Striking a balance between trust and control in a virtual organization: 
a content analysis of open source software case studies

Michael J. Gallivan
Georgia State University, Robinson College of Business, Atlanta, GA 30302-4015, USA, 
email: mgallivan@gsu.edu

Abstract. Many organization theorists have predicted the emergence of the net-
worked or virtual firm as a model for the design of future organizations. 
Researchers have also emphasized the importance of trust as a necessary con-
dition for ensuring the success of virtual organizations. This paper examines the 
open source software (OSS) 'movement' as an example of a virtual organization 
and proposes a model that runs contrary to the belief that trust is critical for virtual 
organizations. Instead, I argue that various control mechanisms can ensure the 
effective performance of autonomous agents who participate in virtual organiza-
tions. Borrowing from the theory of the 'McDonaldization' of society, I argue that, 
given a set of practices to ensure the control, efficiency, predictability and calcula-
tibility of processes and outcomes in virtual organizations, effective performance 
may occur in the absence of trust. As support for my argument, I employ content 
analysis to examine a set of published case studies of OSS projects. My results 
show that, although that trust is rarely mentioned, ensuring control is an impor-
tant criterion for effective performance within OSS projects. The case studies 
feature few references to other dimensions of McDonaldization (efficiency, pre-
dictability and calculability), however, and I conclude that the OSS movement relies 
on many other forms of social control and self-control, which are often unack-
nowledged in OSS projects. Through these implicit forms of control, OSS pro-
jects are able to secure the cooperation of the autonomous agents that participate 
in project teams. I conclude by extrapolating from these case studies to other 
virtual organizations.

Keywords: Control, McDonaldization, networked organization, rationalization, 
trust, virtual organization

© 2001 Blackwell Science Ltd
INTRODUCTION

The open source software (OSS) movement has attracted considerable enthusiasm among investors and managers with decision-making authority for computer products and services. Recent events reported in the media show that both OSS products and OSS processes have captured tremendous media attention and excitement. Some OSS products often mentioned include Emacs (text editor), Apache (web-server), fetchmail and sendmail (Internet mail utilities), X-window (user interface), Perl (programming language) and the latest release of Netscape Navigator (browser software) – better known as the Mozilla project. Although these are but a few of the many products that resulted from networks of programmer teams contributing their efforts to OSS over the past decade, the product that has stolen the limelight is Linux, the OSS variant of the UNIX operating system. Linux has given investors a dramatic ride over the past 18 months, with the stock of companies such as RedHat spiralling to over $250 per share before plummeting back to earth.

Within the past few months, Linux has received a vote of confidence from several critical players in the hardware products sector. In recent press announcements, IBM, Dell Computer and Oracle have all pledged allegiance to Linux products and have promised to offer support services for their customers who operate in a Linux environment. IBM’s president and CEO, Sam Palmisano, announced that ‘Linux is ready for real business’, while a Merrill Lynch investment analyst referred to IBM’s vote of confidence as ‘a mammoth endorsement of Linux’, which signals a ‘re-branding of the IBM server line’ (Anonymous, 2001a,b,c). IBM plans to invest over $1 billion in new business unit activities for Linux, with $300 million earmarked for consulting and support for Linux customers.

Not only have the OSS products drawn considerable attention, but also equally important are its processes – its unique commingling of volunteerism, concurrent engineering and rapid application development. The rise of the OSS ‘movement’ – as it has been dubbed by various observers (DiBona et al., 1999; Markus et al., 2000; O’Reilly, 2000) – is auspicious because it stands as one of the clearest and most successful examples of a virtual organization. Virtual organizations are aggregations of autonomous agents communicating and collaborating to achieve common goals. In this paper, I examine the literature on virtual organizations and interorganizational alliances to understand what ingredients are necessary for their effective performance. Although it is fashionable among business scholars and practitioners to assume that trust is a prerequisite to effective performance in virtual organizations and alliances, this paper offers an alternative view.

I argue that the OSS movement is a prototypical virtual organization, in part, because – like other virtual organizations – it relies not on trust per se, but rather upon an overarching set of control mechanisms (both explicit and implicit) that, in aggregate, ensure effective performance. To support this argument, I review several leading essays and case studies documenting OSS projects to show how various mechanisms of social control and self-control serve to ensure effective performance. Trust, although a lofty ideal and an oft-cited factor, rarely enters the picture – mostly, on account of the fact that key processes within OSS pro-
jects are well controlled, and members rarely make themselves vulnerable to others’ actions (or others’ failure to act). This paper explains how a variety of mechanisms may ultimately ensure high-quality, reliable performance and other attributes of successful organizations, in the absence of trust.

This paper is structured as follows. Next, I define trust and briefly survey the trust literature, as psychologists, sociologists, philosophers and economists have studied it. I summarize the vast research in this area – necessary for understanding trust in both organizational and interorganizational contexts. The third section identifies the growing importance of the new forms of organization, known as virtual or modular organizations, and underscores the general movement away from hierarchical, vertically integrated firms towards greater use of market mechanisms and networked arrangements, such as alliances among autonomous agents. It also documents the conventional wisdom that virtual organizations are believed to require high levels of trust in order to ensure effective performance. The fourth section argues that OSS projects are a type of virtual organization and defines a set of metrics for effective group performance. With this grounding in the concepts of trust and virtual organizations, the fifth section offers an alternative to the conventional belief that trust is necessary for virtual organizations. There, I introduce the framework of ‘McDonaldization’ – based on Ritzer’s (1996) social critique of the ‘rationalization’ that he perceived as destroying the meaning and richness of everyday life. Then, I argue that the precepts of McDonaldization may also be framed in a more neutral light as a lens for understanding how virtual organizations may ensure effective performance, such as in OSS project teams.

The next section presents a content analysis of nine case studies of OSS projects. Drawing from these case studies that have chronicled the rise of the OSS initiative, I seek evidence to corroborate the conventional wisdom that trust is required among the autonomous agents who constitute a virtual organization, only to come up short. Alternatively, I highlight several recurring themes to show that various forms of control appear to guarantee effective performance of OSS projects. I conclude by extrapolating from this observation that trust is not critical in contemporary governance arrangements – such as virtual and networked organizations – and I speculate on the limits of social control and consider how a healthy balance between trust and control can be attained.

TRUST IN AN ORGANIZATIONAL CONTEXT

The concept of trust is widespread in the social sciences and has been examined in the literature of many disciplines: economics, sociology, psychology and philosophy. Not surprisingly, each of these disciplines has viewed trust in a slightly different light. Recent summaries of the literature have proposed several different taxonomies of trust (Zucker, 1986; Brockner & Siegel, 1996; Kramer & Tyler, 1996). In this section, I provide a brief overview of the trust literature, in order to identify key definitions that all (or nearly all) trust researchers would agree upon. Readers interested in further details, including many of the issues that are subjects of
debate, are directed to recent publications by Kramer & Tyler (1996) and to special issues of
academic journals, such as the *Academy of Management Review* (Rousseau et al., 1998),
*Administrative Science Quarterly* and *Organization Science* (forthcoming).

Nearly all definitions of trust share the condition that one party (the truster) must willingly
place himself or herself in a position of vulnerability to or risk from another party (the trustee).
That is, the truster makes itself potentially vulnerable to opportunistic behaviour by the trustee,
regardless of the truster’s ability to monitor the trustee’s behaviour (Brenkert, 1998; Husted,
1998). This condition of vulnerability is key to any definition of trust. For example,
Karahannas & Jones (1999: 347) note that trust is ‘closely related to risk, since without vul-
erability to the risk of opportunism, there is no need for trust.’

Beyond their agreement that trust assumes the truster’s vulnerability to the trustee,
researchers have focused on many different aspects of trust. In the management and soci-
ology literatures alone, at least five styles of trust have been defined by leading researchers
(Husted, 1998; Zucker, 1986). Although precise definitions and examples are beyond the scope
of this paper, I list these five different styles of trust briefly:

- knowledge-based trust – trust based upon a prior history of transactions between two
  parties;
- characteristic-based trust – trust that is assumed, based on certain attributes of the other
  party;
- institutional-based trust – a trusting environment, as ensured by guarantor agencies;
- justice-based trust – related to the concept of procedural justice (i.e. ensuring fair
  procedures);
- swift trust – a ‘fragile’ form of trust that emerges quickly in virtual workgroups and teams.

Although most of these forms of trust have been widely discussed in the literature, the last
term – swift trust – was coined only recently (Meyerson et al., 1996) to describe a unique form
of trust that develops quickly among members of a virtual organizational or team, but is fragile
and may easily be destroyed (Jarvenpaa & Leidner, 1999). To investigate the context of virtual
organizations further, the next section reviews literature on virtual organizations and related
topics, such as alliances, interorganizational exchange relationships and outsourcing.

WHAT ARE VIRTUAL ORGANIZATIONS?

To adapt to the increasingly global competitiveness of the market, shorter product lifecycles,
IT-enabled opportunities and the realities of ‘competing on Internet time’, firms are adopting
new organizational forms. Most of these new forms incorporate some level of ‘virtuality’ or
‘modularity’ – that is, allowing certain processes to be performed outside the traditional bound-
aries of the firm (Lucas, 1996). Among these forms are strategic alliances, joint ventures, part-
nerships, as well as a host of novel terms referring to the increased reliance on external agents
– such as modular (Sanchez & Mahoney, 1996), boundaryless (Ashkenas et al., 1995), virtual
(Mowshowitz, 1997) and network organizations (Nohria & Eccles, 1992). One commonality
among all the novel organizational forms listed above is that they involve external relationships at a deeper level than the traditional, arms-length buyer–supplier exchange. Such organizational forms enable the firm to focus on its core competencies while outsourcing non-core functions and processes to agents that excel in those tasks (Barney, 1986, 1991; Prahalad & Hamel, 1990).

The concept of virtuality has piqued the interest of researchers and practitioners from a range of disciplines. Similar to the concept of virtual computer memory, a virtual organization can assume the appearance of a much larger entity, in part because it achieves many of its goals by relying on close coordination with external parties, partners and other agents (Carr, 1999). Virtual organizations are able to accomplish more than vertically integrated, hierarchical organizations with the same internal resources, by leveraging resources and capabilities beyond their organizational boundaries. Such exchange relationships may range from those of fleeting duration – as in spot market exchanges – to more lasting recurrent and relational exchanges (Ring & Van de Ven, 1992). Sawhney & Parikh (2001) define the advantages of virtual organizations as resulting from the ‘networked intelligence’ they enable among the flexible components or ‘modules’ that comprise them. They argue that disaggregating functionality into modules can ensure flexibility and efficiency yet, in the future, the primary challenge will be the ability to integrate these modules successfully into a coherent whole.

In a networked world... economic value behaves very differently than it does in the traditional, bounded world. Devices, software, organizational capabilities and business processes will increasingly be restructured as well-defined, self-contained modules that can be quickly and seamlessly connected with other modules. Value will lie in creating modules that can be plugged in to as many different value chains as possible. . . . The ability to coordinate among the modules will become the most valuable business skill (Sawhney & Parikh, 2001: 81).

I will use the term virtual organization to describe such non-hierarchical configurations of activity, and I refer to the parties that coordinate their activities as agents. These agents may be individuals, as well as small or large business entities, or possibly non-profit organizations. The term agent is deliberately ambiguous, as it does not specify the size of the entity or whether it is a single person or an organization. That is because it is not the agent’s size that matters, but rather the fact that it has the capability to enter into flexible relationships of varied durations. A virtual organization is goal directed and consists of geographically distributed agents who may or may not ever meet face to face. The term ‘organization’ is general, in terms of referring to a multitude of different types of entities, and it need not correspond to a for-profit business entity.

Research on virtual organizations

One of the first authors to study virtual organizations was Byrne (1993), who defined them as ‘a temporary network of independent companies – suppliers, customers, even erstwhile...
rivals – linked by IT to share skills, costs and access to one another’s markets. . . . [They] have no hierarchy, no vertical integration’ (Byrne, 1993: 99). The label ‘virtual’ has been applied to many domains, in addition to virtual organizations, producing notions of virtual computer memory, virtual teams, virtual workplaces, virtual classrooms and even virtual reality (Mowshowitz, 1997). Computer scientists have defined a virtual organization as one that can ‘dynamically link its business goals with the procedures needed to achieve them’ (Mowshowitz, 1997: 30), while strategy researchers have described virtual organizations as ‘where free agents come together to buy and sell one another’s goods and services . . . [to] harness the power of market forces’ (Chesbrough & Teece, 1996: 65). Some researchers have characterized virtual organizations as ‘hollow’ – although this depiction has been offered as alternatively an asset (Malone & Laubacher, 1998) or a liability (Chesbrough & Teece, 1996).

**Trust requirements for virtual organizations**

Many consultants and researchers have argued that trust will be increasingly important for virtual organizations – indeed, for any firm that relies on outsourcing various functionality to other organizations. In response to the query, ‘how viable is the virtual corporation?’ Jones & Bowie argue that ‘the efficiency of such corporations depends on features – speed and flexibility – that require high levels of mutual trust and cooperation (Jones & Bowie, 1998: 276). O’Leary et al. (2001) observed that ‘in publications regarding distributed work (e.g. Handy, 1995) . . . trust is described as critical . . . because it is impossible to monitor and control geographically distributed employees’ (p. 3). Both sets of authors above cited Handy (1995), who was the first to examine the need for trust in virtual organizations:

> It is easy to be seduced by the technological possibilities of the virtual organization, but . . . the managerial dilemma comes down to the question, How do you manage people whom you do not see? The simple answer is, by trusting them. . . . If we are to enjoy the . . . benefits of the virtual organization, we will have to rediscover how to run organizations based more on trust than on control. Virtuality requires trust to make it work: technology on its own is not enough (Handy, 1995: 42, 44).

Much of contemporary research on new organizational forms has examined the role of trust in outsourcing arrangements and buyer–supplier alliances (Zaheer et al. 1998). For example, Urban et al. (1999) summarized the literature on purchasing alliances, noting that both Morgan & Hunt (1994) and Spekman (1988) ‘emphasized that trust is essential for long-term buyer–supplier relationships’ (Urban et al. 1999: 7). In another study based on a survey of 1500 members of the National Association of Purchasing Management, respondents perceived that the key strategy for successful buyer–supplier relationships was increasing trust along the supply chain (Wisner & Tan, 2000), particularly for virtual organizations – or what the authors characterized as organizations that were structured ‘within the value chain as a unified virtual business entity’. Hartley (2000) reached a similar conclusion that trust was required to improve outsourcing relationships; however, she found this to be problematic, as existing trust levels were so low.
Hartley claimed that ‘high-trust buyer–supplier relations in the US automotive industry were the exception rather than the rule’ (Hartley 2000: 30). Likewise, marketing researchers Urban et al. (2000) have emphasized the importance of ensuring the client firm’s trust in a supplier’s B2B web presence. Recent research on ecommerce systems has even focused on redesigning the user interface to increase buyers’ trust in the vendor and its systems through techniques such as ‘embodied conversational agents’ (Cassell & Bickmore 2000) and ‘trust-based virtual personal advisors’ (Urban et al., 1999).

Although there appears to be consensus that trust is critical for virtual organizations and other new forms of interorganizational exchange, contradictory advice is frequently offered to managers – at times, including conflicting advice within a given study. For example, in offering recommendations to managers, Harrison & St John (1996) cautioned them to ‘avoid formalization and monitoring of contractual agreements, which [may] lead to conflict and distrust’, yet they also advised managers to ‘avoid excessive trust, which leads to its violation (e.g. embezzlement, fraud)’. Markus (2000) likewise advised against using excessive control in overseeing work performed by outsourcing partners. Among other problems, she warns that too much control may ‘damage the relationship between the controllers and controlees’, leading to ‘unintended human and social consequences [which] . . . include low morale and circumventing the rules’ (Markus, 2000: 175).

In summary, the conventional wisdom argues that trust is critical to the success of virtual organizations and interorganizational alliances, but that ensuring trust is difficult and that having either too much trust or too much control can be dysfunctional. I disagree with the claim that trust is critical. Instead, I argue that an agent in a virtual organization need not make itself vulnerable to opportunistic behaviour by other agents, regardless of its own ability to control the outcomes. Instead, I believe that each agent can have confidence in the behaviour and outputs of other agents if certain principles and practices that ensure control are implemented. Rather than seeing a critical role for trust, I believe that, by controlling the conditions for collaboration and norms of behaviour, any agent can have confidence in other members of the virtual organization, thus obviating the need for trust.

In the next sections, I explain why the OSS movement can be considered a prototype of a virtual organization, and I also use the theory of McDonaldization (Ritzer, 1996) to identify specific control mechanisms that ensure effective performance of the individuals and teams who participate in the OSS movement.

**WHY OSS PROJECTS ARE VIRTUAL ORGANIZATIONS**

Many essays and critiques of the OSS movement have emphasized the distributed nature of the networks of programmers that collaborate on its tasks (Raymond, 1996). In this paper, I consider the OSS ‘movement’ as an example of a virtual organization, and I investigate specific OSS development projects as virtual organizations. Although it may appear that the OSS is an unorthodox example of a virtual organization, Markus et al. (2000) emphasized the ‘relatively high degree of correspondence between the open-source movement and popular

The Linux community, a temporary, self-managed gathering of diverse individuals engaged in a common task, is a model for a new kind of business organization that could form the basis for a new kind of economy. The fundamental unit of such an economy is not the corporation but the individual. . . . These electronically connected freelancers – e-lancers – join together into fluid and temporary networks. . . . Far from being a wild hypothesis, the e-lance economy is, in many ways, already upon us. . . . We see it in the emergence of virtual companies, in the rise of outsourcing and telecommuting, and in the proliferation of freelance and temporary workers (Malone & Laubacher, 1998: 146).

In considering how well the OSS movement fits our definition of a virtual organization, it is important to distinguish this concept of a virtual ‘organization’ – in its broadest sense – from other related, but more narrowly defined instances of virtuality, such as virtual corporations, virtual teams and virtual work or workplaces. [A virtual corporation is a specific type of virtual organization, one that is a profit-making corporate entity. A virtual team is a group of distributed individuals who collaborate on a particular objective, perhaps for a finite period of time. A virtual team may be a subset of a virtual organization; conversely, a virtual organization may be composed of many virtual teams. Lastly, virtual work or virtual workplaces focus on the practice of using IT and telecommuting to perform one’s job duties, such as communicating with coworkers, planning activities and writing reports.] Hereafter, I assume this fit between the OSS movement and this definition of a virtual organization, and I evaluate the context and processes that allow the agents (i.e. software developers) who participate in OSS projects to coordinate their work effectively.

If we are to examine how members of the OSS can function effectively as distributed networks of agents, we must first have a clear sense of what group effectiveness means. I borrow from the work of psychologist Richard Hackman, who developed a three-pronged model of group behaviour. In analysing various teams in ‘Groups that Work (and Those That Don’t)’, Hackman (1990) argues that there is more to group effectiveness than one-dimensional measures of productivity, or simply whether a specific result was completed on time. Hackman’s broader metric of group performance takes into account:

- the group’s production of a high-quality ‘product’ that is acceptable to those who receive or review it (whether it is a physical product, a decision, a plan or other output);
- the group’s contribution to the psychological well-being and growth of its members, allowing members to learn new things and help their personal needs to be satisfied;
- group cohesion – the continuing capability of members to work together in the future by not ‘burning out’ or encountering high levels of conflict.

Thus, to understand effective group performance, we must look further than the group’s success in delivering its specific product, to consider also whether the group meets the ‘growth needs’ of its members and if they maintain the ability to work together in the future. More
broadly, Hackman's (1987, 1990) research shows that both tasks and organizational processes for groups must be designed in specific ways to produce effective outcomes. Hackman's model of group effectiveness (Figure 1) has been widely tested and validated using groups as diverse as hospital surgical teams (Edmondson, 1996), factory workers (Safizadeh, 1991), airline flight crews (Hackman, 1990) and mental health care teams in psychiatric hospitals (Sundstrom et al., 1990; Vinokur-Kaplan, 1995).

I believe that the OSS movement is a type of virtual organization and that OSS projects support a set of practices that enable effective virtual team performance. In the following section, I show that the principles of ‘McDonaldization’ provide the foundation of control that can ensure effective performance in the absence of trust.

THE DIMENSIONS OF McDONALDIZATION

Sociologist George Ritzer developed a critique of modern society and its emphasis on standardization, productivity and efficiency. Labelling his framework the ‘McDonaldization of
society’, Ritzer (1996) draws from Max Weber’s views of rationalization, identifying four attributes of modern organizations: efficiency, calculability, predictability and control. As the name implies, Ritzer developed his McDonaldization framework by analysing fast food restaurants (notably McDonalds) and other franchise organizations that ‘commoditize’ and ‘rationalize’ their processes and outputs. Below I define these dimensions and then use them to describe the process by which OSS project teams coordinate their efforts.

Efficiency refers to ‘choosing a means to reach a specific end rapidly, with the least amount of cost or effort’ (Keel, 1998). Firms frequently equate efficiency with speed of responsiveness to customers. Calculability refers to emphasizing ‘things that can be calculated, counted, quantified . . . and a tendency to emphasize quantity, rather than quality’ (Ritzer, 1996: 142). The logic is that, if tasks and outcomes can be quantified, measured and increased, they can serve as proxies for high-quality performance. Predictability means striving for consistency of processes and outcomes – or, as Keel (1998) notes, ‘the attempt to structure our environment so that surprise and “differentness” do not encroach upon our sensibilities’. Lastly, control is similarly concerned with removing uncertainty from processes and outcomes, although it is broader in nature (compared with predictability), as it includes many different ways by which uncertainty may be reduced.

Some graphic examples of the principles of McDonaldization are the techniques used by McDonald’s (Ritzer, 1996) and Mrs Field Cookies (Cash & Ostrofsky, 1989) to standardize procedures and specify behaviours for their employees, often by substituting technology and precise measurement systems for human judgement. Ritzer (1996) even argues that fast food restaurants, such as McDonalds, seek to control their customers by the physical design of their restaurants and norms of behaviour. Furthermore, Zuboff (1988) and Garson (1989) have shown how similar controls are often used to dehumanize work in manufacturing and service sectors jobs respectively.

Although McDonaldization was developed as a critique of modern institutions, Ritzer (1996) did not extend his arguments to virtual organizations. Despite the origins of McDonaldization as a social critique, I believe that the four dimensions are value neutral, and not inherently negative. Moreover, I argue that these four dimensions may apply even more strongly to virtual organizations. Tremendous advances in IT have facilitated both the emergence of virtual organizations and also the ability to use the McDonaldization principles as a means of structuring human behaviour. Without IT-enabled processes to communicate, coordinate and control activities across organizational boundaries, virtual organizations would not be possible on the scale at which they exist (Malone et al., 1987; Benjamin & Wigand, 1995).

RESEARCH FRAMEWORK

My conceptual model builds on earlier work by Das & Teng (1998), who also renounced the conventional wisdom regarding trust as critical to organizations. In their scheme, shown in Figure 2, trust between agents is one pathway that can lead to confidence that desired out-
comes will occur, but it is not the only such pathway. Das & Teng (1998) argue that control is an alternative mechanism that may help to ensure confidence in cooperative behaviour. They refer to ‘control mechanisms’ and ‘trust building’ as generic policies or activities that may engender high levels of control or trust respectively. Consistent with this position, they argue that trust is not a necessary condition for successful cooperation among members of an alliance. Although their framework mentions ‘control mechanisms’ (rules or norms that ensure control) and ‘trust building’ (rules or norms that ensure trust), their study does not examine empirically how these actually operate in practice. Das & Teng (1998) also argue that the level of control has a direct influence on the level of trust within an alliance, whereas trust building does not influence the level of control directly, but rather moderates the relationship between the level of control and confidence in the partner’s performance.

Figure 3 elaborates upon Das & Teng’s (1998) framework by integrating it with the precepts of McDonaldization and with Hackman’s (1990) model of group effectiveness. The four principles of McDonaldization (efficiency, predictability, calculability and control) represent a set of techniques that may be used to ensure desirable outcomes, in terms of behaviour (the work process) or deliverables (the work product). Moreover, on account of the opportunities that IT provides to implement these principles, I believe that successful, high-performing virtual organizations are possible in the absence of trust. As shown by the alternative pathways to effective performance in Figure 3, trust among autonomous agents (or between a firm and its outsourcing partners) is not a necessary condition for successful performance; instead, the four dimensions of McDonaldization may serve as substitutes to obviate the need for trust. Below, I examine the attributes and practices of the OSS movement to evaluate my claim that employing the four attributes of McDonaldization may obviate the need for trust.
CONTENT ANALYSIS OF OSS CASE STUDIES

Research methodology

To evaluate whether the research framework (Figure 3) adequately characterizes one type of virtual organization – OSS projects – I employ secondary analysis of published case studies of OSS projects. I used several search methods to identify such case studies about OSS projects. First, I searched the electronic archives of both ACM and IEEE, using terms such as ‘open source.’ [The Association for Computing Machinery (ACM) and Institute of Electronics and Electrical Engineers (IEEE) are two of the leading professional societies in the IT field. Each society publishes several dozen academic and practitioner-oriented journals and magazines.] Secondly, I searched on Bell & Howell/Proquest’s ABI/Inform, a database of academic and trade publications on business and management (including technology management). Thirdly, I reviewed the ‘position papers’ of the various authors who attended the 1st Workshop on Open-Source Software Engineering (Feller et al., 2001) for relevant case studies. Fourthly, I searched on Harvard Publishing’s websites, which contain case studies published by two leading North American case research schools of business, Harvard Business School.
and University of Western Ontario, Canada. Finally, as I located case studies or other publications about OSS projects, I followed references from them to identify other case studies. Although I found literally hundreds of publications about OSS development, there was a much smaller number of case studies. To fit these selection criteria, the publication needed to describe the process of OSS development, in general, or one or more specific OSS projects (e.g. Linux, Apache, Perl). Secondly, the paper must contain original data and analysis – that is, the authors must have observed or participated in an open source project or, alternatively, they must have had access to archival data about one or more open source projects that they analysed. These criteria eliminated over a dozen studies that discussed OSS projects in general terms, but did not collect or analyse data of their own. Note that, in order to avoid selection bias, I did not explicitly screen the papers on the basis of whether they mentioned trust or control.

From the hundreds of publications about open source development, I identified nine papers that met the above criteria as OSS project case studies. Each case study was reviewed thoroughly three times (in iterative fashion) to identify specific statements that embody each of the themes. For data analysis, I employed content analysis, that is techniques for making replicable and valid inferences from data to their context. I used a traditional form of content analysis, whereby I approached the data with a predefined set of content variables and searched for passages that embodied these themes (Carney, 1972; Andren, 1981).

Results

Brief descriptive data about each study are summarized in Table 1. In general, this section will identify some overall patterns among the nine studies, rather than analyse passages from the original source material in detail. I identify first some commonalities among the context in which these case studies were conducted, before addressing their specific content.

The context of OSS case studies

The first general observation regarding the context of the case studies is that, with the exception of Raymond’s (1996, 1998) seminal essays on OSS development, nearly all the other studies are very recent – that is published during 2000–2001. [Several of the case studies are still in a working paper version and are undergoing journal review at this time.] Secondly, several of the case studies have, as their primary focus, either the Linux project (four studies) or the Apache project (three studies). This may be because Linux and Apache have achieved the most commercial success in terms of their acceptance and market share. Thirdly, more than half the case studies were written by people who were closely involved in the project – usually as project leaders or developers (e.g. Raymond, 1996, 1998; Fielding, 1999; Mockus et al., 2000; Aoki et al., 2001). Less than half the studies were by researchers who lacked direct involvement in the projects as developers (Lakhani & von Hippel, 2000; Lee & Cole, 2000; Markus et al., 2000; Moon & Sproull, 2000). Of these studies by non-OSS
Table 1. Case studies of open source software development.

<table>
<thead>
<tr>
<th>Author(s)</th>
<th>Title of study</th>
<th>Publ. date</th>
<th>Published source</th>
<th>Context of OSS study</th>
<th>Research methods used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fielding</td>
<td>Shared leadership in the Apache project</td>
<td>1999</td>
<td>Communications of ACM</td>
<td>Apache</td>
<td>Participant (developer/researcher)</td>
</tr>
<tr>
<td>Lakhani &amp; von Hippel</td>
<td>How open source software works: free user-to-user assistance</td>
<td>2000</td>
<td>MIT Sloan School of Management working paper (published online)</td>
<td>Apache (technical/help support)</td>
<td>Multiple methods: (1) archival analysis of usenet help data; (2) online survey of ‘seekers’ and ‘providers’ on Apache newsgroups</td>
</tr>
<tr>
<td>Lee &amp; Cole</td>
<td>Linux kernel development as a model of open source knowledge creation</td>
<td>2000</td>
<td>University of California, Berkeley, Haas School of Business working paper (published online)</td>
<td>Linux kernel</td>
<td>Multiple methods: (1) archival analysis of Linux kernel mailing list 1995–2000; (2) second-hand analysis of survey data (based on University of Kiel survey)</td>
</tr>
<tr>
<td>Markus et al.</td>
<td>What makes a virtual organization work?</td>
<td>2000</td>
<td>Sloan Management Review</td>
<td>Miscellaneous OSS Projects: Apache; GNU/Linux; Perl; Mozilla</td>
<td>Secondary analysis of published sources</td>
</tr>
<tr>
<td>Moon &amp; Sproull</td>
<td>Essence of distributed work: the case of the Linux kernel</td>
<td>2000</td>
<td>New York University Stern School of Business working paper (published online)</td>
<td>Linux kernel</td>
<td>Archival analysis of Linux archives</td>
</tr>
<tr>
<td>Raymond</td>
<td>The Cathedral and the Bazaar</td>
<td>1996</td>
<td>Online essay later republished in <em>The Cathedral and the Bazaar</em> (book)</td>
<td>Linux</td>
<td>Participant (developer/essayist)</td>
</tr>
<tr>
<td>Raymond</td>
<td>Homesteading the Noosphere</td>
<td>1998</td>
<td>Online essay later republished in <em>The Cathedral and the Bazaar</em> (book)</td>
<td>Fetchmail</td>
<td>Participant (developer/essayist)</td>
</tr>
</tbody>
</table>
developers, it is noteworthy that the subject of coordination in OSS projects has attracted the interest of several leading scholars from research domains as diverse as product innovation (Eric von Hippel), electronic communities (Lee Sproull), manufacturing process improvement (Robert Cole) and the management of IT in organizations (Lynne Markus). Fourthly, nearly all the case studies address issues of development, testing and releasing OSS software code, with only one exception – a case study of information ‘seekers’ and ‘providers’ in the context of an online newsgroup for Apache technical support (Lakhani & von Hippel, 2000). Fifthly, although the research methods used for collecting data varied widely, many relied on archival data about OSS projects, which are stored as a matter of protocol and, in most cases, are available to the general public online. The other common research method, as noted in Table 1, is the ‘participant’ method, as many authors of the case studies were involved in actual project management or software development for these projects, and then wrote about them retrospectively.

Trust in OSS projects

Appendix A contains the detailed results of the content analysis, showing the relevant, coded passages for each of the five constructs discussed above. I have used direct quotes from the original document in these results, paraphrasing only where necessary to make explicit some information assumed in the original source material or to summarize material that would have been too lengthy to quote directly in the table. This table is structured first by the source and, within each source, according to the five key themes (efficiency, predictability, calculability, control and trust). Although there is not room for a thorough discussion of the themes that I coded from these sources here, I identify a few overall highlights below.

The first observation regarding the content analysis was that there were relatively few references to trust in the nine case studies. Five of the nine studies lacked any reference to trust at all, whereas the other four studies included only a few references to trust. Altogether, the coded passages referring to trust account for a very small fraction of the overall coded passages (about 4.2%). Based on the sheer proportion of coded text in Appendix A – and the fact that five of the nine studies made no mention of trust – it appears that either trust is relatively unimportant in OSS projects or its existence remains unacknowledged and taken for granted. The latter possibility is considered in the Discussion section below. Of the few explicit references to trust in the case studies, some authors referred to trust among the contributing software developers who submitted their code, whereas other authors described the trust that these contributing developers had in the wisdom and fairness of project leaders who have control over a given OSS project. Project leaders have authority for setting ground rules for each OSS project and for deciding which code submissions to accept, modify or reject. For example, one reference to trust that developers hold for the leaders of an OSS project appeared in the case study of Jun (Aoki et al., 2001), under the heading ‘The role of a project leader’ [Any words appearing in boldface are for the purposes of this analysis and did not appear in boldface in the case study]:
One fear in open-source development is that *too many cooks ruin the soup*. Trust and dependability are important aspects in the growth of an open-source project. Potential contributors must believe in the long term direction and quality of leadership. The project leader of the Jun development team has been responsible for deciding which submissions to incorporate and which projects to accept. People trust his decisions. The promise of open-source software is not just access to the source code, but trust in the human leadership (Aoki *et al.*, 2000: 532).

A second type of trust that appeared is the trust that an OSS developer has in the broader community of users who test the software code, identify possible problems and submit bug reports:

**Trusting** the judgment of others is basic to the peer-review process. It’s necessary because nobody has time to review all possible alternatives. So work *used by lots of people* is considered better than work used by a few (Raymond, 1998: 117).

A third type of trust was the trust that developers, who contribute their ‘intellectual property’ to OSS projects, have for the users of these products, who will benefit directly from their labours. In analysing the nature of trust within the Linux development community, Lee & Cole (2000) articulated this theme:

What are the licensing mechanisms that allow developers to trust a community of strangers with their intellectual properties? . . . One of the [Linux] community protocols is a strong norm to properly cite authors whose work is being extended or borrowed. Such protocol shows how much this community respects the contributions and intellectual properties of its members. . . . The requirement to make any modification to the source code publicly available creates the foundation under which developers can trust others, whom they have never met, with their intellectual properties. . . . Since other developers who modify ‘copylefted’ source code are required to return the changes to the public, the likelihood of misappropriation [of one’s contribution] is significantly reduced (Lee & Cole, 2000: 12, 14, 17).

Although some references to trust were ambiguous, in terms of not specifying the role of either the truster or the trustee, even within the small number of references to trust (five total passages), we can observe a range of scenarios – in terms of the identity of the truster and the trustee. This diversity in terms of the identity of the truster and trustee occurs because there are multiple roles in each OSS project. I describe and analyse these roles in the next section.

*Multiple roles in OSS projects*

In addition to the large user community who will benefit from the opportunity to use OSS products (e.g. the Apache server), there are at least three specific roles played by individuals in these projects. These roles include: (1) project leaders who have overall responsibility for an
OSS project; (2) developers who create and submit code for an OSS project, including those whose contributions range from a low level to a high level of involvement; and (3) other individuals who, in the course of using the software, perform testing, identify bugs and submit problem reports, although they may not be developers themselves. Most of the case studies identified these multiple roles within each OSS project, although the authors used a variety of terms to describe these roles. Several studies referred to the distinction between the ‘core developers’ – usually a small group of individuals with significant responsibility and control over a project – vs. a much larger group of ‘peripheral developers’.

For example, the role differentiation between ‘core’ and ‘peripheral’ developers was noted for both the Linux (Lee & Cole 2000) and Apache projects (Mockus et al., 2000; Asundi, 2001). Other synonyms for the ‘core developers’ were the ‘core group’ (Fielding, 1999) and the ‘inner circle’ (Koch & Schneider, 2000). Although the various projects differed, in terms of whether there was a single project leader with overall authority, such as Linus Torvalds for Linux (Raymond, 1998), vs. a small group of controlling members, such as the ‘core group’ for Apache (Fielding, 1999), all projects seemed to recognize the distinction between certain developers who had leadership responsibility, others who contributed heavily to the project but lacked decision authority and still others who were occasional contributors. Linux featured two additional types of developer roles, maintainers and credited developers (Lee & Cole, 2000), as well as a distinction between coders and posters – the latter referring to individuals who frequent newsgroups and ‘participate in discussions pertaining to the software’ (Koch & Schneider, 2000), but who may not program.

By incorporating some of this terminology regarding differentiated roles, it is possible to observe the various structural patterns of trust that can potentially take place within a given OSS project. Figure 4 depicts the three major groups of participants in an OSS project (core developers, peripheral developers and message posters) and identifies several types of trust that occur among these groups. Note that Figure 4 depicts the individuals comprising each group as a set of bullet points and shows all trust relationships as bidirectional arrows, indicating that trust is presumed to be reciprocal. I identify five structural possibilities for trust that may occur in OSS projects – essentially every possible permutation of the three roles (except trust among the message posters). Note that the boundary of the core group appears as a dotted line, because this boundary may be permeable for some projects. In addition, although the arrows are drawn between the individual bullet points (i.e. the individuals who occupy each of the roles), this should be interpreted to denote trust in the anonymous ‘others’ who occupy each of these roles as, in most cases, the individuals will not know each other personally. [An exception to this may occur among the core developers, as these members may know each other personally.] For example, arrow ‘B’ indicates the trust that one peripheral developer has for other peripheral developers (in general), rather than his or her trust in a specific developer.

Control in OSS projects

Turning now to the coded passages in Appendix A representing the principles of McDonaldization, it is apparent that there are far more passages denoting the ‘control’ principle than there
are for the other three principles combined (efficiency, predictability and calculability). In fact, over 85% of the coded text refers to ‘control’, with only 10% of the coded text referring to the other three McDonaldization themes. One possible conclusion may be that specific practices for achieving efficiency, predictability and calculability are not widespread in OSS projects. Another interpretation, however, is that ensuring control is a high-level objective within each project, and that the other three dimensions of McDonaldization (efficiency, predictability and calculability) represent lower level objectives or specific tactics to achieve control. As there were relatively few references to efficiency, predictability or calculability in the nine case studies, it may be that this set of tactics is not an exhaustive list of means of achieving control, and that other tactics are possible – and employed more frequently than the three tactics that appear in Figure 3. It is also important to acknowledge that it is possible to interpret some of the coded passages as representing more than one of the McDonaldization themes – for example, control and efficiency, or control and predictability. Thus, the actual number of coded passages for each of the McDonaldization themes should be interpreted as only an approximate indication of the frequency with which each theme appeared, rather than as absolute patterns. The possibility for double-coding certain passages (e.g. interpreting a passage as signifying control and predictability) is consistent with the prior insight that control is a higher level objective, whereas efficiency, predictability and calculability are specific subgoals.

Although practices to ensure control through these other principles of McDonaldization have been noted in case studies of interorganizational alliances, for example at Ford Motor
Company (Austin, 1999) and Sun Microsystems (Holloway et al., 1996), techniques such as efficiency, predictability and control may not be the mechanisms used most commonly in OSS projects to ensure control. Instead, a quick perusal of the coded passages in the ‘control’ section of Appendix A suggests that there are other ways of ensuring control. Examples of other control-oriented tactics include behavioural norms and member voting (both forms of social control) or the emphasis placed on the individual’s professional reputation (a form of self-control).

OSS projects may indeed differ from other types of virtual organizations in the importance placed on social control and self-control. Markus et al. (2000) should be credited with this observation, given their detailed and insightful analysis of how these two general classes of control (social control and self-control) are widely used in OSS projects. Specifically, they identify four groups of control mechanisms that are used in OSS projects: the importance of individual reputation, ‘membership management’ (i.e. regulating who may participate), rules and institutions and, finally, sanctions (or the threat of sanctions) against undesirable behaviours. Markus et al. (2000) emphasize how effective, in aggregate, the ‘governance mechanisms’ of social control and self-control can be, despite what may appear to be a software development process that is subject to anarchy [again, boldface has been added to highlight terms referring to control or its synonyms]:

The open-source movement exhibits many . . . mechanisms and processes ensuring that order reigns. . . . Despite the clear potential for chaos, open-source projects are often surprisingly disciplined and successful through the action of multiple, interacting governance mechanisms. Membership management, rules and institutions, monitoring and sanctions, and reputation build on the precondition of shared culture to self-regulate open-source projects. . . . Although the importance of organizational culture is often stressed in traditional organizations, much less emphasis has been placed on building self-control (to preserve one’s reputation) and social control (for peers to monitor and sanction others’ behavior) into formal organizational control systems (Markus et al., 2000: 24, 14–15, 25).

Figure 5 depicts the patterns of control in OSS projects, with each arrow specifically noting both the source and the targets of control. Although Figure 5 (structural patterns of control) is analogous to Figure 4 (structural patterns of trust), the critical difference is that, in Figure 5, some arrows are unidirectional (representing one-way control), rather than bidirectional (representing reciprocal control). For example, the one-way arrows show that the direction of control is from the core developers to the peripheral developers or, alternatively, from the core developers to the message posters. I characterize these as one-way controls because the core developers have the authority to set the policies and, in any specific situation, to decide whether to accept, modify or reject code submissions from other developers. [Figure 5 depicts the relationship between peripheral developers and posters with dotted lines, as it is unclear whether these groups exert control over each other and, if so, which direction of control predominates over the other.] Moon & Sproull (2000) elaborated on the control patterns within the Linux community:
Because the Linux-kernel mailing list is the central organizing forum for kernel developers and supports a very heavy volume of message traffic, people need a clear understanding of how to contribute to the list. Maintainers of the Linux-kernel mailing list FAQ (frequently asked questions) create that understanding. The FAQ is the document that explains the rules of the road for kernel development to newcomers. It also reiterates the norms and values of the community (Moon & Sproull, 2000: 14).

Furthermore, violating these norms can result in various sanctions, as described by Lee & Cole (2000):

The Linux development community has documented in the Linux-kernel mailing list FAQ on the web and in the 'maintainers' file many suggestions on how to submit patches and report bugs. . . . The consequence of rule violation is reduced likelihood that the violator's contribution will be taken seriously or even noticed . . . (Lee & Cole, 2000: 30–31).

Behavioural norms have value only if there is a credible threat of sanction and, much like one of the popular ‘reality shows’ on television, OSS projects have various ways of sanctioning members' behaviour, such as 'voting a member out', reducing a member’s privileges or not allowing them to be 'voted in' to begin with. These are all examples of the principle of 'membership management'.

Legend:
1: Control among 
2: Control among Peripheral Developers 
3: Control of Core Developers over Peripheral Developers 
4: Control of Core Developer Group over “Posters” 
5: Control between Peripheral Developers and “Posters”

Figure 5. Structural patterns of control in open source software projects.
DISCUSSION

Based on the coded passages contained in Appendix A and on the structural patterns of trust and control depicted in Figures 4 and 5, it appears that trust and control may operate independently of each other to shape the behaviour of individual contributors, as well as to determine the overall effectiveness of OSS projects. In addition, given the much greater number of references to control than to trust in the nine case studies, I argue that the OSS movement appears to rely on explicit forms of control to a much greater degree than on trust. Given this observation, there are two possible explanations. The first possibility is that trust may be implicit and unacknowledged in many of the OSS project activities, and this is why it does not appear explicitly in most authors’ retrospective descriptions of OSS projects. Instead, the authors of these case studies may simply take for granted that trust operates tacitly in many forms.

The other possible explanation is that most OSS projects deliberately avoid relying on trust because this can make them vulnerable to members’ misdeeds. Instead, they employ control through a variety of techniques – some explicit and others relatively unobtrusive. For example, given the relative frequency with which control and trust appear in Appendix A, it may be the case that OSS projects use trust only as a last resort when effective controls do not exist or when control measures can undermine working relationships among project members. This latter possibility – avoiding too much control (or certain forms of control) because it may damage relationships – is consistent with the literature, particularly as excessive control can generate ‘conflict and distrust’ (Harrison & St John, 1996), as well as other ‘unintended human and social consequences’ (Markus, 2000).

There exists some limited evidence for each of these possible explanations (i.e. that trust is widespread but unacknowledged vs. that trust is deliberately avoided unless there is no alternative); however, a clearer determination requires further research into this issue. In acknowledging the need for additional research to understand which explanation for the paucity of references to trust is valid, it is also important to recognize that, although Figure 5 shows the structural patterns of control that can occur among the three role types, further investigation is needed to identify which specific control mechanisms are practised within and between these groups.

This review of nine OSS case studies concludes that many forms of control are practised in OSS projects – some explicit (rules and norms stated in the FAQ, for example) and others implicit (the emphasis on individual reputation). Among the nine case studies of OSS projects reviewed here, there was an overwhelming number of references to control, but fewer references to the three other principles of McDonaldization (efficiency, calculability and predictability) and even fewer explicit references to trust. These findings contradict the widely assumed value of trust as a necessary condition for successful virtual organizations, partnerships and other exchange relationships.

Although these findings contradict the conventional wisdom about trust, they are supported by other research on trust, notably Das & Teng’s (1998) research, which concluded that either
‘trust building’ or ‘control mechanisms’ can ensure confidence in another party’s behaviour. While the conceptual framework proposed in this paper (Figure 3) leverages Das & Teng’s (1998) work, the principal difference is that I specify the principles of McDonaldization as a set of control techniques, as well as identifying a set of group effectiveness metrics (Hackman, 1990). In contrast, Das & Teng (1998) proposed a different set of ‘control mechanisms’ that can be used between the agents in a virtual organization or alliance, including: (1) goal setting (establishing specific and challenging goals); (2) structural specifications (ex ante deterrents and ex post deterrents); and (3) cultural blending (blending and harmonizing values and beliefs across multiple firms or individuals). Beyond the general level of agreement between the ideas proposed here and the contributions of Das & Teng (1998), my arguments are also consistent with research in a variety of other domains, including the principles of control theory from organizational sociology (Eisenhardt, 1985; Ouchi, 1979, 1980; O’Reilly, 1996) and agency theory from new institutional economics (Eisenhardt, 1989). Through my integration of diverse literatures including sociology (Ritzer, 1996), social psychology (Hackman, 1987, 1990), and field work on OSS projects, this paper makes an important contribution to understanding why trust may not appear frequently in descriptive summaries of OSS projects. My primary conclusion is that high levels of trust are not necessarily required for collaborative software development, as practised by OSS project teams, or in other types of virtual organizations.

The insights from this study have implications beyond explaining the forces that help to make OSS effective and successful. Many have argued that virtual, networked or modular organizations are the models for the future (Nohria & Eccles, 1992; Lucas, 1996; Sanchez & Mahoney, 1996) and that most large organizations will increasingly make use of market-like mechanisms, employing temporary alliances or project teams to achieve a specific goal. Given this greater prevalence of temporary alliances as a means of accomplishing work – our definition of a virtual organization – I believe that the OSS movement is a fascinating crucible for observation and analysis, and that many of these insights can be generalized to other settings. One interesting illustration of this is how the concept of loosely structured teams of high-talent individuals sharing information has been generalized from the practice of software development to that of stock market investing. In reviewing a recent book, entitled Change Wave Investing (Smith, 2000), which introduces a new type of ‘open source investing’, one journalist claims that, by copying the ethos and techniques of OSS development teams, this new paradigm of educated investors pooling their stock market information may threaten the livelihood of traditional market makers (Schmerken, 2000).

Although it is important to realize that there are certain attributes of OSS project teams that distinguish them as instances of a particular type of virtual organization (e.g. the fact that most contributors are highly educated volunteers), nevertheless, I believe that certain insights can indeed be generalized to other settings. As evidence of this generalizability, I cite the rationale used by Markus et al. (2000) as their justification for why general managers should pay heed to the outcomes and insights provided by OSS project teams. In their aptly named article, ‘What makes a virtual organization work?’, Markus et al. (2000) argued that the OSS movement is an exemplar of a virtual organization and a template from which tomorrow’s organizational leaders can learn:
In short, there is a relatively high degree of correspondence between the open-source movement and popular depictions of the organization of the future and the virtual networked organization. Therefore, the open-source movement provides some suggestions about how management must change in the years ahead. . . . Although managers in industries other than software development may prefer more traditional styles of management, they should remember that the world is changing and workers are changing along with it. In a labor force of volunteers and virtual teams, the motivational and self-governing patterns of the open-source movement may well become essential to business success (Markus et al., 2000: 25).

LIMITATIONS AND FUTURE DIRECTIONS

There are, of course, certain limitations to this work that should be acknowledged. First, I did not conduct my own field research of an OSS project but, instead, I relied on secondary data, based on the observations of other researchers or developers who participated in these projects. Admittedly, these authors had other objectives in writing up their work, and they may have overlooked behaviours related to trust and control, simply because it was not their objective to address these issues. Secondly, this study is based on a relatively small number of case studies of OSS projects, and a significant number of them focused on one of two projects that are receiving considerable attention in both the academic press and the popular media (Linux and Apache) (Moody, 2001). Thirdly, not all OSS projects are the same. It is a common misconception among the general public that the term OSS describes a particular and single set of processes by which software is developed. On the contrary, the term OSS really applies to the software product itself – that is to say, the licensing of the resulting software must abide by the terms of the General Public License (Raymond, 1998), yet there can be a range of different processes used to create the product (Fitzgerald & Feller, 2000).

Nakakoji & Yamamoto (2001) have suggested recently that there may be as many different processes for creating OSS products as there are primary objectives that different OSS products seek to achieve. Their taxonomy identifies three different categories of OSS products (security, archetype and rapidness products), each of which has a different primary objective associated with it (to achieve rapid recovery, to produce reference models and to facilitate timely development respectively). Finally, there were many other case studies and conceptual analyses of OSS projects that appeared in my literature search, yet which I omitted from the content analysis. Among the many reasons for excluding certain OSS studies from the content analysis were: (1) some studies were still in progress and results were not available at the time this special issue of Information Systems Journal went to press (Asundi, 2001; Nakakoji & Yamamoto, 2001; Scacchi, 2001; Schmidt & Porter, 2001); (2) they did not collect or analyse any primary source data, but were conceptual in nature (Malone & Laubacher, 1998; Cook, 2001); (3) they examined the psychological motivations of individuals who contributed to OSS projects (Hars & Ou, 2001), without addressing the coordination processes that developers and project leaders employed; or (4) they were ‘case studies’ of the resulting software
artifacts themselves, rather than studies of the work practices that were employed to create them (Godfrey & Tu, 2000; Koch & Schneider, 2000).

Clearly, more research is required into the different ways that trust and control may be employed in virtual organizations. Despite attempting to forge new ground conceptually, this paper adheres to the rather conventional notion that trust and control are opposites or substitutes for each other. It may also be instructive to consider a very different possibility, namely that trust and control may be synergistic, which might lead to novel questions about how their presence (or absence) may reinforce each other over time. This insight was offered recently by O’Leary et al. (2001), who suggested that trust and control are mutually interdependent in ‘distributed’ or virtual organizations. In their study of one 350-year-old distributed organization, the London-based Hudson’s Bay Company retailer, O’Leary et al. (2001) argue that:

Trust and control have typically been viewed as opposites or substitutes, and this is especially true in publications regarding distributed work. . . . We argue that trust and control are closely intertwined and often mutually reinforcing approaches to managing distributed work. . . . Our investigation of the Hudson’s Bay Company suggests that at least three practices – socialization, communication, and participation – were particularly important sources of trust and control (O’Leary et al., 2001: 3, 27).

There are many interesting questions raised by this study and, in future work, I hope to be able to fill in many of the details that are merely suggested by the broad outlines of the structural patterns of trust (Figure 4) and control (Figure 5). For example, it would be helpful to understand which types of controls or trust-building techniques are used by the various groups, or whether the relationships that are presumed to be reciprocal (in Figure 4) or unidirectional (in Figure 5) indeed operate in the indicated fashion. It would be interesting to investigate whether OSS developers who coordinate their activities ‘virtually’ produce solutions that are more creative and of higher quality, compared with software developers working face to face – a possibility that has been suggested by experimental work with student teams (Ocker et al., 1995, 1998; Ocker & Fjermestad, 1998), but not yet validated in the field. Finally, it may also be worthwhile to consider the unconventional hypothesis that trust and control are ‘inter-twined with and mutually reinforcing’ each other (O’Leary et al., 2001), rather than substitute ways of achieving the same objectives. Such a possibility would require a longitudinal analysis of one or more OSS projects, in which a focus on the dynamic patterns of trust and control, including their influence on each other, would frame the field study’s data collection and analysis from its inception.

SUPPLEMENTARY MATERIAL

The following material is available from http://www.blackwell-science.com/products/journals/suppmat/isj/isj108/isj108sm.htm

Appendix A: Content analysis of open source software case studies.
REFERENCES


Biography

Michael J. Gallivan is an Assistant Professor in the CIS Department at Georgia State University Robinson College of Business in Atlanta, USA. Dr Gallivan conducts research on human resource practices for managing IT professionals, as well as strategies for managing effective IT implementation, IT outsourcing and strategic alliances. Michael received his PhD from MIT Sloan School of Management and served on the faculty at New York University Stern School of Business before his present appointment at Georgia State. He has published in IEEE Transactions on Professional Communication, Data Base for Advances in Information Systems and Hawaii International Conference on System Sciences (HICSS).