three releases [7]. This shows the amount of work developers need to do to synchronize and keep up with development in parallel projects.

Visual exploration of the collaboration networks in the WebKit project was the focus of a study that aimed to observe how key events in the mobile-device industry affected the WebKit collaboration network over its lifetime. [49] They found that *coopetition* (both competition and collaboration) exists in the open source community; morover, they observed that the "firms that played a more central role in the WebKit project such as Google, Apple and Samsung were by 2013 the leaders of the mobile-devices industry. While more peripheral firms such as RIM and Nokia lost market-share [49]".

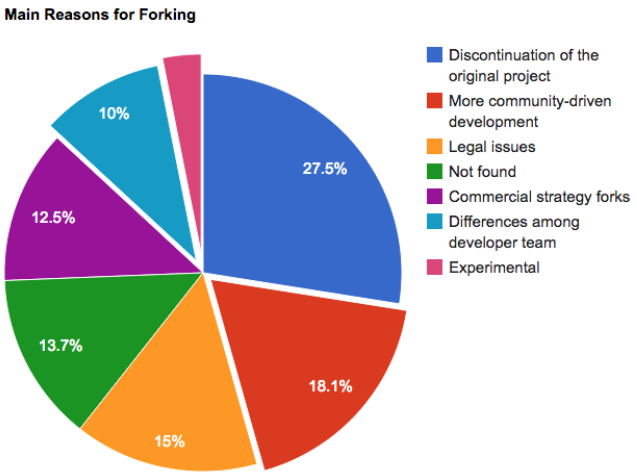
The study of communities has grown in popularity in part thanks to advances in social network analysis. From the earliest works by Zachary [51] to the more recent works of Leskovec et al. [28][29], there is a growing body of quantitative research on online communities. The earliest works on communities was done with a focus on information diffusion in a community [51]. Zachary investigated the fission of a community; the process of communities splitting into two or more parts. He found that fission could be predicted by applying the Ford-Fulkerson min-cut algorithm [16] on the group's communication graph; "the unequal flow of sentiments across the ties" and discriminatory sharing of information lead to "subcommunities with more internal stability than the community as a whole [51]."

Community splits in free and open source software development are referred to as forks, and are relatively common. Forking is defined as "*when a part of a development community (or a third party not related to the project) starts a completely independent line of development based on the source code basis of the project.*" The study by Robles and Gonzalez-Barahona [40] identified 220 significant FOSS projects that have forked over the past 30 years, and compiled a comprehensive list of the dates and reasons for forking. They classified these into six main categories. (Table 1.) which we build on extensively. They identified a gap in the literature in case of "how the community moves when a fork occurs [40]."

Table 1: The main reasons for forking as classified by Robles and Gonzalez-Barahona [40]

Reason for forking	**Example forks**
Technical (Addition of functionality)	Amarok & Clementine Player
More community-driven development	Asterisk & Callweaver
Differences among developer team	Kamailio & OpenSIPS
Discontinuation of the original project	Apache web server
Commercial strategy forks	LibreOffice & OpenOffice.org
Legal issues	X.Org & XFree

Figure 1: The frequency of reasons for forking as classified by Robles and Gonzalez-Barahona [40]



The dynamic behavior of a network and identifying key events was the aim of a study by Asur et al [1]. They studied three DBLP co-authorship networks and defined the evolution of these networks as following one of these paths: a) Continue, b) k-Merge, c) k-Split, d) Form, or e) Dissolve. They also defined four possible transformation events for individual members: 1) Appear, 2) Disappear, 3) Join, and 4) Leave. They compared groups extracted from consecutive snapshots, based on the size and overlap of every pair of groups. Then, they labeled groups with events, and used these identified events [1].

Table 2: The behavioral measures used by Asur et al. [1]

Metrics	Meaning
Stability	Tendency of a node to have interactions with the same nodes over time
Sociability	Tendency of a node to have different interactions
Influence	Number of followers a node has on a network and how its actions are copied and/or followed by other nodes. (e.g. when it joins/leaves a conversation, many other nodes join/leave the conversation, too)
Popularity	Number of nodes in a cluster (how crowded a sub-community is)

The communication patterns of free and open source software developers in a bug repository were examined by Howison et al. [23]. They calculated out-degree centrality as their metric. Out-degree centrality measures the proportion of the number of times a node contacted other nodes (outgoing) over how many times it was contacted by other nodes (incoming). They calculated this centrality over time "in 90-day windows, moving the window forward 30 days at a time." They found that "while change at the center of FOSS projects is relatively uncommon," participation across the community is highly skewed, following a power-law distribution, where many participants appear for a short period of time, and a very small number of participants are at the center for long periods. Our proposed approach is similar to theirs in how we form collaboration graphs. Our approach is different in terms of our project selection criteria, the metrics we examine, and our research questions.

The tension between diversity and homogeneity in a community was studied by Kunegis et al. [27]. They defined five network statistics, listed in Table 3, used to examine the evolution of large-scale networks over time. They found that except for the diameter, all other measures of diversity shrunk as the networks matured over their lifespan. Kunegis et al. [27] argued that one possible reason could be that the community structure consolidates as projects mature.

Community dynamics was the focus of a more recent study by Hannemann and Klamma [20] on three open source bioinformatics communities. They measured "age" of users, as starting from their first activity and found survival rates and two indicators for significant changes in the core of the community. They identified a survival rate pattern of 20-40-90%, meaning that only 20% of the newcomers survived after their first year, 40% of the survivors survived through the second year, and 90% of the remaining ones, survived over the next years. As for the change in the core, they suggested that a falling maximal betweenness in combination with an increasing network diameter as an indicator for a significant change in the core, e.g. retirement of a central person in the community. Our approach in our initial network-specific study built on top of their findings, and the evolution of betweenness centralities and network diameters for the projects in our study are depicted in the following sections.

3. MOTIVATION

The run-up to the forking events are seldom studied. There are unanswered questions about the run-up to a fork: Was it a sudden change, or a long-run trend? Was the community polarized or united, before forking happened? Was there a shift of influence? Did the center of gravity of the community change? What was the tipping point? Was it predictable? Is it predictable? We are missing that context.

To better understand and measure the evolution, social dynamics of forked FOSS projects, and integral components to understanding their evolution and direction, we need new and better tools. With this knowledge and these tools, we could help projects reflect on their actions, and help community leaders make informed decisions about possible changes or interventions. It will also help potential sponsors make informed decisions when investing in a project, and throughout their involvement to ensure a sustainable engagement.

Identification is the first step to rectify an undesirable dynamic before the damage is done. We want to map the dynamics of communities to real world phenomena. A community that does not manage growing pains may end up stagnating or dissolving. Managing growing pains is especially important in case of FOSS, where near half the project contributors are volunteers [17]. Oh et al. [37] have argued that openness in FOSS is "[...] generally perceived as having a positive connotation, however, the term can also be interpreted as referring to some unconstructive characteristics, such as unobstructed exit, susceptible, vulnerable, fragile, lacking effective regulation, and so on. The unobstructed exit and lack of regulatory force inherent in the OSS community can result in a community's susceptibility and vulnerability to herded exits by its participants. Commercial vendor intervention, an alternative project becoming available, and licensing issues can result in some original core members ceasing to provide their loyal service for the community, which can prompt their coworkers to leave as well" [37].

Recipes for success or stagnation, sustainability or fragmentation could be identifiable, leading to a set of best practices and pitfalls.

4. RESEARCH GOALS

Social interactions reflects the changes the community goes through, we argue. And so, it can be used to describe the context surrounding a forking event. Three of the six main reasons for forking [40], as listed in Table 1 are socially related, and so should be reflected in the social interactions data. For example, if a fork occurred because of a desire for "more community-driven development", we expect to see interaction patterns in the collaboration data showing a strongly-connected core that is hard to penetrate for the rest of the community. In other words, in this case, the power stays in the hands of the same people throughout, as new people come and go.

We aim to 1) Analyze, quantify and visualize how the community is structured, how it evolves, and the degree to which community involvement changes over time; 2) Test whether the results of the analysis match what developers in the project's community remember? Specifically, our research

Table 3: The measures of diversity used by Kunegis et al. [27]

Network property	Network is diverse when	Diversity Measures
Paths between nodes	Paths are long	Effective diameter
Degrees of nodes	Degrees are equal	Gini coefficient of the degree distribution
Communities	Communities have similar sizes	Fractional rank of the adjacency matrix
Random walks	Random walks have high probability of return	Weighted spectral distribution
Control of nodes	Nodes are hard to control	Number of driver nodes

objective and research questions are listed in the following.

4.1 Research Objective: What are the social patterns associated with different types of undesirable forking?

4.1.1 Do forks leave traces in the collaboration artifacts of open source projects?

To study the properties of possible social patterns, we need to verify their existence. More specifically, we need to check whether the possible social patterns are manifested in the the collaboration artifacts of open source projects, e.g. mailing list data, issue tracking systems data, source code data.

4.1.2 Do differences types of forks leave different types of traces?

If forks leave traces in the collaboration artifacts, do forks exhibit different social patterns? Are there patterns that exemplify these categories? For example, is there a prototypical "personal differences" fork collaboration pattern? If so, do different forking reasons have distinctly different social patterns associated with them?

Is a project labeled as a "*technical differences*" fork really only a "*technical differences*" fork? Or, alternatively, can they be a mix of several reason categories?

4.1.3 What are the key indicators that can tell us distinguish between different types of forks?

What establishes an inflection point (fork)? Which metrics are indicative of inflection? Is there a metric that can be used to monitor the odds of change, e.g. forking, ahead of time?

4.1.4 Does our analysis match what people in the community remember?

To validate what our quantitative data mining approach finds, we can compare it to what people remember of the situation, and what is written about that project. This validation check requires interviewing people from the studied forked projects. Semi-structured interviews need to be conducted, with as many developers from the forked projects, till the interviewers reach a point of saturation, i.e. when no new information is gained by doing more interviews. These semi-structured interviews will be recorded, transcribed, and coded, to find common patterns in the interviewee responses.

To complement the interview study, a literature review and web research will be done to gather data that was written about the project, before the fork and after the fork dates. The content of the messages send and received by the top contributors of the project in the month leading to the forking events can also be a read and analyzed to complement the findings of the interview study.

5. METHODOLOGY

To detect change patterns, we need to gather relevant data, clean it, and analyze it. In the following subsections, we describe the process in detail.

5.1 Phase 1: Data Collection

5.1.1 Not all forks are equal: Undesirable forking

To find patterns uniquely associated with undesirable forks, we need to gather data on projects from the following three categories:

Table 4: Not all forks are bad; Three types of forking for which data will be collected

Type of forking	Abbreviation
Undesirable forking	U.F.
Other (Healthy/Other) forking	H.F.
No forking at all	No.F.

Our results will only be valid if the patterns found in the *undesirable forking* (U.F.) category are unique to that category, and is not shared in the other two control groups (H.F. and No.F.).

To find projects in U.F. and H.F. categories, we looked at the list of all significant open source software forks in the past three decades as compiled by Robles and Gonzalez-Barahona [40]. They also found the reasons behind each fork, as listed in 1. We applied three selection criteria to the 220 projects on that list to find projects in U.F. and H.F. categories. A

Table 5: Forked projects (U.F. and H.F.) for which collaboration data was collected, and sociograms were formed

Projects	Reason for forking	Year	Type
Kamailio & OpenSIPS	Differences among developer team	2008	U.F.
ffmpeg & libav	Differences among developer team	2011	U.F.
Asterisk & Callweaver	More community-driven development	2007	U.F.
rdesktop & FreeRDP	More community-driven development	2010	U.F.
freeglut & OpenGLUT	More community-driven development	2004	U.F.
Amarok & Clementine Player	Technical (Addition of functionality)	2010	H.F.
ApacheCouchDB & BigCouch	Technical (Addition of functionality)	2010	H.F.
Pidgin & Carrier	Technical (Addition of functionality)	2008	H.F.

project was short-listed as a U.F. or H.F. if **a)** it was recent, i.e. happened after the year 2000, **b)** its data was accessible online, or was made accessible to us after our requests; and **c)** the project had a sizable developer community, i.e. more than a dozen developers; large enough to make a sociogram for our statistical analysis. This three-stage filtering process resulted in the projects listed in Table 5.

The data sources to collect are **a)** developer mailing lists, where developers' interact by sending and receiving emails, **b)** Issue(bug) tracking systems, where developers interact by reporting an issue/bug, discussing how to resolve it, and closing the issues, and , **c)** Source-code repository contribution logs, where developers interact by modifying the code, and/or working on the same source files. The sociograms will be formed based on interactions among developers in any of the above data sources. **d)** The content of the messages send and received by the top contributors of the project in the month leading to the forking events can be a good data source to be sentiment-analyzed.

The time period data to be collected for is the year in which the fork happened, as well as three months after the fork date. This should capture the social context prior, at, and after the time of the fork.

5.2 Phase 2: Sociogram Formation and Statistical Study

Many social structures can be represented as graphs. The nodes represent actors/players and the edges represent the interaction between them. Such graphs can be a snapshot of a network – a static sociogram – or a changing network, also called a dynamic sociogram. In this phase, we process interactions data to form a communication sociogram of the community. Two types of analysis can be done on sociograms; **a)** a *cross-sectional* study, and **b)** a *longitudinal* study. We are interested in patterns in the run-up to forks, so, we should do a longitudinal study, unlike most existing research on forking.

Our longitudinal study can look at the sociograms in two distinct approaches: **a)** a *network-specific* approach, in which, we focus on the actual networks under study, to describe the structure of the *observed network* with numerical summaries/descriptors to describe the structure and what we actually see, e.g. in Figures 3 and 4. Or, **b)** a *population-processes* approach, in which, we treat the observed network as one realization from a set of all possible networks with the same number of nodes, and with similar characteristics. To this end, the observed network, is only good to help us understand the social forces/processes that generated it.

5.2.1 Why bother find a statistical model?

One may wonder why we should look beyond the observed network. We need to do so because of the following reasons: **a)** For the *observed network*, a small observation error or sampling error, and the uncertainty involved with real-world communication, can result in large perturbation in the numerical descriptors used to describe static graphs. **b)** The traditional *network-specific* approach assumes edges in sociograms are statistically independent, (and/or identically distributed). This can be misleading, as, social network data is relational. For example, in real-life human communication, the likelihood of forming ties with friends of a friend is higher than a stranger, as Balance Theory suggests [21]. **c)** In population-specific approach, we try to identify the social forces that have formed the observed network, by simulation a population of similar networks of the same characteristics. After finding the statistical distribution of network population, then we can compare the observed network to the population distribution of possible networks of that size, and see how significant and likely it is to observe such a graph, as compared to observing a randomly-generated graph. This is useful, because it gives us a reference point to compare our observed graph with, weed out the properties generated by random processes, and find the statistically significant network statistics. In short, with a statistical model, we can do inferences about whether certain network structures and substructures are more commonly observed in the observed network than might be expected by chance [39]. **d)** Stochastic models capture the regularities in the processes that caused the network ties form, as well as variability that are hard to model otherwise. A model that considers stochasticity allows us to understand the uncertainty associated with an observed network. It makes it possible to learn about the distribution of possible networks for a given specification of a model [39]. **e)** Different social processes may manifest similar network structures. For example, clustering in a network might be because of structural effects, e.g. structural balance, or through node-level effects, e.g. homophily. To determine which one is the case in our observed network, a statistical model that incorporates both covariates can help. We then can assess the contribution of each covariate, and infer which social process underlies the observed network [39]. **f)** Localized processes might not scale to the entire network well. The combination of the overall structure and the localized processes is hard to investigate

Table 6: All projects forked because of "*personal differences among the developer team*" (U.F.) [40] in chronological order

Original	**Forked**	**Date**	**Data available?**	**Collected?**
GNU Emacs	X Emacs	1991, ?	Only after 2000	N/A
NetBSD	OpenBSD	1995, Oct	Yes, but scarce	N/A
xMule	aMule	2003, Aug	Only 2006-2007	N/A
lMule	xMule	2003, Jun	-	N/A
Sodipodi	Inkscape	2003, Nov	Yes	Req.
Nucleus CMS	Blog:CMS	2004, May	Only after Sept 2004	N/A
BMP	Audacious	2005, Oct	Yes	Req.
ntfsprogs	NTFS-3G	2006, Jul	-	N/A
OpenWRT	FreeWRT	2006, May	Only after Oct 2006	N/A
QtiPlot	SciDavis	2007, Aug	-	N/A
Kamailio	OpenSIPS	2008, Aug	Yes	**Yes**
Blastwave.org	OpenCSW	2008, Aug	-	N/A
jMonkeyEngine	Ardor3D	2008, Sept	Yes, but scarce	N/A
Frog CMS	Wolf CMS	2009, Jul	-	N/A
Aldrin	Neil	2009, ?	-	N/A
Ffmpeg	libav	2011, Mar	Yes	**Yes**

Table 7: All projects forked because of the need for "*more community-driven development*" (U.F.) [40] in chronological order

Original	**Forked**	**Date**	**Data available?**	**Collected?**
Nethack	Slash'EM	1996, ?	-	N/A
GCC	EGCS	1997, ?	-	N/A
SourceForge	Savane	2001, Oct	-	N/A
PHPNuke	PostNuke	2001, Sum	Not found	N/A
QTExtended	OPIE	2002, May	-	N/A
GraphicsMagick	Graphics	2002, Nov	Only after 2003	N/A
freeglut	OpenGLUT	2004, Mar	Yes	**Yes**
Mambo	Joomla!	2005, Aug	-	N/A
SER	Kamailio	2005, Jun	Only after 2006	N/A
PHPNuke	RavenNuke	2005, Nov	Not found	N/A
Hula	Bongo	2006, Dec	-	N/A
Compiere	A Dempiere	2006, Sept	No Dev. mailing list	N/A
Compiz	Beryl	2006, Sept	Only after Jun 2007	N/A
SQL-Ledger	LedgerSMB	2006, Sept	No Dev. mailing list	N/A
Asterisk	Callweaver	2007, Jun	Yes	**Yes**
CodeIgniter	KohanaPHP	2007, May	Not found	N/A
OpenOffice.org	Go-oo.org	2007, Oct	Only after Jun 2011	N/A
Mambo	MiaCMS	2008, May	-	N/A
TORCS	Speed Dreams	2008, Nov	Yes, but scarce	N/A
MySQL	MariaDB	2009, Jan	Yes	No
Nagios	Icinga	2009, May	Yes	Req
Project Darkstar	RedDwarf	2010, Feb	Yes, but scarce	N/A
SysCP	Froxlor	2010, Feb	-	N/A
Dokeos	Chamilo	2010, Jan	Not found	N/A
GNU Zebra	Quagga	2010, Jul	-	N/A
rdesktop	FreeRDP	2010, Mar	Yes	**Yes**
OpenOffice.org	LibreOffice	2010, Sept	Only after Jun 2011	N/A
Redmine	ChiliProject	2011, Feb	Yes, but scarce	N/A

without a model. (This micro-macro difference may be investigated through model simulation.)[39]

We need to find a well-fitting statistical model of our observed interactions network, because

5.2.2 *The statistical model*

A recent method for population-process approach is exponential family random graph models (ERGM) [39], in which, the possible edges between nodes of a graph are regarded as random variables. Assumptions about dependencies among

these random variables determine the form of the ERGM model for the network. The model parameters are estimated using the observed network, which in turn, can be used to interpret the underlying processes that generated the observed network. For example, we can infer whether a model parameter is significantly different from zero, and hence, the characteristic seen in the observed network is more or less likely than expected by chance.

The general form of the exponential family random graph models is [25]:

$$P(Y = y) = \frac{\exp(\theta' g(y))}{k(\theta)} \quad (1)$$

where:

- Y is the random variable for the state of the network,
- $g(y)$ is the vector of model statistics for network y,
- θ is the vector of coefficients for model statistics,
- $k(\theta)$ represents the quantity in the numerator summed over all possible networks with the same node set as y [25].

This can be written in terms of the conditional log-odds of a single actor pair [25]:

$$\text{logit}\left(Y_{ij} = 1 | y_{ij}^c\right) = \theta' \delta(y_{ij}) \quad (2)$$

where:

- Y_{ij} is the random variable for the state of the actor pair i, j (with realization y_{ij}),
- y_{ij}^c signifies the complement of y_{ij}, i.e. all dyads in the network other than y_{ij}.
- $\delta(y_{ij})$ equals $g(y_{ij}^+) - g(y_{ij}^-)$, where
- y_{ij}^+ is defined as y_{ij}^c along with y_{ij} set to 1,
- y_{ij}^- is defined as y_{ij}^c along with y_{ij} set to 0.
- That is, $\delta(y_{ij})$ equals the value of $g(y)$ when $y_{ij} = 1$ minus the value of $g(y)$ when $y_{ij} = 0$, but all other dyads are as in $g(y)$. This emphasizes the log-odds of an individual tie conditional on all others.
- $g(y)$ is called the *statistics* of the model, and $\delta(y_{ij})$ the "*change statistics*" for actor pair y_{ij} [25].

The model parameters are usually estimated using a pseudo-likelihood estimation (PLE) method or Markov chain Monte Carlo maximum likelihood estimation (MCMCMLE) [43].

As in the case of our longitudinal study of dynamic networks, we need to use an extension of ERGMs, namely, Separable Temporal Exponential Family Random Graph Models (STERGM), which takes two models, one for the formation, and another for dissolution [25].

5.2.3 Model Covariates

What covariates to include depends on the context, and the data available for developers. Table 8 lists several explanatory variables that can be included for modeling the community evolution. Table 9 lists what we expect to see associated with different types of forking. Specifically, for each of the following forking categories, we expect the following model parameters:

For "*personal differences*" forks (U.F.), we expect to see a decrease in reciprocity, a decrease in 3-cycles indicating an increase in hierarchy, an increase in betweenness, which highlights the brokerage position of emergent fork leaders, and an increase in diameter, when broker positions are emptied, and hence, the distance between pairs of developers in the network increases.

For "*more community-driven development*" forks (U.F.), we expect to see a decrease in 3-cycles, which indicates a stronger hierarchical structure, as well as an increase in positive assortativity, which highlights preference for homophile interactions.

For "*technical differences*" forks (H.F.), we expect to see a decrease in betweenness, as developers temporarily leave the main project to work on new features, and this, we suppose, is done by some central figures, whose absence at the center, will distance pairs of developers in the network.

For "*no forks at all*" projects (No.F.), we expect to see a steady progression, in contrast to all other projects that have forked.

5.2.4 Theories

Three theories about human behavior are the bases for the hypotheses in this research study.

1) Balance theory [21] talks about a motivation of individuals to move toward psychological balance. Balance theory considers cognitive consistency as a motive that drives sentiment or liking relationships, as well as liking of things created by or associated with the alter in the relationship.

We may explain U.F. forked projects due to "personal differences" as a results of cognitive inconsistency between the liking relationship of other developer(s) and the software and community created by or associated with them.

2) Signaling theory, from evolutionary biology, and economics [45], tries to explain communication between individuals, in terms of signals. As an example, an individual in a community will help others who are in need, if helping signals the helper's desirable personality. This signal helps in attracting mates and results in higher reproduction of that individual. In this scenario, the cost of helping another individual is less than the benefit gained by the individual because of signaling their conditions.

We may explain the increase of decrease in preferential attachment in U.F. projects, as a result of software developers signaling others in the community of their status, by helping others in need (i.e. peripheral community members), which in turn brings them social prestige.

Table 8: Explanatory variables to include in the initial model [46]

Network effect	**Network Statistic**	**Description**
Outdegree	$\sum_j x_{ij}$	Overall tendency to have ties (Negative parameter means, on average, cost of friendship ties higher than their benefits)
Reciprocity	$\sum_j x_{ij}x_{ji}$	Tendency to have reciprocated ties
Balance	$\sum_j x_{ij} strsim_{ij}$	Tendency to have ties to structurally similar others (structural equivalence with respect to outgoing ties)
Covariate similarity	$\sum_j x_{ij} sim_{ij}$	Tendency to have ties to similar others (homophile selection)
3-cycles	$\sum_j x_{ij} \sum_h x_{jh}x_{hi}$	Tendency to form relationship cycles (negative parameter means absense of hierarchy)
Betweenness	$\sum_j x_{ij} \sum_h x_{hi}(1 - x_{hj})$	Tendency to occupy an intermediate position between unrelated others (represents brokerage)
Transitive triplets	$\sum_j x_{ij} \sum_h x_{ih}x_{hj}$	Tendency toward triadic closure of the neighborhood (linear effect of the number of indirect ties)
Transitive ties	$\sum_j x_{ij} max_h(x_{ih}x_{hj})$	Tendency toward triadic closure of the neighborhood (binary effect of indirect ties)
Covariate alter	$\sum_j x_{ij}(z_j - \bar{z})$	Main effect of alter's behavior (covariate determines popularity in network)
Covariate ego	$\sum_j x_{ij}(z_i - \bar{z})$	Main effect of ego's behavior on tie preference (covariate determines activity in network)
Actors at distance 2	$\sum_j (1 - x_{ij}) max_h(x_{ih}x_{hj})$	Tendency to keep others at social distance 2 (negative measure of triadic closure; lower means stronger network closure)

3) Assortativity theories, which tries to explain collaboration, and people's preference for interacting with others who are similar to them in some way. This is related to reciprocity, in forms of contingent, direct, and indirect reciprocity. Homophily and Heterophily, which affect diversity of networks, are also closely related to assortative selection.

We may explain U.F. forked projects due to "more community-driven development" as a by-result of developers' homophile collaborator selection.

6. INITIAL STUDY: TEMPORAL NETWORK-SPECIFIC ANALYSIS

In our initual study[4][5][6], which was a network-specific study, we wanted to analyze the network-specific changes that happen to the community over a given period of time,

Table 9: Summary of expectations for each forking category

Model parameter	U.F. Pers. Dif.	U.F. More Comm.	H.F. Tech. Dif.	No.F.
Outdegree	-	-	-	-
Reciprocity	decrease	-	-	-
Balance	-	-	-	-
3-cycle	decrease	decrease	-	-
Transitive triplets	-	-	-	-
Transitive ties	-	-	-	-
Betweenness	increase	-	increase	-
Diameter	increase	-	-	-
Clustering Coefficient	-	-	-	-
Preferential Attachment	-	-	-	-
Assortativity	-	increase	-	-
CUSUM betweenness	sharp increase	-	increase	-

e.g. three months before and three months after the year in which the forking event happened. For this network-specific study, we measured betweenness centrality [11] of the most significant nodes in the graph, and the graph diameter over time. Figures 6, 7, and 8 show the betweenness centralities over the 1.5 year period for the Kamailio, Amarok and Asterisk projects respectively. To do temporal analysis, we had two options; 1) look at snapshots of the network state over time, (e.g. to look at the network snapshots in every week, the same way that a video is composed of many consecutive frames), and 2) look at a period through a time window. We preferred the second approach, and looked through a time window of three months wide with 1.5 month overlaps. To create the visualizations, we used a 3 months time frame that progressed six days a frame. In this way, we would have had a relatively smooth transition.

There are many ways of looking at an individual's importance/prestige/status within a network. One is called *closeness centrality*. The *farness* of a node is defined as the sum of its distances to all other nodes. The *closeness* of a node is defined as the inverse of the farness. More informally, the more central a node is the lower its total distance to all other nodes. *Closeness centrality* can be used as a measure of how fast information will spread through the network [12]. Secondly, if we are looking for people who can serve as bridges between two distinct communities, we could measure the node's *betweenness centrality*. Betweenness centralities for mediators who act as intermediate entities between other nodes are higher [12]. Third, if cross-community collaboration is the focus, we can measure *edge betweenness centrality*. Edges connecting nodes from different communities have higher edge centrality values. In the community collaboration graph, edge betweenness or stress of an edge is the number of these shortest paths that the edge belongs to, considering all shortest paths between all pairs of nodes in the graph. Fourth, one can claim that certain people in the community are more important than others, and whoever is close to them, is relatively more important than others. In graph terms, this is measured by *eigenvector centrality*, which is based on the assumption that connections to high-profile nodes contribute more to the importance of a node. Google's PageRank link-analysis algorithm [38] is a variant

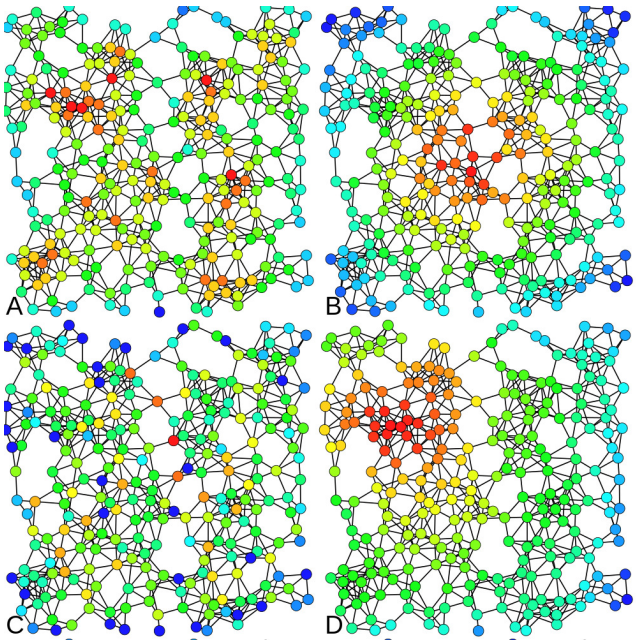


Figure 2: Heat-map color-coded examples of nodes with high centrality metric are shown above. The same network is analyzed four times with the following centrality measures: A) Degree centrality, B) Closeness centrality, C) Betweenness centrality and D) Eigenvector centrality [41]

of the eigenvector centrality measure. In short, centrality measures have been used in several studies to identify key player in a community.

In addition to the centrality measures, we planned to look into the *resilience* of the community as well. By resilience, we mean how well the network holds its structure and form when some parts of it are deleted, added, or changed. For a graph, the resilience of a graph is a measure of its robustness to node or edge failures. This could occur for instance when an influential member of the community leaves. Many real-world graphs are resilient to random failures but vulnerable to targeted attacks. Resilience can be related to the *graph*

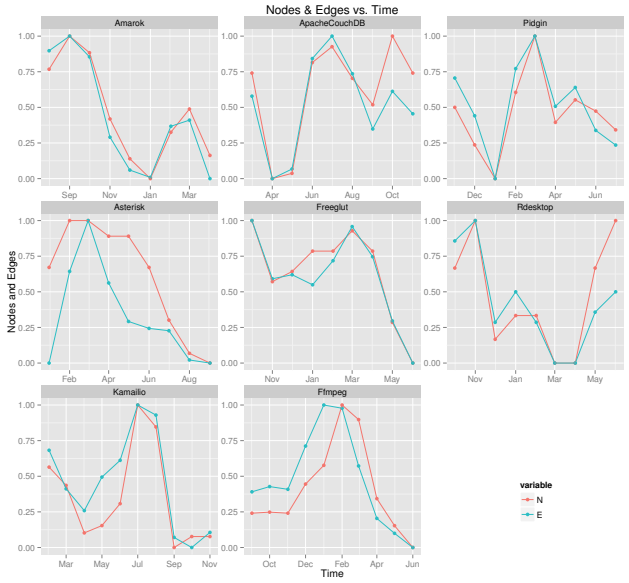


Figure 3: Nodes and Edges over Time. Note that the and number of nodes and number of edges are normalized to the range [0,1] to make comparison across projects meaningful, by emphasizing change in ratio, rather than the varying counts. Hence, the measurements were normalized for drawing this graph. The three projects in the first row belong to the *"technical differences"* forking reason category, the three projects in the second row belong to the *"more community-driven development"* forking reason category, and the two projects in the third row belong to the *"personal differences"* forking reason category.

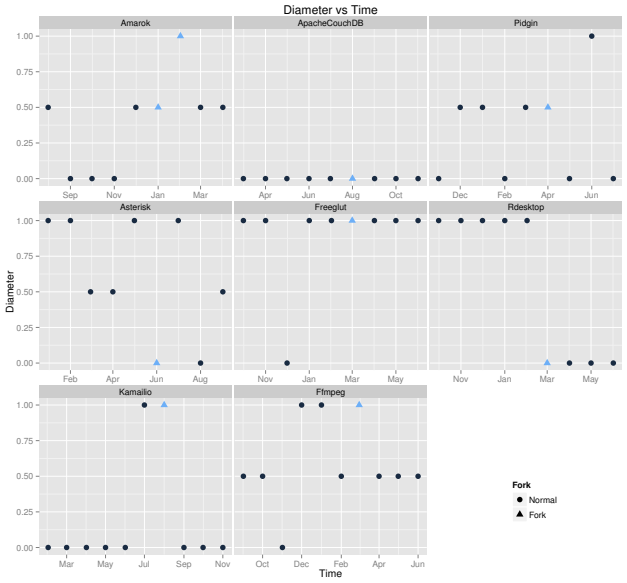


Figure 4: Diameter changes over Time. Note that the diameter measurements were normalized to the range [0,1]

diameter: a graph whose diameter does not increase much on node or edge removal has higher resilience [12].

6.1 Visualization

Several visualization techniques and tools are used in the field of social network analysis, for instance, Gephi [8], which is a FLOSS tool for exploring and manipulating networks. It is capable of handling large networks with more than 20,000 nodes and features several SNA algorithms. We used it for dynamic network visualization. We visualized the dynamic network changes using Gephi [8]. The videos[1] show how the community graph is structured, using a continuous force-directed linear-linear model, in which the nodes are positioned near or far from each other proportional to the graph distance between them. This results in a graph shape between Früchterman & Rheingold's [18] layout and Noack's LinLog [32].

6.2 Initial study results and discussion

6.2.1 *Kamailio Project*

Figure 5 shows four key frames from the Kamailio project's social graph around the time of their fork (the events described here are easier to fully grasp by watching the video. A node's size in a proportional to the number of interactions the node (contributor) has had within the study period and the position and edges of the nodes change if they had interactions within the time window shown, with six day steps per frame. The 1 minute and 37 seconds video shows the life of the Kamailio project between October 2007, and March 2009. Nodes are colored based on the modularity of the network.

The community starts with the GeneralList as the the biggest node, and four larger core contributors and three lesser size core contributors. The big red-colored node's transitions are hard to miss, as this major contributor departs from the core to the periphery of the network (Video minute 1:02) and then leaves the community (Video minute 1:24) capturing either a conflict or retirement. This corresponds to the personal difference category of forking reasons.

Figure 6 shows the betweenness centrality of the major contributors of Kamailio project over the same time period. The horizontal axis marks the dates, (each mark represents a 3-month time window with 1.5 months overlap). The vertical axis shows the percentage of the top betweenness centralities for each node. The saliency of the GeneralList – colored as light blue – is apparent due to to its continuous and dominant presence in the stacked area chart. The chart legend lists the contributors based on the color and in the same order of appearance on the chart starting from the bottom. One can easily see that around the "Aug. 15, 2008 - Nov. 15, 2008" tick mark on the horizontal axis, several contributors' betweenness centralities shrink to almost zero and disappear. This helps identify the date of fork with a month accuracy. The network diameter of the Kamailio project over the same time period is also shown in Figure 6. An increase in the network diameter during this period as suggested by findings of Hannemann and Klamma [20] is seen.

This technique can be used to identify the people involved

[1] Video visualizations available at http://eecs.oregonstate.edu/~azarbaam/OSS2014/

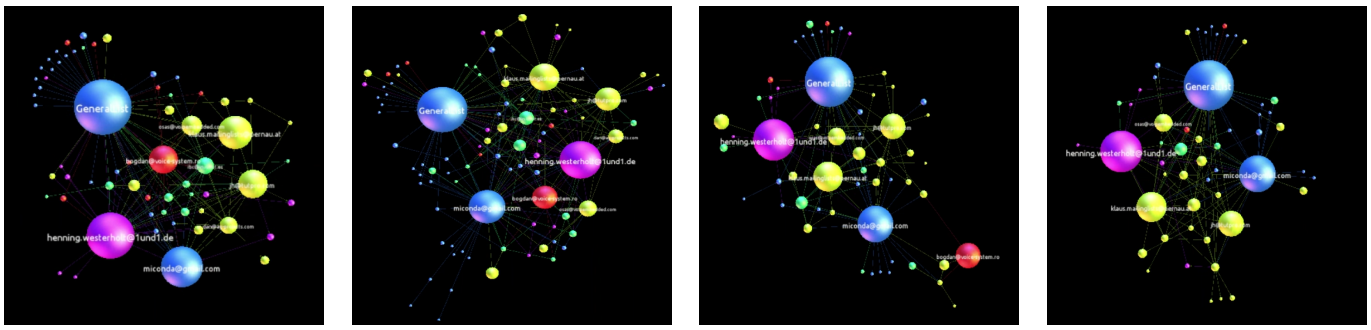


Figure 5: Snapshots from video visualization of Kamailio's graph (Oct. 2007 - Mar. 2009) in which a core contributor (colored red) moves to the periphery and eventually departs the community.

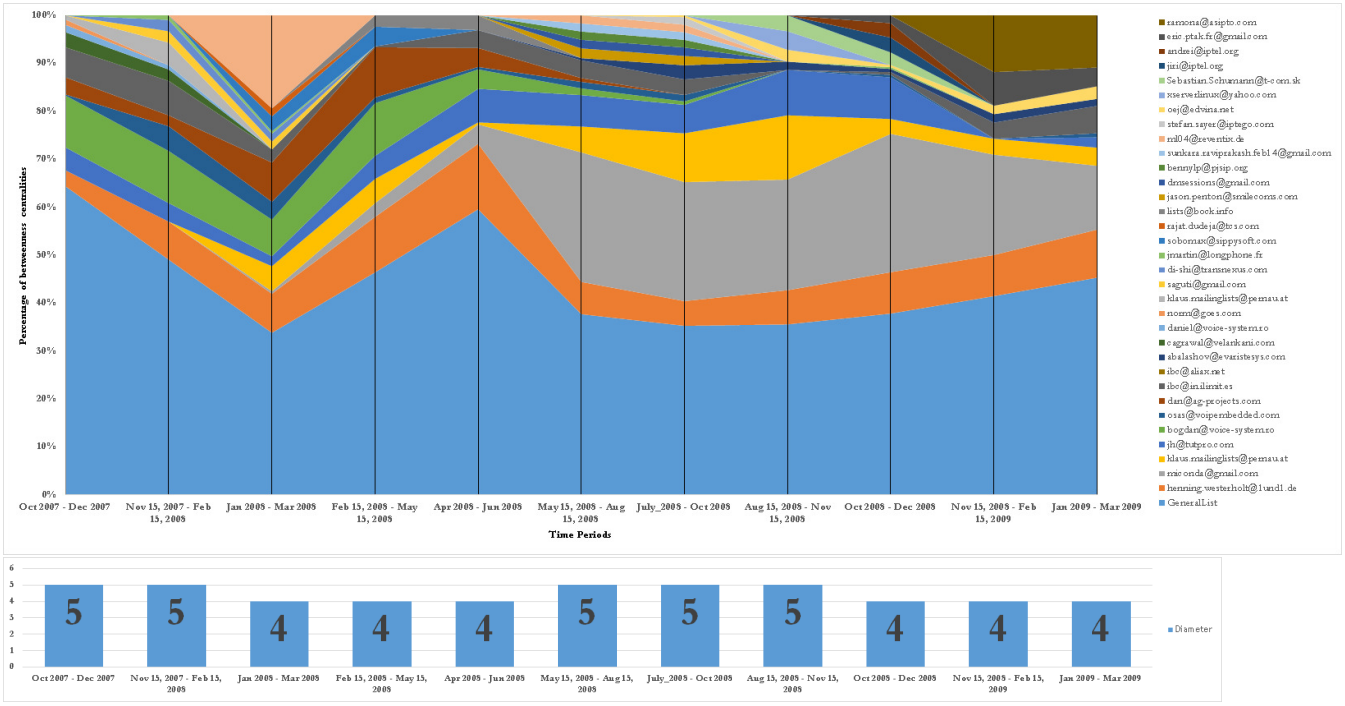


Figure 6: Kamailio top contributors' betweenness centralities and network diameter over time (Oct. 2007 to Mar. 2009) in 3-month time windows with 1.5-month overlaps

in conflict and the date the fork happened with a months accuracy, even if the rival project does not emerge immediately.

6.2.2 *Amarok Project*

The video for the Amarok project fork is available online[2], and the results from our quantitative analysis of the betweenness centralities and the network diameters are shown in Figure 7. The results show that the network diameter has not increased over the period of the fork, which shows a resilient network. The video shows the dynamic changes in the network structure, again typical of a healthy network, rather than of simmering conflict. These indicators suggest that the Amarok fork in 2010 belongs to the "addition of technical functionality" rationale for forking, as there are no visible social conflict.

6.2.3 *Asterisk Project*

The video for the Asterisk project is also available online, and the results from our quantitative analysis of the betweenness centralities and the network diameters are shown in Figure 8. The results show that the network diameter remained steady at 6 throughout the period. The Asterisk community was by far the most crowded project, with 932 nodes and 4282 edges. The stacked area chart shows the distribution of centralities, where we see an 80%-20% distribution (i.e. 80% or more of the activity is attributed to six major players, with the rest of the community ac-

[2]Video visualizations available at http://eecs.oregonstate.edu/~azarbaam/OSS2014/

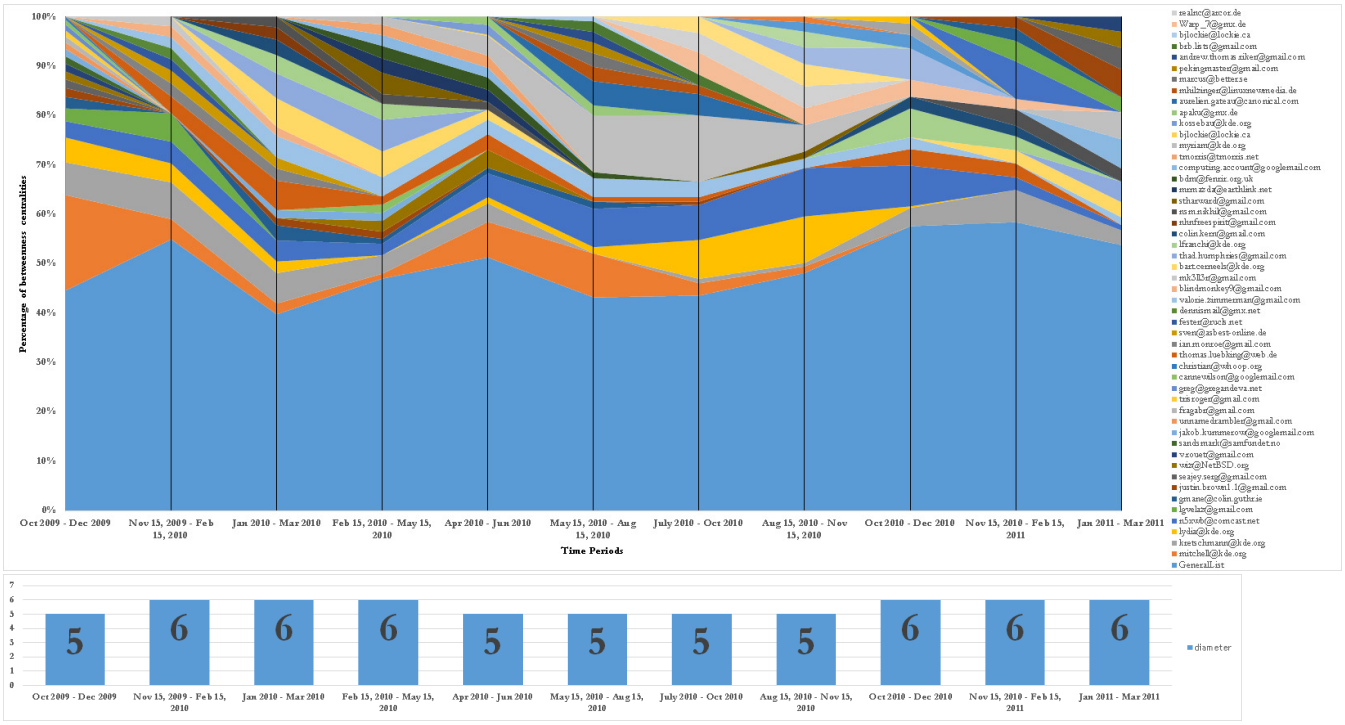


Figure 7: Amarok project's top contributors' betweenness centralities and network diameter over time between Oct. 2009 to Mar. 2011 in 3-months time windows with 1.5 months overlaps

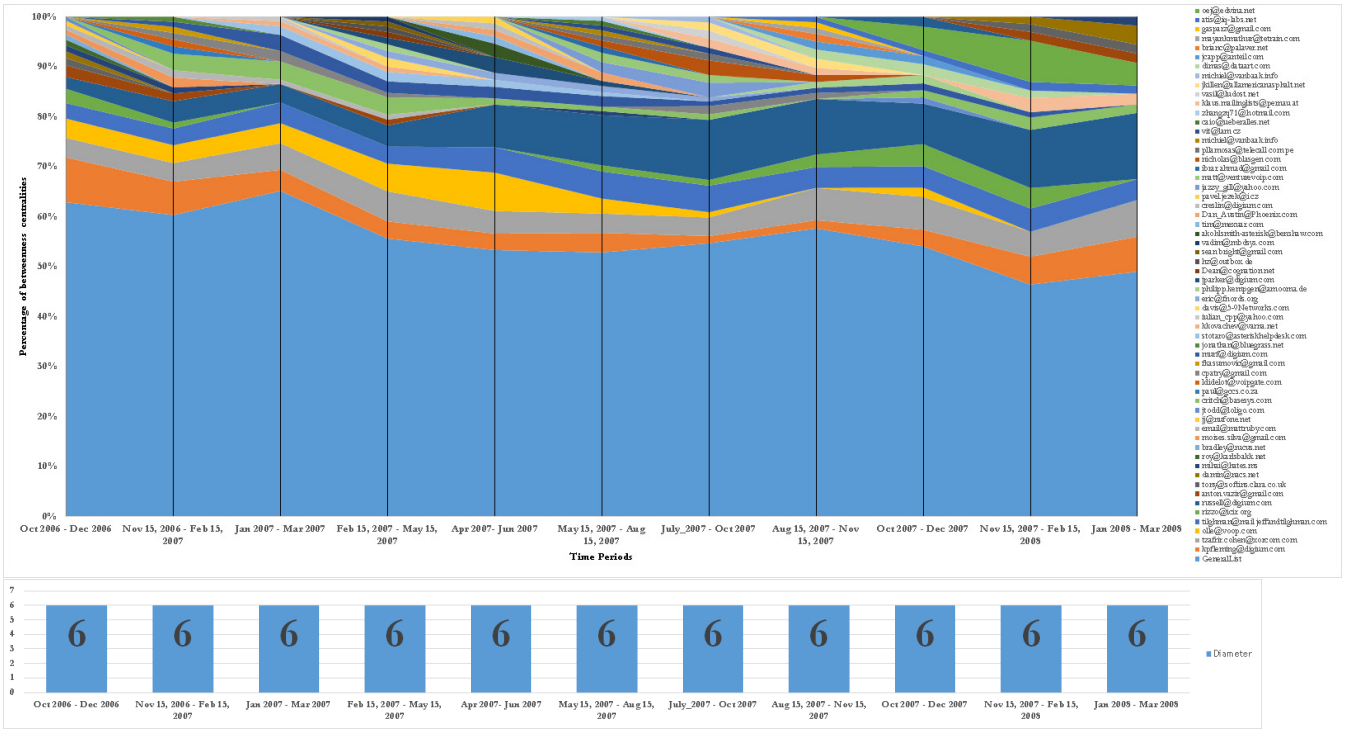


Figure 8: Asterisk project's top contributors' betweenness centralities and network diameter over time between Oct. 2009 to Mar. 2011 in 3-months time windows with 1.5 months overlaps

counting for only 20%). This is evident in the video representation as well, as the top-level structure of the network holds throughout the time period. The results from the visual and quantitative analysis links the Asterisk fork to the more community-driven category of forking reasons.

6.2.4 *Initial study conclusion*

We studied the collaboration networks of three FOSS projects using a combination of temporal visualization and quantitative analysis. We based our study on two papers by Robles and Gonzalez-Barahona [40] and Hannemann and Klamma [20], and identified three projects that had forked in the recent past. We mined the collaboration data, formed dynamic collaboration graphs, and measured social network metrics over an 18-month period time window.

We also visualized the dynamic graphs (available online) and as stacked area charts over time. The visualizations and the quantitative results showed the differences among the projects in the three forking reasons of personal differences among the developer teams, technical differences (addition of new functionality) and more community-driven development. The personal differences representative project was identifiable, and so was the date it forked, with a month accuracy. The novelty of the approach was in applying the network-specific *temporal* analysis rather than static analysis, and in the temporal visualization of community structure. We showed that this approach shed light on the structure of these projects and reveal information that cannot be seen otherwise.

More importantly, the initial study showed the limitations of a *network-specific* approach, and hence, we need to adopt a *population-processes* approach for our proposed main study.

7. CONCLUSION

Software development in free and open source software development are prime examples of how communities can be leveraged to create and maintain software. Forking, either as an undesirable violent split, or a non-undesirable friendly divide, affects the community. We propose to examine several software development projects that have forked over the past decade, and analyze the evolution and change of dynamics in the developers' interaction patterns. This may answer several open questions, particularly, about the run-up to forks, which is seldom studied.

We proposed to model the developers interactions statistically, to find the population processes that underlie the formation, changes, and dissolution of developer communities. We expect to see distinct changes of such processes for each forking category; i.e. 1) Undesirable forks because of personal conflicts among the developer team, 2) Undesirable forks because of the need for more community-driven development, when a few elite developers behave in a clannish manner and put a glass ceiling for the rest of the community, 3) Other non-undesirable (e.g. Healthy) forks, in which, a fraction of developers temporarily fork away to add new features, and plan to merge back to the main project. We will also look at stable projects that have not forked undesirably, as control group.

We expect to find patterns specific to each of the above categories, which then may be used to identify early warning signs of forking. The identification of such measures may inform those who are interested in the sustainability of their project community to stay informed and take action to amend undesirable dynamics.

Table 10 Timeline

Spring 2012	Literature review	
Fall 2012	Literature review & data collection	
Fall 2013	Data cleaning and wrangling	
Winter 2014	Creating communication graphs	
Spring 2014	Temporal visualization and temporal SNA	
Fall 2014	Preliminary statistical analysis	
Spring 2015	Planning and preliminary examination	
Summer 2015	Statistical analysis	RQ 4.1.1-3
Fall 2015	Interviews designs (including pilots)	
Winter 2016	Interviews	RQ 4.1.4
Spring 2016	Thesis writing	
June 2016	Defense	

8. TIMELINE

Table 10 outlines my proposed research timeline. It also shows the mapping between research questions and the phases of the study.

9. THREATS TO VALIDITY

The presented findings may not be generalized to all OSS projects. The projects studies in this paper were selected from a pool of candidate projects, partly because of the availability of their data. Given access, a better sampling approach could result in a more robust investigation. Furthermore, the proposed technique uses the data from online communications. The assumption that all the communication can be captured by mining repositories is intuitively imperfect, but inevitable. The qualitative approach of interviewing key individuals from the community also reflects what they remember about their perception at the time, which should be taken with a grain of salt, as many of the projects forked several years ago. And, also perception of individuals cannot be taken as granted ground truth.

Acknowledgement

The author would like to thank his academic adviser, Prof. Carlos Jensen, and his committee members, Prof. Margaret Burnett, Prof. Ronald Metoyer, and Prof. Christopher Scaffidi for their help and guidance throughout the author's PhD program, and particularly, in help with framing the proposed research in this paper. The author would also like to thank Derric Jacobs, of Sociology Department at Oregon State University, for his help with search for theories and lending books to the author; and also, many thanks goes to Prof. Drew Gerkey, of Department of Anthropology at Oregon State University for his advice and pointers to theories for human behavior in social sciences. The author would also like to thank the open source developers of the projects

studied for making their data available, without which this study would not have been possible.

10. REFERENCES

[1] Asur, S., S. Parthasarathy, and D. Ucar, (2009), "*An event-based framework for characterizing the evolutionary behavior of interaction graphs,*" ACM Trans. Knowledge Discovery Data. 3, 4, Article 16, (November 2009), 36 pages. 2009.

[2] Azarbakht, A. and C. Jensen, "*Drawing the Big Picture: Temporal Visualization of Dynamic Collaboration Graphs of OSS Software Forks,*" Proc. 10th Int'l. Conf. Open Source Systems, 2014.

[3] Azarbakht, A. and C. Jensen, "*Temporal Visualization of Dynamic Collaboration Graphs of OSS Software Forks,*" Proc. Int'l. Network for Social Network Analysis (INSNA) Sunbelt XXXIV Conf., 2014.

[4] Azarbakht, A., "*Drawing the Big Picture: Analyzing FLOSS Collaboration with Temporal Social Network Analysis,*" Proc. 9th Int'l. Symp. Open Collaboration, ACM, 2013.

[5] Azarbakht, A. and C. Jensen, "*Analyzing FOSS Collaboration & Social Dynamics with Temporal Social Networks,*" Proc. 9th Int'l. Conf. Open Source Systems Doct. Cons., 2013.

[6] Azarbakht, A., "*Temporal Visualization of Collaborative Software Development in FOSS Forks,*" Proc. IEEE Symp. Visual Languages and Human-Centric Computing, 2014.

[7] Baishakhi R., C. Wiley, and M. Kim, "*REPERTOIRE: a cross-system porting analysis tool for forked software projects,*" Proc. ACM SIGSOFT 20th Int'l. Symp. Foundations of Software Engineering, ACM, 2012.

[8] Bastian, M., S. Heymann, and M. Jacomy, "*Gephi: an open source software for exploring and manipulating networks,*" Int'l AAAI Conf. on Weblogs and Social Media, 2009.

[9] Bezrukova, K,, C. S. Spell, J. L. Perry, "*Violent Splits Or Healthy Divides? Coping With Injustice Through Faultlines,*" Personnel Psychology, Vol 63, Issue 3. 2010.

[10] Bird, C., D. Pattison, R. D'Souza, V. Filkov, and P. Devanbu, "*Latent social structure in open source projects,*" Proc. 16th ACM SIGSOFT Int'l. Symposium on Foundations of software engineering, ACM, 2008.

[11] Brandes, U. "*A Faster Algorithm for Betweenness Centrality*", Journal of Mathematical Sociology 25(2):163-177, 2001.

[12] Chakrabarti, D., and C. Faloutsos. "*Graph mining: Laws, generators, and algorithms,*" ACM Computing Surveys, 38, 1, Article 2, 2006.

[13] Crowston, K., K. Wei, J. Howison, and A. Wiggins. "*Free/Libre open-source software development: What we know and what we do not know,*" ACM Computing Surveys, 44, 2, Article 7, 2012.

[14] Davidson, J, R. Naik, A. Mannan, A. Azarbakht, C. Jensen, "*On older adults in free/open source software: reflections of contributors and community leaders,*" Proc. IEEE Symp. Visual Languages and Human-Centric Computing, 2014.

[15] Ernst, N., S. Easterbrook, and J. Mylopoulos, "*Code forking in open-source software: a requirements perspective,*" arXiv preprint arXiv:1004.2889, 2010.

[16] Ford, L. R. and D. R. Folkerson, "*A simple algorithm for finding maximal network flows and an application to the Hitchcock problem,*" Canadian Journal of Mathematics, vol. 9, pp. 210-218, 1957.

[17] Forrest, D., C. Jensen, N. Mohan, and J. Davidson, "*Exploring the Role of Outside Organizations in Free/ Open Source Software Projects,*" Proc. 8th Int'l. Conf. Open Source Systems, 2012.

[18] Fruchterman, T. M. J. and E. M. Reingold, "*Graph drawing by force-directed placement,*" Softw: Pract. Exper., vol. 21, no. 11, pp. 1129-1164, 1991.

[19] Guzzi, A., A. Bacchelli, M. Lanza, M. Pinzger, and A. van Deursen. "*Communication in open source software development mailing lists,*" Proc. 10th Conf. on Mining Software Repositories, IEEE Press, 2013.

[20] Hannemann, A and , R. Klamma "*Community Dynamics in Open Source Software Projects: Aging and Social Reshaping,*" Proc. Int. Conf. on Open Source Systems, 2013.

[21] Heider, F. The Psychology of Interpersonal Relations. John Wiley & Sons. 1958.

[22] Howison, J. and K. Crowston. "*The perils and pitfalls of mining SourceForge,*" Proc. Int'l. Workshop on Mining Software Repositories, 2004.

[23] Howison, J., K. Inoue, and K. Crowston, "*Social dynamics of free and open source team communications,*" Proc. Int'l. Conf. Open Source Systems, 2006.

[24] Howison, J., M. Conklin, and K. Crowston, "*FLOSSmole: A collaborative repository for FLOSS research data and analyses,*" Int'l. Journal of Information Technology and Web Engineering, 1(3), 17-26. 2006.

[25] Krivitsky, P. N., and M. S. Handcock. "*A separable model for dynamic networks,*" Journal of the Royal Statistical Society: Series B (Statistical Methodology) 76, no. 1: 29-46. 2014.

[26] Kuechler, V., C. Gilbertson, and C. Jensen, "*Gender Differences in Early Free and Open Source Software Joining Process,*" Open Source Systems: Long-Term Sustainability, 2012.

[27] Kunegis, J., S. Sizov, F. Schwagereit, and D. Fay, "*Diversity dynamics in online networks,*" Proc. 23rd ACM Conf. on Hypertext and Social Media, 2012.

[28] Leskovec, J., Kleinberg, J., and Faloutsos, C.: "*Graphs over time: densification laws, shrinking diameters and possible explanations,*" Proc. SIGKDD Int'l. Conf. Knowledge Discovery and data Mining, 2005.

[29] Leskovec, J., K. J. Lang, A. Dasgupta, and M. W. Mahoney, "*Statistical properties of community structure in large social and information networks,*" Proc. 17th Int'l. Conf. World Wide Web, ACM, 2008.

[30] Nakakoji, K., Y. Yamamoto, Y. Nishinaka, K. Kishida, and Y. Ye. "*Evolution patterns of open-source software systems and communities,*" Proc. Int'l. Workshop Principles of Software Evolution,

ACM, 2002.

[31] Mikkonen, T., L. Nyman, "*To Fork or Not to Fork: Fork Motivations in SourceForge Projects,*" Int'l. J. Open Source Softw. Process. 3, 3. July, 2011.

[32] Noack, A., "*Energy models for graph clustering,*" J. Graph Algorithms Appl., vol. 11, no. 2, pp. 453-480, 2007.

[33] Nowak, M. A. "*Five rules for the evolution of cooperation,*" Science 314, No. 5805: 1560-1563. 2006.

[34] Nyman, L. , "*Understanding code forking in open source software,*" Proc. 7th Int'l. Conf. Open Source Systems Doct. Cons., 2011.

[35] Nyman, L., T. Mikkonen, J. Lindman, and M. Fougère, "*Forking: the invisible hand of sustainability in open source software,*" Proc. SOS 2011: Towards Sustainable Open Source, 2011.

[36] Nyman, L., "*Hackers on Forking,*" Proc. Int'l. Symp. on Open Collaboration, 2014.

[37] Oh, W., Jeon, S., "*Membership Dynamics and Network Stability in the Open-Source Community: The Ising Perspective*" Proc. 25th Int'l. Conf. Information Systems. 2004.

[38] Page, B, B. Sergey, R. Motwani and T. Winograd, "*The PageRank Citation Ranking: Bringing Order to the Web,*" Technical Report, Stanford InfoLab, 1999.

[39] Robins, G., P. Pattison, Y. Kalish, and D. Lusher. "*An introduction to exponential random graph (p*) models for social networks,*" Social networks 29, no. 2: 173-191. 2007.

[40] Robles, G. and J. M. Gonzalez-Barahona, "*A comprehensive study of software forks: Dates, reasons and outcomes,*" Proc. 8th Int'l. Conf. Open Source Systems, 2012.

[41] Rocchini, C. (Nov. 27 2012), Wikimedia Commons, Available: http://en.wikipedia.org/wiki/File:Centrality.svg, 2012.

[42] Singer, L., F. Figueira Filho, B. Cleary, C. Treude, M. Storey, and K. Schneider. "*Mutual assessment in the social programmer ecosystem: an empirical investigation of developer profile aggregators,*" Proc. Conf. Computer supported cooperative work, ACM, 2013.

[43] Snijders, T. AB. "*Markov chain Monte Carlo estimation of exponential random graph models,*" Journal of Social Structure 3, no. 2: 1-40. 2002.

[44] Sowe, S., L. Stamelos, and L. Angelis, "*Identifying knowledge brokers that yield software engineering knowledge in OSS projects,*" Information and Software Technology, vol. 48, pp. 1025-1033, Nov 2006.

[45] Spence, M. "*Job market signaling,*" Quarterly Journal of Economics, 87: 355-374. 1973.

[46] Steglich, C., T. AB Snijders, and M. Pearson. "*Dynamic networks and behavior: Separating selection from influence,*" Sociological methodology 40, no. 1: 329-393. 2010.

[47] Storey, M., L. Singer, B. Cleary, F. Figueira Filho, and A. Zagalsky, "*The (R) Evolution of social media in software engineering,*" Proc. Future of Software Engineering, ACM, 2014.

[48] Syeed, M. M., "*Socio-Technical Dependencies in Forked OSS Projects: Evidence from the BSD Family,*" Journal of Software 9.11 (2014): 2895-2909. 2014.

[49] Teixeira, J., and T. Lin, "*Collaboration in the open-source arena: the webkit case,*" Proc. 52nd ACM conf. Computers and people research (SIGSIM-CPR '14). ACM, 2014.

[50] Torres, M. R. M., S. L. Toral, M. Perales, and F. Barrero, "*Analysis of the Core Team Role in Open Source Communities,*" Int. Conf. on Complex, Intelligent and Software Intensive Systems, IEEE, 2011.

[51] Zachary, W., "*An information flow model for conflict and fission in small groups,*" Journal of Anthropological Research, vol. 33, no. 4, pp. 452-473, 1977.