View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by AIS Electronic Library (AISeL)

Computational Approaches for Analyzing Latent Social Structures in Open Source Organizing[1]

Completed Research Paper

Aron Lindberg
Case Western Reserve University
Cleveland, OH
aron.lindberg@case.edu

Nicholas Berente
University of Georgia
Athens, GA
berente@uga.edu

James Gaskin
Brigham Young University
Provo, UT
james.eric.gaskin@gmail.com

Kalle Lyytinen
Case Western Reserve University
Cleveland, OH
kalle.lyytinen@case.edu

Youngjin Yoo
Temple University
Philadelphia, PA
youngjin.yoo@temple.edu

Abstract

Open source software represents a novel form of organizing that leaves digital trace data for organizational researchers to analyze using computational methods. Computational social science has emerged as an important approach to understanding patterns that represent latent social structures in sociological, organizational, and technical phenomena. Within the context of open and digitalized collaboration the clearest manifestation of computational social science has been social network analysis. While social network analysis is a powerful approach for understanding social phenomena in terms of their latent relational social structure, the network lens does not capture the entirety of social structures. Procedural social structures undergirding recurrent patterns of action form another important element of latent social structure. Analyzing such structures requires alternative methods able to deal with history-dependent patterning of activities. Therefore, we investigate the concepts of latent relational and procedural structures, and discuss computational approaches for analyzing patterns and interdependencies among such structures.

Keywords: Open Source Software, Social Structure, Procedural Structure, Relational Structure, Sequence Analysis, Social Network Analysis, Digital Collaboration

[1] This paper is based on work supported, in part, by the National Science Foundation under grants VOSS 1121935 and VOSS 1120966. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Introduction

The Open Source Software (OSS) movement represents a distinct, new form of organizing (Von Hippel and Von Krogh 2003) and can be regarded a harbinger of the sorts of organizations that lie in the future as digitally-enabled collaboration expands across space, time, and organizational boundaries. A good deal of recent research has investigated structures of open source software development (Crowston et al. 2012) and how they differ from familiar forms of organizing. Although for two decades scholars have speculated about the "coming of new organizations" – forms that are fluid, informal and dynamic (Drucker 1988) – rigorous characterizations of the exact nature of this coming have been elusive. Because of its open and strictly digital character, open source organizing offers one domain to examine emergent forms. At the same time, there are no blueprints available to study such forms since most structures of such organizations have to be discovered from observed patterns of action and associated technological affordances. Fortunately, there is a wealth of trace data that can be used to reveal patterns of action in order to detect the underlying structures of open source organizing (Howison et al. 2011). This, however, requires development of new computational methods that can analyze the abundance of trace data and come to grips with latent structures associated with this form of organizing.[2]

In most cases Social Network Analysis (SNA) has been the dominant computational method for the discovery of latent social structures (Lazer et al. 2009). However, the type of latent structures identified through social network analysis does not make up the entirety of social structures at play in open source organizing. The structures revealed by SNA involve latent *relational structures* that are concerned with the regularities of who interacts with whom (Wasserman and Faust 1994). Such relational structures are different from, and complementary to *procedural structures* which are concerned with who does what, how, and in what order (Pentland and Rueter 1994). All human action takes place in historical, cultural, and institutional contexts where rules for appropriate action are inherited and continually enacted over time (Powell and DiMaggio 1991). Contemporary structurational thinkers emphasize that existing patterns of action and underlying institutional scripts act as social structures that guide human behavior while being enacted (and revealed) through human action (Bourdieu 1977; Giddens 1984). In organizational scholarship, a number of scholars have recently taken this insight further by exploring procedural structures through the means of sequence analytic techniques (Abbott 1992; Feldman and Pentland 2003; Gaskin et al. forthcoming). This has increased our understanding of how organizational routines operate as social structure and can complement SNA techniques to provide a more comprehensive picture of latent social structures present in open source organizing. Although there are a number of recent studies seeking to integrate various computational approaches addressing both relational and procedural social structure (e.g., Butts 2008; Brandes et al. 2009), a comprehensive mixed methods integration of the two, as well as with qualitative data is still a long way off.

The goals of this paper are both theoretical and methodological. Theoretically we explore how information systems and organizational scholars can theoretically attend to both latent relational and procedural structures in their theorizing about social structures. Methodologically, we discuss how the complementary combination of SNA and sequence analytic techniques can advance the empirical inquiry into open source organizing (Crowston et al. 2012). The remainder of the manuscript is organized as follows. We next trace the broad evolution of conceptualizations of social structure and describe how attention to both latent relational and procedural elements of social structure can help us analyze so far hidden elements of open source organizing. Then we address computational methods associated with studies that heed into relational and procedural latent structure: social network and sequence analysis, respectively. We conclude with an example and discussion of how researchers can combine relational with procedural approaches to theorize about social structures in the context of open source organizing.

[2] By latent structures we are referring to those structures that are not readily accessible to process participants or researchers. Latent structures are precisely what researchers aim to uncover with pattern identification associated with computational analysis of trace data. Merton (1957) contrasted latent phenomena with "manifest" phenomena – which, in the context of social structure, describes the planned, explicit, and readily observable elements of structure.

Social Structure in Organizations

Although various notions of "structure" have pervaded social theories since the seminal works of Durkheim, Marx, and Weber, it is specifically in the functionalist view of Parsons and Merton where the contemporary usage of the term "social structure" gained its footing (Giddens 1979). Both Parsons (1960) and Merton (1957) conceived of human activity in a context of nested social systems. According to Parsons, a social system is a "collectivity" of humans interacting with each other in particular roles – and these roles are fundamental to the collectivity (examples of collectivities include organizations and societies). He distinguished between collectivities and institutions. The collectivities comprise the elements of the system, whereas institutions are "generalized patterns of norms which define categories of prescribed, permitted and prohibited behavior in social relationships" (Parsons 1960: 177). Institutions embody the cultural goals, values, and prescriptions (scripts) for appropriate action (Merton 1957; Parsons 1960) and stand as regulations for the "allowable procedures" in the system (Merton 1957: 133). As such, institutions go hand-in-hand with collectivities and define appropriate sequences of activities for the various roles in the system for situations that the system might face. Thus, taken together, according to a functionalist view, social systems have two elements: (1) the *relations* between elements of the system - formal roles of people and workflow-based interactions, and (2) the *procedural* elements of the system – defined by goals, values, and institutions associated with proper course of action in the context of the system. This fundamental distinction between the two structuring aspects of a social system - its relational structure and its rule based procedural structure - is one of the foundational distinctions in organizational scholarship.

Since early organizational scholarship, there has been an increasing realization that institutions are not somehow enclosed within particular organizational contexts, but instead transcend organizational activity in both time and space (Giddens 1979). Institutions are "the enduring features of social life" (Giddens 1984: 24). Institutional norms, values, scripts, and related typifications exist outside, are prior to, and result from organizational activity while being shared across social systems and fields (DiMaggio and Powell 1983). Organizational actors therefore draw upon a broader institutional order for the rules and resources that they continually reproduce (Giddens 1984). Procedural structures transcend any one organization and are therefore, although related to relational structures, fundamentally different. Hence, we distinguish between two forms of structure that jointly comprise social structure: the relational structure which involves the structural elements of a system and their relations to each other; and the procedural structure which involves the rule/habit based patterns of activity in a system.

In addition to recognizing differing forms of structure, it is important to note that each can involve either manifest or latent dimensions. Merton (1957) indicated that systems have "manifest" functions – those that are conscious and deliberate. He also described a dysfunctional subtext within systems that he named "latent" functions – those that are unconscious and unplanned (see Table 1). Thus, organizational structures may be manifest or latent. Manifest structures are made explicit by being planned out, and documented and thereby made available to concerned actors. For relational social structures, this could involve the formal organizational structure (Mintzberg 1979), or interorganizational arrangements such as alliances (Powell 1990). For procedural structures, the manifest dimension involves explicit written norms and rules expressed in ostensive aspects of organizational routines such as specifications of business processes (Pentland and Rueter 1994), or related institutionalized scripts for appropriate action, e.g. preferred ways of hiring faculty (DiMaggio and Powell 1991). Latent structures, on the other hand, are not readily available for either organizational participants or for researchers and represent the informal, organic, and unplanned structures that emerge during action. They are not necessarily dysfunctional, as Merton suggested, but in contrast, they often benefit organizations in specific circumstances (Blau and Scott 1962). For relational social structures, latent forms include informal interpersonal networks in which organizational actors are embedded (Granovetter 1985); latent procedural structures involve performative or habit-based behavior - the enactment of "actual" organizational routines (Feldman and Pentland 2003) rife with improvisation and bricolage which all can be described in terms of enacted institutional practice (Thornton et al. 2012).

The manifest forms of structure are generally accessible through classic methods of social inquiry since they are generally well represented. Latent structures are challenging since they are not readily accessible and cannot easily be made explicit to any social actor observing the social system. To reveal them we need trace data to find out who interacts with whom, when who did what, and then based on such observations determine recurrent patterns. In both tasks we need ample computational resources to aid the researcher in recording traces and identifying patterns and revealing the enacted latent structure. Emerging computational social science is therefore generally concerned with uncovering the latent patterns of human activity from the recorded trace data. Before digitalization obtaining such data was extremely difficult or time consuming. Now, with extensive digitalization of work and interactions, such data has become more abundant (Lazer et al. 2009). Next we will briefly address each form of latent structure and describe how it specifically relates to open source organizing.

Table 1. Four Dimensions of Organizational Social Structure

	Definition	**Example Units of Analysis**	**Example Citations**
Relational Structure	***Manifest Relational Structure:*** Arrangements for human interaction.	Organizational chart and inter-organizational forms	Mintzberg 1979; Powell 1990
	Latent Relational Structure: Regularities of human interaction	Informal social networks	Granovetter 1985; Burt 2000
Procedural Structure	***Manifest Procedural Structure:*** Prescriptions for human action	Written ostensive organizational routines / institutional scripts	Nelson and Winter 1982; DiMaggio and Powell 1991
	Latent Procedural Structure: Repeatedly enacted rules and resources for human action	Performative organizational routines / institutional practices	Feldman and Pentland 2003; Thornton et al 2012

Latent Relational Structure & Open Source Organizing

In the 1950s and 1960s, a number of organizational scholars noted that analyses of phenomena using categories of groups or classes, were seriously confounded by overlaps and interconnections with what at first had seemed to be easily separable groups of individuals (Wellman and Berkowitz 1988). Certain "rebellious" scholars searched for alternative, empirically-driven perspectives for capturing the latent relations between individuals without the need for theorizing groups ex ante – groups that may not be as relevant as underlying social bonds between individuals (Granovetter 1990).

Table 2. Key Aspects of Latent Relational Social Structure in Open Source Organizing

	Definition / Description	**Original citations**	**Findings in open source organizing**	**OSS citations**
Power-law Distributions	Network structures follows a highly skewed power-law distribution	Barabási and Reka 1999	Projects' size, intra-project contributions, & communication patterns follow power law distributions	Madey et al. 2002; Von Krogh et al. 2003; Crowston and Scozzi 2004; Moon and Sproull 2000
Transitivity	Local clusters tend to form in networks	Watts and Strogatz 1998	OSS tend to exhibit a core/periphery structure which varies considerable between projects	Crowston et al. 2006; Dinh-Trong and Bieman 2005; Healy and Schussman 2003; Koch and Schneider 2002; Mateos-Garcia and Steinmueller 2008; Valverde and Solé 2007

Embeddedness	Highly connected nodes tend to exhibit superior performance	Granovetter 1985	Projects and individuals that are tightly embedded in network structures exhibit higher performance	Grewal et al. 2006; Oh and Jeon 2007; Hahn et al. 2008; Shen and Monge 2011; Conaldi et al. 2012

The relational view of social structure was born; its conception of interaction relationships as the foundational unit of social structure began to take form with a somewhat alternative set of assumptions compared to the mainstream social research. Rather than looking at individual actors and their characteristics, this wave of research analyzed relations among the actors as the unit of analysis (Coleman 1958) and created a decisive break from the current sociological and socio-psychological research. It essentially argued that individual's behaviors are determined by context embedding and not by individual characteristics. The core concepts associated with relational structure involve thinking of organizations in terms of networks of nodes that have ties of some form with each other (Monge and Contractor 2003). A node can be any actor (even non-human actors), and a tie can be any type of interaction or association between actors including forms of communication or social association. Ties can have a variety of characteristics including strength, direction, stability, etc. (Monge and Contractor 2003). Using this basic theoretical lexicon, relational theorists have found several common patterns that characterize organizational networks. These include power-law degree of distributions, network transitivity and embeddedness (Table 2).

Power-law Degree Distributions: Patterns of associations among people depart significantly from random patterns. The tendencies of people to form ties with some people more readily than others were described by *power-law degree distributions* (Barabási and Reka 1999). This means that the number of ties that the individuals have in a population are distributed according to a power-law. Most have relatively few ties, but as we move to some type of center of the network, the number of ties per individual increases dramatically.

Power-law distributions are common in social networks in general, and so is the case with OSS networks. OSS projects are distributed in terms of power-laws both with regard to the size of their codebase, as well as the number of developers involved (Madey et al. 2002). The very first accounts of OSS, from the practitioner literature, pointed out the importance to the OSS model of large numbers of developers who provide "enough eyeballs" to make all bugs "shallow", i.e. solvable (Raymond 2001). Often, the necessary numbers required to create such massive communities have been elusive (Krishnamurthy 2002; Howison and Crowston, forthcoming), with the exception of major projects such as Linux (Moon and Sproull 2000), Apache, and Mozilla (Mockus et al. 2002). Hence, it seems as if most projects are small, driven by individual developers, perhaps with a smaller community reporting bugs and requesting features. Such power-law distributions can also be discerned in communication flows (von Krogh et al. 2003), code contributions (Moon and Sproull 2000), and fixing bugs (Crowston and Scozzi 2004). The process through which such power-laws seem to develop is preferential attachment (Madey et al. 2002) or herd behavior (Oh and Jeon 2007), driven by the desires of OSS developers to work on the most interesting and popular projects, features, and bugs (Conaldi et al. 2012).

Transitivity: Networks tend to create local clusters, not always with a single center. These clusters are described by the property of *network transitivity* (Watts and Strogatz 1998). Essentially this means that if a node has ties to two other nodes, these two other nodes have a high probability of being connected. A generic 80/20 pattern in the balance between core and periphery has been argued to be common in OSS (Mockus et al. 2002), and while this pattern can be seen in many projects, there is considerable variance in the exact distributions of activity between core and periphery in various projects (Crowston et al. 2006; Dinh-Trong and Bieman 2005; Healy and Schussman 2003; Koch and Schneider 2002; Mateos-Garcia and Steinmueller 2008; Valverde and Solé 2007). Based on Raymond's prescriptive account, it can be expected that such variance in centralization of OSS communities covaries with various forms of performance. Some of recent accounts have established initial predictions to that effect, but these studies remain far from conclusive. Exploring the effects of being embedded in certain structures can help us understand the links between network structures and performance.

Embeddedness: SNA represents a middle ground between over-socialized (overly deterministic structural explanations) and under-socialized accounts of social actions (atomistic and individualistic explanations). By looking at how actors are *embedded* in a network, we can see how various forms and magnitudes of embeddedness impacts social action and its outcomes (Granovetter 1985). The concept of embeddedness allows us to see how nuances in relational structures influence the different ways actors relate to a network. In the context of OSS, scholars have recognized that various form of embeddedness can be associated with performance. For example, Grewal et al. (2006) found that projects deeply embedded in networks of other projects generally saw a higher level of success, while the embeddedness of project managers had mixed effects on success. Further, the popularity of engaging with certain bugs has been shown to be highly dependent on the local network structures that such bugs are embedded in (Conaldi et al. 2012).These empirical findings are corroborated by strong evidence of herding effects in OSS (Oh and Jeon 2007). It seems that because of previous collaboration patterns (Hahn et al. 2008), and processes of strategic selection and homophily (Shen and Monge 2011), individuals tend to cluster to each other in ways that embed successful developers in contexts with other successful developers – thus promoting overall project success.

Latent Procedural Structure & Open Source Organizing

In an organizational context, latent social structure is encoded not only in networks, but also in routines that are continually reenacted (Nelson and Winter 1982). Such routines are on the one hand stable and persistent, but on the other hand they are also a source of flexibility and change (Dionysiou and Tsoukas 2013). Routines are the locus for a wealth of an organization's knowledge; within routines – and the meta-routines through which they change – rest also the dynamic capabilities that enable organizations to adapt (Eisenhardt and Martin 2000). As noted, one can distinguish between the ostensive (manifest) and performative (latent) elements of organizational routines (Feldman and Pentland 2003).

A particular tension in routines that has been explored extensively is that of between change and stability (Feldman 2003). Routines seem to be a source of both stability (their traditional goal), but also capable of generating significant amounts of contextual adaptation and improvisation (Feldman and Pentland 2003). We note three key elements in studying such patterns: 1) ordering, 2) rhythm, and 3) generative mechanisms (Table 3).

Table 3. Key Aspects of Latent Procedural Social Structure in Open Source Organizing

	Definition / Description	**Original citations**	**Findings in open source organizing**	**OSS citations**
Ordering	Specific ordering of events has contextual significance	Abbott and Hrycak 1990	Feature requests and bugs tend to initiate action sequences	Von Krogh et al. 2003; Christley and Madey 2007; Crowston and Scozzi 2004
Rhythm	Grooved and entrained sequences of events	Ancona and Chong 1992; Gersick and Hackman 1990	Growth in communities, contributions, and codebases are entrained to each other; growth is either constant or punctuated	Capiluppi 2004; Capiluppi et al. 2005; Robles et al. 2005; Koch 2005; Feller and Fitzgerald 2000
Generative Mechanisms	Processes that generate routines/grooved patterns (meta-routines)	Pentland 2005	Multiple forms of practices shape distinct lifecycle stages	Monteiro and Østerlie 2004; Capiluppi et al. 2007

Ordering: Some scholars have explored particular ways in which activities become staggered across time – their *ordering*. For example, Von Krogh et al. (2003) suggest that different projects have different "joining scripts" that specify a typical ordering of activity suitable for entering a certain development community. Such joining scripts typically start with joining a mailing list to get access to project communications, after which suggestions of bug fixes or new features supported by actual code are made. When such a script is followed, joining developers were more successful in being granted access to a developer community. Hence, institutions both common to the open source community at large as well as local to specific communities, tend to favor a certain ordering of activity as the preferred way of conducting software development. Christley and Madey (2007) find that distinct activity sequences seem to be initiated by forum postings, bug reports, or feature requests. Such a sequencing of micro-procedures provide us with a view of the generic structuring of procedures over time and thus portrays organizing not just as configuring relationships, but also as configuring activity elements staggered in time. It also mirrors the observations made by Raymond (2001) that the initiative towards continued development comes from the community periphery in the form of bug reports, feature requests, and suggested fixes. Crowston and Scozzi (2004) examined bug fixing procedures and suggested that while bug fixing lacks a formal coordination process, it still consists of a typical sequence of activities: a bug is submitted, then analyzed, fixed, and eventually closed since it is no longer a concern. While the ordering of such procedures is facilitated by technical functionalities in a version control system or a bug tracker, the ordering of activity itself also shows how a certain 'practical' logic expresses itself in the procedural structuring of OSS projects.

Rhythm: Another important concern has been to establish the overall *rhythm* through which project grows. In studies of OSS, this has often been expressed through attempts at establishing typical growth rates of various projects. It has been argued that the size of the codebase and the number of developers grow at similar rates (Capiluppi 2004), that most OSS projects grow at a linear rate, although there are exceptions, such as the Linux kernel which has grown at a super-linear rate (Robles et al. 2005), and others have been found to grow at a quadratic rate (Koch 2005).

The massively parallel nature of OSS development (Feller and Fitzgerald 2000) means that while activity levels at different times and across different components of a project may vary, they are likely to be associated with each other through processes of entrainment (Capiluppi 2004). Such grooving of activities create periods of distinct activity levels and types, often leading to punctuated equilibria – distinct periods of time were a project moves at a uniform rate. Hence, understanding the structure of procedures as it is expressed in somewhat stable patterns of recurring activities help us to understand the characteristics of activities in a certain period or project.

Generative Mechanisms: Finally, procedural structures are patterns of activities that make up the "life blood" (Monteiro and Østerlie 2004) of any OSS community. OSS activities unfold to fulfill specific organizational, ritual, and technical goals. Through these activity patterns, the community is maintained as the emergent codebase grows while existing ones are being pruned and refined. As such, understanding the *generative mechanisms* – underlying motors that continuously shape the emergent and manifest patterns - behind the unfolding of a particular development process is important for understanding how a particular community is created and sustained. The accounts of such mechanisms in the literature range from prescription (Raymond 2001) to a host of empirical accounts (e.g. Capiluppi et al. (2007)). Further, Van de Ven and Poole (1995) suggest several possible generative mechanisms: life cycle, evolution, dialectic and teleology. Lifecycle models suggests that occurrence of events are imminent. That is, earlier events already contain what is to come, which will manifest itself when certain conditions are met. It has been a common way of portraying software methodologies (Rajlich and Bennett 2000) and product evolution. The second generative mechanism, evolution, involves the recurring cycle of variation, selection and retention. It shows how the unfolding of events follows a non-deterministic probabilistic logic. The third generative mechanism, dialectic, suggests that the underlying engine that gets things moving is conflict and tension in pluralistic environments. The recurring cycle of thesis, antithesis, and synthesis is the basic underlying structure from a dialectic mechanism. Finally, with teleology as a generative mechanism, there is no prescribed recurring pattern as generated by the three other mechanisms. Instead, the process moves forward towards a goal, and the process takes on whatever form facilitates reaching said goal.

Social Network Analysis: a Computational Approach to Latent Relational Structures

The dominant computational approach to latent relational social structures is social network analysis (SNA). While SNA emerged from the theoretical concern of understanding relational structures, its growth was fueled by the methodological developments, mostly from mathematical approaches to graphs as an abstraction of relational structures formed by nodes and ties. In fields such as physics (Barabási and Reka 1999) and geography (Cressie 1993), abstract network concepts have been developed, that allows scholars to characterize generic features of networks. Fueled by such generic network science approaches, SNA grew to become a field mainly concerned with the analytical possibilities offered by new algorithms, as well as theorizing and calculating measures of co-variation between various topologies and other phenomena, such as diffusion of innovation, social capital, etc. This has led to a wealth of analytical tools to characterize abstract network structures of various kinds in different fields such as sociology, telecommunication, physics, and biology. From an abstract network perspective, it does not matter whether the nodes are individuals, proteins or routers. In organization studies, scholars often reduce these network structures into a set of variables to run regressions on other variables of interest, and thus provide a network based explanation of why certain phenomena, such as innovation, unfold in a certain way. However, initially the focus of SNA methods has been to analyze a static network for its structural characteristics.

Structural assumptions with regards to networks are not homogenous across studies. At least two separate ontological perspectives can be discerned: topological (structural) and connectionist (focusing on flows/ties) (Borgatti and Foster 2003). The former is the more traditional view – focusing on the overall structure of the network, or the structural position of an ego. The latter, focuses more on the flows between nodes, and have often been used as a foundation for studies on contagion and diffusion (Granovetter 1983). While there is a diversity of studies that describe static networks in various ways, these forms of analysis are still unable to capture the longitudinal aspects of social communities. Indeed, such a shift towards longitudinal network studies have been called for from within the SNA literature (Brass and Galaskiewicz 2004).

Recent computational techniques such as agent-based modeling (Macy and Willer 2002) and ERGM/p* methods for simulating the *evolution* of social networks is starting to make headway (Snijders 2001). However, there is a long way to go for integrating time smoothly into the larger framework of empirical studies utilizing SNA (see however Strang and Tuma (1993)). In the context of studying dynamic and complex social phenomena such as OSS we are especially concerned about the difficulties that such SNA studies have in portraying the structure of activities because the relational aspect of OSS activity is clearly limited – often there is very little by way of social relationships in OSS. However, there are well-established procedures and sequence-analytic methods that can help us to understand the structure of activities directly.

Sequence Analysis: a Computational Approach to Latent Procedural Structures

Traditionally process theories attempt to identify patterns of behavior in terms of their order and rhythm and further identify underlying generative mechanisms that drive processes forward and explain observed outcomes as necessary or sufficient conditions (Mohr 1982). These patterns of behaviors and their underlying generative mechanisms have been mainly captured through qualitative methods which through inductive reasoning seek to reveal the specific driving forces among individuals and their interactions in generating outcomes (Langley 1999). As a computational alternative, scholars such as Abbott (1992) have proposed a form of narrative positivism. This approach focuses on capturing and analyzing patterns of behavior or events so as to observe specific conditions under which specific outcome emerge. They also offer ways to capture underlying regularities in behavior and events. This approach uses computational sequence-analytic techniques deployed widely in gene sequencing within genetics and genomics. By using sequence analytic approaches to elicit behavioral patterns occurring in time, scholars can use large samples of digital trace data (e.g. Anjewierden and Efimova 2006) to detect common behavioral patterns over extended periods of time. Such techniques allow scholars to identify primitive patterns (behaviors or events) that are repeated in the data set, and by what regular pattern, if any, these primitives recur. Therefore, one can start with an observed pattern of ordering and rhythm, and eventually, the underlying generative mechanisms that give rise to the ordering and rhythm.

Sequence analysis is heavily computational (Gabadinho et al. 2011) and uses a finite set of categorical variables to express behaviors or events and their varying distributions across time. Sequence analysis has two primary functions: (1) to detect sequential patterns of actions over time and (2) to identify the alignment (or similarity) between those patterns. This happens typically through optimal matching techniques, which can be explained by considering the following two structured sequences typical in OSS development:

1: Issue / Commit / Commit
2: Pull Request / Commit / Commit

In order to measure the similarity in these activity patterns – the extent to which sequences align, or are similar – we need to estimate the effort required to transform one of the sequences into the other. In this example, we have to replace the Issue in sequence 1 with a Pull Request to arrive at sequence 2. This simple replacement allows us to quantify the "cost" of alignment, or distance between sequences, which in this example is 1 (Abbott and Hrycak 1990). Aligning sequences with more differences results in higher costs and distances. Higher costs means the sequences are more different, lower costs indicate more similarity. In order to align sequences to each other, an optimal matching algorithm must be applied to sequence data (two sequences minimum) that measures the distances in terms of number of substitutions, insertions and deletions (called indels) needed to transform one sequence into another. The total distance between two sequences is called an optimal matching distance (OM distance) and is a measure of dissimilarity – the degree to which two sequences are *not* similar to each other. In the context of OSS activity, computing OM distances enables us to describe the extent to which the sampled activity structures are similar or dissimilar.

Once sequences are aligned and described, we can estimate the probability distributions of activity transitions. Such transition probabilities are the key to detecting latent procedural structure in the form of ordering and rhythm. Wu (2000) points out that the transition probabilities that are derived from OM distances are agnostic with regards to whether sequences are compared forward or backwards, and that the ordering of events across time is not appropriately considered. Hence, our conceptualization of procedural structures needs to capture not only static transition tendencies, but also dynamic reconfigurations that form alternative grooved patterns across time. Such dynamic regularities in events over time can be captured using Variable Length Markov Chains (Buhlmann and Wyner 1999) captured in a Probabilistic Suffix Tree (PST). These describe the transition probabilities of events, predicated on the history of events in a particular sequence. Hence the probability of a certain event occurring next, is predicated on a history of events across several time periods. Given such a set of transition probabilities, we can then describe the dynamic ordering and rhythm of a latent procedural structure (Pentland 1995).

Methods Combining Relational and Procedural Structure

While both relational and procedural structures can offer useful insights into new forms of organizing for OSS, what holds the most potential for contributing to future research is the union of the two types of structure, and the types of studies that are made possible by this combination. Through describing human interaction as both relational and procedural we are able to see the way that configurations of relationships and streams of activities are associated with each other. This adds additional analytical value by providing means to theoretically and empirically triangulate phenomena that are not available when utilizing only a single view of social structure.

While procedural and relational forms of social structure are analytically distinct, they are derived from the same underlying behavioral material. Perhaps analogous to the way in which light can be understood as both waves and particles, the structure of social interaction can be analyzed in its procedural and relational forms. Both characterizations tell us something about the same phenomenon – yet in analytically different ways. When looking at data that describes interaction as it unfolds across time in a group of collaborators we can derive relational and procedural structure data from the same dataset. The different information extracted from the common ontological material is illustrated in figure 1 below.

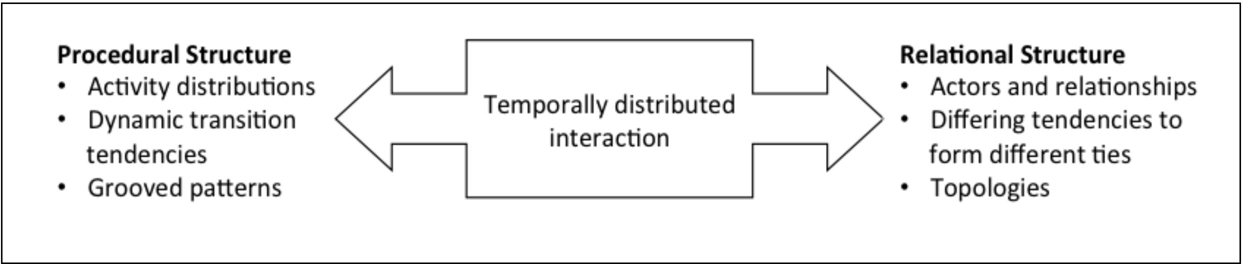
Procedural Structure
• Activity distributions
• Dynamic transition tendencies
• Grooved patterns
Temporally distributed interaction
Relational Structure
• Actors and relationships
• Differing tendencies to form different ties
• Topologies

Figure 1. Same source of data for both relational and procedural structures

Some headway has already been made in understanding how relational and procedural structures interact (See Table 4). For example, Pentland and Feldman (2007) draws upon Actor-Network Theory to propose narrative networks as a device for understanding how different events unfold in recurrent patterns that can be captured by evolving links between activities placed in networked relationships to each other. Further, Butts (2008) provides us with a relational event framework that allows us to model event sequences as they are predicated by past activities executed by multiple interconnected actors. Similarly, Brandes, Lerner, and Snijders (2009) proposed a model for statistical analysis of network evolution based on relational interaction data. Lastly, agent-based modeling (Macy and Willer 2002) provide a way of simulating the interaction of micro-level agents. Through computational simulations such interactions yield procedural structures, which, over time, build up to emergent relational structures. Through analytical models such as these, we can describe how the unfolding of activities is contingent on previous activities as well as the relational structures in which these event streams are embedded (see Table 4).

Table 4. Methods combining Relational and Procedural Structures

Method	**Description**	**Source of empirical data**	**How is Relational Structure captured?**	**How is Procedural Structure captured?**	**Citations**
Narrative Networks	Events are placed in a network where nodes activities are nodes, and ties are transitions	Qualitative Interviews	Activities are network nodes	Transitions between activities	Pentland and Feldman 2007
Relational Event Framework	Events are modeled based on preceding events executed by interacting alters	Qualitative Interviews	Activities are predicated on interaction partners	Activities are dependent on history	Butts 2008
Modeling Dyadic Event Data	Event probabilities are modeled based on previous interaction	Event Data programmatically extracted from news sources	Events encode who does what to whom	Interactions are temporally ordered	Brandes, Lerner, and Snijders 2009
Agent-based models	Micro-level agents interacting according to certain rules create emergent macro-structures	Qualitative data operationalized as interaction rules	Relational macro structures emerge based on micro-interactions	Procedural structures of interaction are generated by interaction rules	Bonabeau 2002; Macy and Willer 2002; Tubaro and Casilli 2010

Narrative networks, the relational event framework, and modeling of dyadic event data are essentially techniques for longitudinal social network analysis. As such, although they do pay attention to how networks change over time, they offer no clear operationalization of the pacing and rhythm of the sequences of activities associated with these changing networks. Further, agent-based models can generate both relational and procedural structures, but provides no means by which to analyze them. In order to provide for specific analyses of relational and procedural structure in concert with each other, we will outline an integrated example using a minimalist, fictive dataset. We show how relational and procedural structures can be elicited from the same data, and provide opportunities for viewing temporally distributed interaction from two distinct perspectives.

Table 5. Example Data

id	time	event	from	to
1	1	pull request	John	-
1	2	comment	Carl	John
1	3	commit	Lee	John
2	2	pull request	Don	-
2	4	commit	Lee	Don
3	3	pull request	Walter	John
3	4	comment	John	Walter
3	5	comment	Lee	John
3	7	commit	Carl	Walter

Table 5 above describes the dataset. In this fictitious example, we have three parallel processes that take place over 7 time periods. The id column identifies three distinct processes that partially overlap temporally, as can be seen from the time column. Each process consists of 2, 3, or 4 events, each of which is recorded in the event column. Further, most of these events are directed from a certain actor to other actors. For example, in process 1, at time 2, Carl comments on a pull request made by John. However, the pull request made by John at time 1 was not directed specifically at another actor. This dataset can now be displayed in its relational (Figure 2) and procedural (Table 6) forms, as can be seen below. In order to illustrate how relational and procedural structures may interrelate, we have focused our analysis on two distinct time periods, time 3 and 7. This allows us to see how relational and procedural structures preceding and succeeding each other in time are related. Further, notice how the network representations in Figure 2 compresses parallel processes into a cross-sectional snapshot for each time period, whereas the procedural representations in Table 6 and Figure 3 allows for analysis of the multi-threaded nature of temporally overlapping processes.

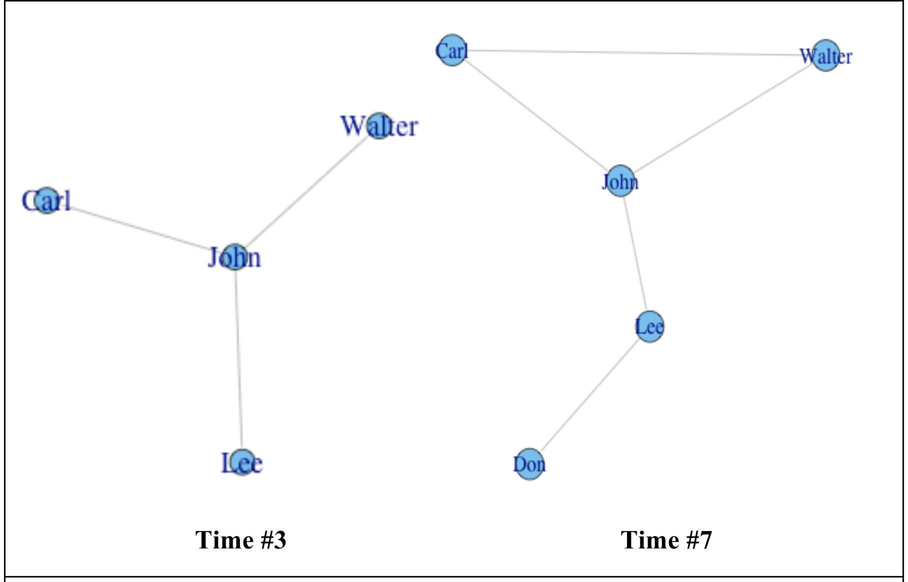

Figure 2. Temporally Distributed Interaction as Relations

Through descriptive network analysis a number of characteristics can be elicited to measure various aspects of the structure. For example the density (ratio of number of ties to number of possible ties) of the network is 0.5 at both t3 and t7, whereas the triad census (counting the number of triads), a measurement of the degree of transitivity in the network is 0 at t3 and 1 at t7. Additionally, for each node a coreness measure (Wasserman and Faust 1994) can be assessed. This will give us an indication of which nodes are part of a "core", meaning a subset of the network where each node has a density above a certain threshold. At t7, using a threshold of 2, Carl, Walter, & John would be part of a subgraph that could be considered a core, whereas Lee and Don fall below this threshold, and therefore are part of the periphery. At t3 however, only John would have a degree above 2.

Table 6. Temporally Distributed Interactions as Procedures

id	1	2	3	4	5	6	7
1	pull request	comment	commit	-	-	-	-
2	-	pull request	-	commit	-	-	-
3	-	-	pull request	comment	comment	-	commit

The sequence information displayed in Table 6 can then be transformed into a PST, as can be seen in Figure 3. In a minimalist example such as this one, the PST can be calculated by hand, through looking at the probability distribution across events at each position. Here, the term position refers not to each event type that is part of the alphabet (i.e. "pull request", "comment", and "commit"), but rather each position that has a unique history. For example, looking at the PST for t7, both pull request and commit has uniform probability distributions for subsequent events (i.e. pull requests are followed by comments in .67 of the cases, and commits in .33 of the cases, whereas commits are never followed by another event, thereby ending each branch). However, the comment event type occupies two distinct positions (pull request -> comment, and pull request -> comment -> comment). Therefore the event type comment appears in two positions, and the probabilities for subsequent events are different for each position.

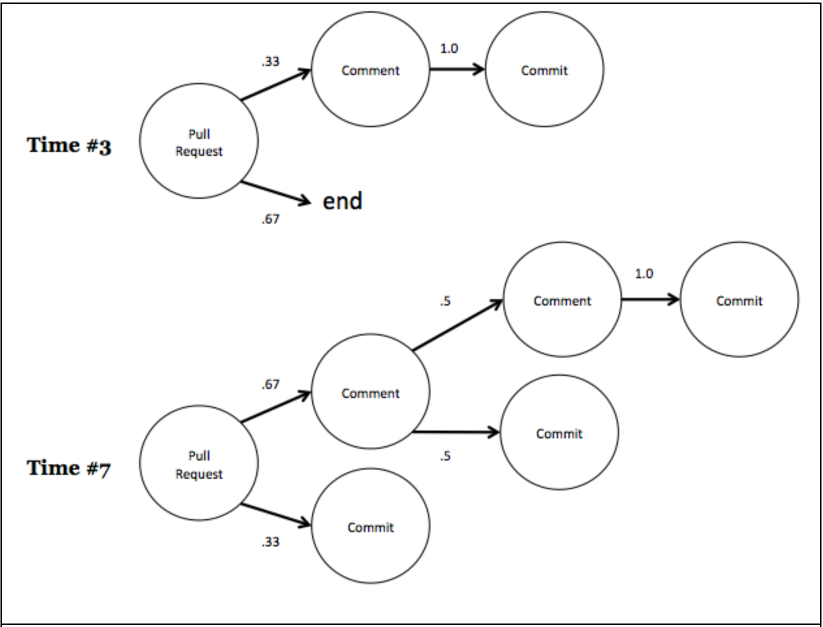

Figure 3. Fitted Probabilistic Suffix Trees

Beyond fitting the PST, we can also elicit a number of descriptive statistics with regards to the sequence data. For example, we can measure the minimum (1 at t3, 2 at t7) and maximum (3 at t3, 4 at t7) lengths of sequences in the data. We can also measure the average level of entropy (0.52 at t3, 0.66 at t7), which can be understood as the "uncertainty" of predicting the event distribution in the data (Gabadinho, Ritschard, & Studer, 2011). As such, it is a type of "pseudo-variance" measure that we can use in lieu of variance or standard deviation, which have no formal meanings in the context of sequence analysis.

From this example it is clear that same dataset can be viewed in distinctly different ways, through treating essentially the same data differently. These two representations highlight the dynamic nature of social structures as they unfold over time. The structure of the fitted PST models for time t3 and t7, show the generative process behind the relationships exhibited in the relational structures at time t3 and t7, respectively. Hence we can characterize the process, in terms of its procedural structure, that generates a certain relational structure. Further, looking at the relational structure at time t3, we can see the relationships set the stage for the unfolding of a specific sequence of events, captured by the PST fitted for t7. The specific relationships contained in the relational structure for t3, provides the context out of which a set of events, procedurally structured in a particular way, unfolds.

Research Questions: Combining Relational and Procedural Structure

Using the example above, we can now begin to form a number of research questions for investigating interactions between relational and procedural structures in the context of open source organizing. In the OSS community, relational and procedural structures are instantiated in coding practices and community structures. These structures relate to each other in complex and varied ways—relationships that have not yet been investigated. Therefore we propose a number of research questions on how these two interrelated types of structure can be approached methodologically in a balanced way.

RQ1: What procedural structures generate relational structures characterized by a strong core and a wide periphery? Previous research, as well as prescriptive accounts, have indicated the importance of a strong core handling a majority of code writing responsibilities while a wide periphery generates bug reports and feature requests (Mockus et al. 2002; Raymond 2001). The literature contains some tentative attempts at approaching this question (i.e. RQ1), looking at lifecycle models (Capiluppi et al. 2007) and growth patterns (Godfrey 2000). However, the processes through which such relational structures emerge still need to be explained. By examining the procedural structures that *precede* the emergence of desirable relational structures we can better understand the generativity of OSS processes. In particular, we can look at the particular ordering and rhythm of procedural structures. In the example above, it is clear that pull requests begin most sequences, as well as that commits end most sequences, which gives an indication of the general ordering of activities. We can also see that intermediate periods of commenting stretches out the rhythm by which these procedures unfold. Such longer rhythms contribute to building denser relational structures, and increase the degree of transitivity in the overall network. Such in-depth exploration of the linkages between distinct aspects of relational and procedural structure can help to uncover the generative mechanisms through which the various aspects of structure interrelate.

RQ2: How do procedural structures embedded in different relational structures unfold differently? The importance of a strong core and wide periphery in OSS is predicated on the desirable balance between code editing, bug reports, and feature requests that is argued to emerge from such a structure (Raymond 2001). While relational structures have been co-varied with various forms of success (technical success and community growth) in OSS (Grewal et al. 2006), we are still seeking to understand how it is that these relational structures shape unfolding activity streams – characterized by their procedural structures. Through examining the procedural structures that *follow* the emergence of a certain relational structures we can examine the effect of the relational structure on activities in later time periods. For example, in our data above, the emergence of a node with a higher density at t3 (a first step towards increased transitivity) heralds the unfolding of a more rhythmically complex procedural structure (indicated by the increasing degree of entropy from t3 to t7). Exploring the specifics of how different nodes are embedded in a relational structure characterized by a particular degree of transitivity can tell us about the procedural structures that may unfold subsequently. Further, the emergence of transitivity often point towards an increasing power-law distribution in the relational structure.

RQ3: What configurations of relational and procedural structures are sufficient and/or necessary for community growth? Combining relational and procedural structures allows us to research the way their combinations affect OSS outcomes. In other words, distinct procedural structures may produce different effects if they are embedded in, and therefore moderated by, different relational structures. The question can also be turned on its head to ask if the same relational structure can produce different effects given the unfolding of disparate procedural structures. Further, we might explore whether different configurations of procedural and relational structures produce sufficient and/or necessary conditions for certain outcomes to be produced. Hence, questions with regards to configurations of procedural and relational structures can be asked. Such configurations, arrived at using sequence analysis and SNA, can be analyzed using techniques such as Qualitative Comparative Analysis (Rihoux and Ragin 2008) to determine whether certain combinations of procedural and relational structural features are sufficient and/or necessary conditions of certain outcomes. This requires us to be able to define various features of social structure in a binary sense (either a structure has a certain characteristic, or it does not). For example, we might argue that the combination of procedural structures exhibiting stable ordering and rhythm, and relational structure showing a moderately transitive core and a periphery exhibiting lower transitivity are necessary conditions of high growth rates in the context of OSS. These types of configural effects allow us to examine multiple views of social structures simultaneously. Further, not only can we understand the impact of several types of social structures, but also the way that their combinations affect the outcomes of social activities.

Discussion

Computational social science has been made possible via increasing computational power, storage capabilities, automatic data collection and availability of public datasets. Several forms of computational social science have proliferated: SNA (Wasserman and Faust 1994), data mining (Kantardzic 2011), and natural language processing (Jurafsky and Martin 2009), just to name a few. Out of these, the analysis technique that has been most tightly associated with the analysis of social structure is SNA. All of these methods are empirical, have gained academic legitimacy, and are able to deliver insights which depart significantly from those that can be arrived at by traditional psychometric and econometric correlational analysis techniques.

The rise of SNA as a dominant force in empirical computational social science has also led to a proliferation of cross-sectional studies and the static assumptions that they come with. Due to its strong mathematical and structural assumptions, SNA studies tend to lean towards structural determinism (Burt 1987). This means that network topologies are argued to cause other constructs, such as performance, thereby establishing simple cause and effect linkages. Such structural determinism ignores the long tradition since Simmel (1950) of theorizing on agency and structure and how each constitutes the other (Giddens 1984). On a practical level, this structural determinism can be explained not by theoretical choices, but rather the relative mathematical tractability of various problems. It has simply been easy to calculate descriptive topological statistics and run regressions on external covariates, but difficult to model the process through which myriads of actions builds a network with a certain topology.

However, static representations of social structure cannot account for the temporal grooving of social patterns over time. This form of structure, however, is aptly captured by our conceptualization of procedural structure. Through procedural structures we can analyze the ways in which social interaction is patterned over time, and how such patterns relate to covariates of various kinds. While procedural structures are representations of social structure rather than agency, tracing procedural structures over time can help us to see how structure is in flux, and how changes in structure are related to the unfolding of activity streams, where the execution of activity represents an act of agency. Thus, the expansion of the social structure concept beyond relational structures to include procedural structures opens up a wider space for agency to be analyzed.

Additionally, there are clear data implications for this way of conceptualizing relational and procedural structures. In order to be able to examine research questions such as the ones proposed, detailed data needs to be collected on both relationships and activities. While this type of data can be collected through interviews and then coded by hand (Gaskin et al. forthcoming), this approach will hit severe limitations as we attempt to scale up studies of social structure to the larger sample sizes that are required to statistically detect subtle effects. Programmatic extraction of digital traces left by actors on platforms, such as version control systems hold the potential for collecting large datasets, with high levels of fidelity. Such data has a fine level of granularity (i.e., each action an actor executes on a platform is recorded faithfully) but cannot record psychometric data – only traces of behavior are left. While the behavioral data can tell us about patterns in behavior and correlations between various patterns, it can benefit from being complemented with qualitative data that can help us understand the generative mechanisms behind patterns and relationships.

Our research questions help us conceptualize not only spatial relationships, but also the ways in which we theorize about the unfolding of events across several temporal dimensions (Boland 2001). Our way of conceptualizing time does not only act metaphorically as clock-time, but also lays the conceptual basis for analyzing procedural structures quantitatively. Such possibilities help us to break down the traditional association between process studies and qualitative methods. Through mining processes for typical patterns we quantify processes and therefore co-vary them with external factors or overall characteristics that may influence (or be influenced by) the overall configuration of activities in a sequence. Further, such co-variation can be conditioned based off of present relational structures that may moderate or act in configural ways together with the particular procedural structures that are unfolding.

More broadly, we are arguing that the time is ripe also for going beyond the divide between quantitative and qualitative methods analyzing latent structures. Quantitative methods such as those showcased above do not rest on the linearity and independence assumptions of traditional variance based techniques. Rather, they explicitly recognize the interdependence of actors and the dynamic unfolding of relationships and events across time. While mixed methods research has become prevalent recently, much of it is about "sandwiching" traditional quantitative and qualitative studies so as to triangulate empirical results for complementarity and credibility (Onwuegbuzie and Collins 2007). However, we argue that through investigating the mutual constitution of latent relational and procedural structures using recent computational methods (Kilduff et al. 2006), we can move past simply layering studies on top of each other, and towards tighter theoretical integration of them at a more granular level.

Through capturing the logic of relational structure through networks as spatial ways of portraying interaction and supplementing this with a view of procedural structure as the dynamic unfolding of temporal activity patterns, we lay the groundwork for a multi-faceted way of understanding and analyzing social structures. These analytically distinct forms of social structure can be analyzed to understand how various combinations of structures affect the outcomes of relevant covariates. Since our distinction between relational and procedural structures is analytical, we can expect there to be associations between the ways in which relational and procedural structures emerge in the same empirical context. This is not meant to imply that associations between procedural and relational structures on the one hand, and relevant covariates (e.g. growth) on the other hand, will be identical. However, through examining both relational and procedural structures at the same time, we can better triangulate the underlying generative mechanisms. Therefore it is important to analyze the ways in which procedural and relational structures interact with each other so as to produce various effects. This may help us to tease out the nature of the underlying generative mechanisms more effectively than the study of each form of social structure separately.

Finally, in this paper, we demonstrate how procedural and relational structures can be applied to the context of open source software development – an example of open and digitalized collaboration. Through viewing coding practices and developer communities as instantiations of procedural and relational structures, we can use the analytical tools illustrated in this paper to analyze the way relational and procedural structures interact. The questions posed in this question should help us to dig deeper into the various ways in which social structure unfolds in the context of OSS. Through understanding the mechanisms through which relational and procedural structures mutually constitute each other, we can more aptly identify opportunities for practitioners to intervene in a project in ways that are more exact and effective. Further, scholars are enabled to isolate various dynamics that create emergent relational and procedural structures as well as their associations with outcome variables such as technical success and community growth.

Engaging in applying novel computational methods other than SNA, as well as crafting rich theoretical accounts will help to connect the various empirical patterns that have been uncovered in studies of OSS. While identifying core features of relational and procedural structures, both in the abstract as well as how they are instantiated in a certain context is important groundwork, it still begs questions with regards to how such patterns are generated and how they are related to each other as well as important outcome variables. In the context of OSS, project managers need to apply subtle community management strategies, since traditional command-and-control strategies are not available to be leveraged on a mostly volunteer-based community. Therefore a detailed understanding of the various dynamics through which structures are generated, configured, and combined so as to achieve certain outcomes is vital.

References

Abbott, A. 1992. “From Causes to Events: Notes on Narrative Positivism,” *Sociological Methods & Research* (20:4), pp. 428–455.

Abbott, A., and Hrycak, A. 1990. “Measuring resemblance in sequence data: An optimal matching analysis of musicians’ careers,” *American Journal of Sociology* (96:1), pp. 144–185.

Ancona, D. G., and Chong, C. L. 1992. “Entrainment–cycles and synergy in organizational behavior,”Alfred P. Sloan School of Management, Massachusetts Institute of Technology, 1992.

Anjewierden, A., and Efimova, L. 2006. “Understanding Weblog Communities Through Digital Traces: A Framework , a Tool and an Example,” In *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*Springer Berlin Heidelberg, pp. 279–289.

Barabási, A.-L., and Reka, A. 1999. “Emergence of Scaling in Random Networks,” *Science* (286:5439), pp. 509–512.

Blau, P., and Scott, W. 1962. *Formal Organizations: A Comparative Approach*, Stanford University Press.

Boland, R. 2001. “The tyranny of space in organizational analysis,” *Information and Organization* (11), pp. 3–23.

Bonabeau, E. 2002. “Agent-based modeling: methods and techniques for simulating human systems,” *Proceedings of the National Academy of Sciences of the United States of America* (99 Suppl 3), pp. 7280–7287.

Borgatti, S., and Foster, P. 2003. “The Network Paradigm in Organizational Research: A Review and Typology,” *Journal of Management* (29), pp. 991–1013.

Bourdieu, P. 1977. *Outline of a theory of practice.*, Cambridge University Press, pp. 248.

Brandes, U., Lerner, J., and Snijders, T. 2009. “Networks Evolving Step by Step: Statistical Analysis of Dyadic Event Data,” In *International Conference on Advances in Social Network Analysis and Mining*, pp. 200–205.

Brass, D., and Galaskiewicz, J. 2004. “Taking Stock Of Networks And Organizations: A Multilevel Perspective,” *Academy of Management Journal* (47:6), pp. 795–817.

Buhlmann, P., and Wyner, A. J. 1999. “Variable Length Markov Chains,” *The Annals of Statistics* (27:2), pp. 480–513.

Burt, R. 1987. “Social contagion and innovation: Cohesion versus structural equivalence,” *American Journal of Sociology* (92:6), pp. 1287–1335.

Burt, R. 2000. “The network structure of social capital,” In *Research in Organizational Behavior* (Vol. 22), pp. 345–423.

Butts, C. 2008. “A Relational Event Framework for Social Action,” *Sociological Methodology* (38:1), pp. 155–200.

Capiluppi, A. 2004. “Improving comprehension and cooperation through code structure,” *26th International Conference on Software Engineering - W8S Workshop “Collaboration, Conflict and Control: The 4th Workshop on Open Source Software Engineering,”* pp. 23–28.

Capiluppi, A., González-Barahona, J. M., Herraiz, I., and Robles, G. 2007. “Adapting the ‘staged model for software evolution’ to free/libre/open source software,” *Ninth international workshop on Principles of software evolution in conjunction with the 6th ESEC/FSE joint meeting - IWPSE ’07*New York, New York, USA: ACM Press, pp. 79.

Christley, S., and Madey, G. 2007. “Analysis of Activity in the Open Source Software Development Community,” In *40th Annual Hawaii International Conference on System Sciences (HICSS’07)*, pp. 1–10.

Coleman, J. S. 1958. “Field Methods and Techniques Relational Analysis: The Study of Social Organizations with Survey Methods,” *Human Organization* (17:4), pp. 28–36.

Conaldi, G., Lomi, A., and Tonellato, M. 2012. “Dynamic Models of Affiliation and the Network Structure of Problem Solving in an Open Source Software Project,” *Organizational Research Methods* (15:3), pp. 385–412.

Cressie, N. 1993. *Statistics for Spatial Data*, Wiley-Interscience, pp. 928.

Crowston, K., and Scozzi, B. 2004. “Coordination practices within FLOSS development teams: The bug fixing process,” *Computer Supported Activity Coordination* (4:1), pp. 21–30.

Crowston, K., Wei, K., Howison, J., and Wiggins, A. 2012. “Free/Libre open-source software development: What We Know and What We Do Not Know,” *ACM Computing Surveys* (44:2), pp. 1–35.

Crowston, K., Wei, K., Li, Q., and Howison, J. 2006. "Core and periphery in Free/Libre and Open Source software team communications," In *Proceedings of the 39th Hawaii International Conference on System Sciences*, pp. 1–7.

DiMaggio, P., and Powell, W. 1983. "The Iron Cage Revisited: Institutional Isomorphism and Collective Rationality in Organizational Fields," *American Sociological Review* (48:2), pp. 147–160.

DiMaggio, P., and Powell, W. 1991. *The new institutionalism in organizational analysis*, Chicago: University Of Chicago Press, pp. 486.

Dinh-Trong, T., and Bieman, J. 2005. "The FreeBSD Project: A Replication Case Study of Open Source Development," In *IEEE Transactions on Software Engineering* (Vol. 31), pp. 481–494.

Dionysiou, D. D., and Tsoukas, H. 2013. "Understanding the (Re)Creation of Routines from Within: A Symbolic Interactionist Perspective," *Academy of Management Review* (38:2), pp. 181–205.

Drucker, P. 1988. "The Coming of the New Organization," *Harvard Business Review* (January-Fe), pp. 3–11.

Eisenhardt, K., and Martin, J. 2000. "Dynamic capabilities: what are they?," *Strategic Management Journal* (21:10-11), pp. 1105–1121.

Feldman, M. S. 2003. "A performative perspective on stability and change in organizational routines," *Industrial and Corporate Change* (12:4), pp. 727–752.

Feldman, M. S., and Pentland, B. T. 2003. "Reconceptualizing Organizational Routines as a Source of Flexibility and Change," *Administrative Science Quarterly* (48:1), pp. 94–121.

Feller, J., and Fitzgerald, B. 2000. "A framework analysis of the open source software development paradigm," *Proceedings of the twenty first international conference on Information systems*, pp. 58–69.

Gabadinho, A., Ritschard, G., and Studer, M. 2011. "Analyzing and Visualizing State Sequences in R with TraMineR," *Journal Of Statistical Software* (40:4).

Gaskin, J., Berente, N., Lyytinen, K., and Yoo, Y. (forthcoming). "Toward Generalizable Sociomaterialy Inquiry: A Computational Approach for Zooming In and Out of Sociomaterial Routines," *MIS Quarterly* (X:X).

Gaskin, J., Lyytinen, K., Thummadi, V., Schutz, D., Yoo, Y., Weiss, A., and Berente, N. 2010. "Sequencing Design DNA: A Set of Methodological Artifacts for Sequencing Sociotechnical Design Routines," In *ICIS 2010 Proceedings*, pp. 1–15.

Gersick, C. J., and Hackman, J. R. 1990. "Habitual routines in task-performing groups.," *Organizational behavior and human decision processes* (47), pp. 65–97.

Giddens, A. 1979. *Central Problems in Social Theory: Action, Structure and Contradictions in Social Analysis*, University of California Press, pp. 294.

Giddens, A. 1984. *The Constitution of Society: Outline of the Theory of Structuration*, University of California Press.

Godfrey, M. 2000. "Evolution in open source software: a case study," In *Proceedings International Conference on Software Maintenance ICSM-94*, pp. 131–142.

Granovetter, M. 1983. "The Strength of Weak Ties: A Network Theory Revisited," *Sociological Theory* (1), pp. 201–233.

Granovetter, M. 1985. "Economic Action and Social Structure : The Problem of Embeddedness," *American journal of sociology* (91:3), pp. 481–510.

Granovetter, M. 1990. "The Myth of Social Network Analysis as a Special Method in the Social Sciences," *Connections* (13:1-2), pp. 13–16.

Grewal, R., Lilien, G. L., and Mallapragada, G. 2006. "Location, Location, Location: How Network Embeddedness Affects Project Success in Open Source Systems," *Management Science* (52:7), pp. 1043–1056.

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

Healy, K., and Schussman, A. 2003. "The Ecology of Open-Source Software Development,"Tucson, AZ, pp. 1–24.

Von Hippel, E., and Von Krogh, G. 2003. "Open Source Software and the 'Private-Collective' Innovation Model: Issues for Organization Science," *Organization Science* (14:2), pp. 209–223.

Howison, J., and Crowston, K. (forthcoming). "Collaboration through open superposition," *MIS Quarterly* (X:X).

Howison, J., Wiggins, A., and Crowston, K. 2011. "Validity Issues in the Use of Social Network Analysis with Digital Trace Data," *Journal of the Association for Information Systems* (12:12), pp. 767–797.

Jurafsky, D., and Martin, J. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech*, Pearson Prentice Hall, pp. 1024.

Kantardzic, M. 2011. *Data Mining: Conceps, Models, Methods, and Algorithms*, Wiley-IEEE Press, pp. 328–384.

Kilduff, M., Tsai, W., and Hanke, R. 2006. "A Paradigm Too Far? A Dynamic Stability Reconsideration Of The Social Network Research Program," *Academy of Management Review* (31:4), pp. 1031–1048.

Koch, S. 2005. "Evolution of open source software systems–a large-scale investigation," *Proceedings of the First International Conference on Open Source Systems Genova, 11th-15th July 2005*, pp. 148–153.

Koch, S., and Schneider, G. 2002. "Effort, co-operation and co-ordination in an open source software project: GNOME," *Information Systems Journal* (12), pp. 27–42.

Krishnamurthy, S. 2002. "Cave or Community?: An Empirical Examination of 100 Mature Open Source Projects," *First Monday* (2).

Von Krogh, G., Spaeth, S., and Lakhani, K. R. 2003. "Community, joining, and specialization in open source software innovation: a case study," *Research Policy* (32:7), pp. 1217–1241.

Langley, A. 1999. "Strategies for theorizing from process data," *Academy of Management Review* (24:4), pp. 691–710.

Lazer, D., Pentland, A. S., Adamic, L., Aral, S., Barabasi, A. L., Brewer, D., Christakis, N., Contractor, N., Fowler, J., Gutmann, M., and others. 2009. "Life in the network: the coming age of computational social science," *Science* (323:5915), pp. 721–723.

Macy, M. M. W., and Willer, R. 2002. "From Factors to Actors: Computational Sociology and Agent-Based Modeling," *Annual Review of Sociology* (28:1), pp. 143–166.

Madey, G., Freeh, V., and Tynan, R. 2002. "The Open Source Software Development Phenomenon: An Analysis Based on Social Network Theory," In *AMCIS 2002 Proceedings*, pp. 1806–1813.

Mateos-Garcia, J., and Steinmueller, W. 2008. "The Institutions of Open Source Software: Examining the Debian Community," *Information Economics and Policy* (20:4), pp. 333–344.

Merton, R. K. 1957. *Social Theory and Social Structure*, New York: Free Press.

Mintzberg, H. 1979. "The structuring of organizations: A synthesis of the research," In *University of Illinois at Urbana-Champaign's Academy for Entrepreneurial Leadership Historical Research Reference in Entrepreneurship*.

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

Mohr, L. 1982. *Explaining Organizational Behavior*, Jossey-Bass, pp. 260.

Monge, P., and Contractor, N. 2003. *Theories of Communication Networks*, New York: Oxford University Press, pp. 432.

Monteiro, E., and Østerlie, T. 2004. "Keeping it going: The everyday practices of open source software," *Working paper, Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway*.

Moon, J. Y., and Sproull, L. 2000. "Essence of Distributed Work: The Case of the Linux Kernel," *First Monday* (5:11), pp. 1–16.

Nelson, R. R., and Winter, S. G. 1982. *An Evolutionary Theory of Economic Change*, Cambridge, MA: Harvard University Press, pp. 437.

Oh, W., and Jeon, S. 2007. "Membership Herding and Network Stability in the Open Source Community: The Ising Perspective," *Management Science* (53:7), pp. 1086–1101.

Onwuegbuzie, A., and Collins, K. 2007. "A typology of mixed methods sampling designs in social science research," *The Qualitative Report* (12:2), pp. 281–316.

Parsons, T. 1960. *Structure and Process in Modern Societies*, New York: Free Press, pp. 344.

Pentland, B., and Rueter, H. 1994. "Organizational routines as grammars of action," *Administrative Science Quarterly* (39:3), pp. 484.

Pentland, B. T. 1995. "Grammatical Models of Organizational Processes," *Organization Science* (6:5), pp. 541–556.

Pentland, B. T. 2005. "Organizational routines as a unit of analysis," *Industrial and Corporate Change* (14:5), pp. 793–815.

Pentland, B. T., and Feldman, M. S. 2007. "Narrative Networks: Patterns of Technology and Organization," *Organization Science* (18:5), pp. 781–795.

Powell, W. 1990. "Neither Market nor Hierarchy: Network Forms of Organization," *Research in Organizational Behavior* (12), pp. 295–336.

Rajlich, V., and Bennett, K. 2000. "A Staged Model for the Software Life Cycle," *Computer* (33:7), pp. 66–71.

Raymond, E. S. 2001. *The cathedral and the bazaar: musings on Linux and Open Source by an accidental revolutionary*, O'Reilly, pp. 241.

Rihoux, B., and Ragin, C. 2008. *Configurational Comparative Methods: Qualitative Comparative Analysis (QCA) and Related Techniques*, (Vol. 51).

Robles, G., Amor, J. J., Gonzalez-barahona, J. M., Herraiz, I., Rey, U., and Carlos, J. 2005. "Evolution and Growth in Large Libre Software Projects," In *Proceedings of the 2005 Eighth International Workshop on Principles of Software Evolution (IWPSE'05)*, pp. 1–10.

Shen, C., and Monge, P. 2011. "Who connects with whom? A social network analysis of an online open source software community," *First Monday* (16:6), pp. 1–15.

Simmel, G. 1950. *The Sociology of Georg Simmel*, New York: The Free Press, pp. 445.

Snijders, T. 2001. "The statistical evaluation of social network dynamics," *Sociological Methodology* (31:1), pp. 361–395.

Strang, D., and Tuma, N. 1993. "Spatial and Temporal Heterogeneity in Diffusion," *American Journal of Sociology* (99:3), pp. 614–639.

Thornton, P. H., Ocasio, W., and Lounsbury, M. 2012. *The Institutional Logics Perspective: A New Approach to Culture, Structure, and Process*, Oxford University Press, pp. 234.

Tubaro, P., and Casilli, A. 2010. "An Ethnographic Seduction": How Qualitative Research and Agent-based Models can Benefit Each Other," *Bulletin de Méthodologie Sociologique* (106:1), pp. 59–74.

Valverde, S., and Solé, R. 2007. "Self-organization versus hierarchy in open-source social networks," *Physical Review E* (76:4), pp. 1–8.

Van De Ven, A. H., and Poole, M. S. 1995. "Explaining Development and Change in Organizations," *Academy of Management Review* (20:3), pp. 510–540.

Wasserman, S., and Faust, K. 1994. *Social Network Analysis: Methods and Applications*, Cambridge, England: Cambridge University Press, pp. 825.

Watts, D., and Strogatz, S. 1998. "Collective dynamics of 'small-world' networks," *Nature* (393:June), pp. 440–442.

Wellman, B., and Berkowitz, S. 1988. *Social structures: A network approach*, Cambridge, England: Cambridge University Press, pp. 513.

Wu, L. 2000. "Some comments on' Sequence analysis and optimal matching methods in sociology: Review and prospect'," *Sociological methods and research* (29:1), pp. 41–64.