MIS Quarterly

SPECIAL ISSUE

OUTSOURCING TO AN UNKNOWN WORKFORCE: EXPLORING OPENSOURCING AS A GLOBAL SOURCING STRATEGY[1]

By: **Pär J. Ågerfalk**[†]
Department of Information Science
Computer and Systems Science
Uppsala University
Uppsala
SWEDEN
par.agerfalk@dis.uu.se

Brian Fitzgerald
Lero – The Irish Software Engineering Research Centre
University of Limerick
Limerick
IRELAND
bf@ul.ie

Abstract

This paper presents a psychological contract perspective on the use of the open source development model as a global sourcing strategy—opensourcing, as we term it here—whereby commercial companies and open source communities collaborate on development of software of commercial interest to the company. Building on previous research on information systems outsourcing, a theoretical framework for exploring the opensourcing phenomenon is derived. The first phase of the research concerned qualitative case studies involving three commercial organizations (IONA Technologies, Philips Medical Systems, and Telefonica) that had "liberated" what had hitherto been proprietary software and sought to grow a global open source community around their product. We followed this with a large-scale survey involving additional exemplars of the phenomenon. The study identifies a number of symmetrical and complementary customer and community obligations that are associated with opensourcing success. We also identify a number of tension points on which customer and community perceptions tend to vary. Overall the key watchwords for opensourcing are openness, trust, tact, professionalism, transparency, and complementariness: The customer and community need to establish a trusted partnership of shared responsibility in building an overall opensourcing ecosystem. The study reveals an ongoing shift from OSS as a community of individual developers to OSS as a community of commercial organizations, primarily small to medium-sized enterprises. It also reveals that opensourcing provides ample opportunity for companies to headhunt top developers, hence moving from outsourcing to a largely unknown OSS workforce toward recruitment of developers from a global open source community whose talents have become known as a result of the opensourcing experience.

Keywords: Open source, opensourcing, offshoring, outsourcing, global software development, crowdsourcing, multimethod research

Introduction

In recent years, the open source software (OSS) development model has gained significant momentum and is now generally

[1]This paper was recommended for acceptance by Associate Guest Editor Rudy Hirschheim.

[†]Senior Research Fellow in Lero – The Irish Software Engineering Research Centre.

considered a viable approach also in commercial settings. Similar to outsourcing, and particularly offshore sourcing, the OSS development model promises many advantages, including reduced salary costs; reduced cycle time arising from "follow-the-sun" software development; cross-site modularization of development work; access to a larger skilled developer pool; innovation and shared best practice; and closer proximity to customers (Carmel 1999, 2006; Carmel and Agarwal 2001; Carmel and Tjia 2005; Ebert and De Neve 2001; Herbsleb and Grinter 1999). Given that the primary force driving offshore sourcing appears to be cost savings, it is perhaps natural that companies might eventually focus on the OSS development model as a potentially even-cheaper alternative, as there are significant cost savings associated with the acquisition of OSS (Wheeler 2004). Carmel and Tjia (2005) have characterized offshore sourcing as "outsourcing to a global workforce." In this study, we investigate the emerging trend toward opensourcing; that is, outsourcing to a global but largely unknown workforce of OSS developers.

Many different theoretical frameworks have been used to study outsourcing and offshoring. Three of the more popular are agency theory (Aubert et al. 2005; Cheon et al. 1995), relational exchange theory (Cheon et al. 1995; Goles and Chin 2005) and transaction cost theory (Aubert et al. 2005; Cheon et al. 1995; Grover et al. 2002; Wang 2002). While these could potentially provide candidate theoretical frameworks for our study, we saw shortcomings with each. For example, transaction cost theory and agency theory are heavily oriented toward a traditional economic perspective (Goles and Chin 2005), whereas the majority of OSS development takes place outside the realm of traditional economic theories. A further assumption of agency theory is that the agent (vendor) and the principal (company) are able to share common goals and accept the same degree of risk willingness and aversion (Gottshalk and Solli-Sæther 2005), which is probably hard to establish given the "unknown" factor inherent in opensourcing. While relational exchange theory does recognize the importance of interactions, interdependencies, reciprocities, and informally negotiated rules of exchange between parties, it also builds on the assumptions that a formal contract is in place (Goles and Chin 2005) and that the company can develop and secure common norms with the vendor (Gottshalk and Solli-Sæther 2005). Neither of these assumptions is intuitively valid in an OSS context given the voluntary nature of typical OSS communities and the tension between economic value that drives commercial organizations and the community values that drive OSS (Fitzgerald 2006). In order to overcome this bias toward "traditional" long-term economic relationships, we chose to adopt the perspective suggested by Koh et al. (2004), using psychological contract theory (PCT) as a basis for understanding the mutual relationships between managers of offshore outsourcing customer organizations and members of their global OSS community. Our reason for choosing PCT was thus because of its emphasis on mutuality and reciprocity, its individual level of analysis, and its acknowledgment of the importance of unwritten commitments. As explained below, all of these three core assumptions of PCT are core also to understanding OSS. Most importantly, though, PCT does not impose *a priori* assumptions other than that people expect people to fulfill their obligations in order for their relationship to be successful. Such theoretical openness lends itself well to the exploration of an emerging phenomenon such as opensourcing.

Koh et al. note that most research on offshore outsourcing tends to focus on the customer perspective, and argue for the importance of studying both the customer and supplier side of the relationship. Interestingly, research to date on open source has focused inward on investigating the characteristics of the development process and projects, that is, on the supplier side of the relationship, and far less has been conducted on the customer side, in the sense of investigating the consequences of the OSS phenomenon for organizations, for example. Building on Koh et al.'s results, our research question is thus: ***What are the critical customer and community obligations in a successful opensourcing relationship?***

This study focuses on a phenomenon for which we have coined the term ***opensourcing***, which is the use of the OSS development model as a global sourcing strategy for an organization's software development process. One of the novel features of the study is that it considers both sides of the process—that is, the organization that is "commissioning" the opensourcing, and, in turn, the OSS community who are doing the development work. The work presented here also builds on and extends the work of Koh et al. in two significant ways. First, it considers the unexplored concept of opensourcing, as opposed to traditional domestic outsourcing. Second, Koh et al. focused on onshore outsourcing while this work considers ***offshore*** outsourcing, as implicated by the opensourcing strategy, which is premised on the global nature of open source development as confirmed in a wide range of studies on the issue and further discussed below (e.g., Dempsey et al. 2002; Ghosh et al. 2002; Lakhani and Wolf 2001; Robles-Martinez et al. 2001). To investigate this area, we adopted a two-phased research approach (see Appendix A for a schematic overview): In the first phase, we drew on the set of obligations identified and validated by Koh et al. and conducted an exploratory qualitative multiple case study approach to refine and elaborate those obligations for which the customer must bear responsibility and the obligations for which the OSS community must bear responsibility in order for the opensourcing arrangement to be successful. In the

second phase, we conducted a quantitative survey study to explore the validity of those obligations. Such a dual-phase approach, which has been used to good effect in previous research (Kaplan and Duchon 1988), allowed us to triangulate from multiple data sources, an established strategy for improving reliability and validity (Yin 1994).

The structure of the paper is as follows: In the next section, we define basic terminology to distinguish the related concepts of outsourcing, offshoring, and opensourcing. We then focus on PCT, explaining why we have selected it for this study, and how it has been used. Following this, our qualitative research approach is discussed, and the case study findings presented and analyzed. After that, the quantitative survey study is presented. Finally, the overall conclusions and implications are presented. In the final section, we also present some limitations of the study and discuss possible future directions for research on opensourcing as a global sourcing strategy.

Basic Terminology

The terms *offshoring* and *outsourcing* are often used almost as synonyms, however here, in keeping with Davis et al. (2004), we distinguish between the two: Offshoring is about location—when an activity is offshored, it is performed in a different location to the main operation (which is then the onshore location). Outsourcing, on the other hand, is about governance—when an activity is outsourced, it is performed by another organization, as opposed to in-house by the organization itself. Consequently, the two concepts are orthogonal: an activity can be performed either offshore or onshore and can be performed in-house or be outsourced. Figure 1 shows the distinction and relationship between the two concepts.

Open source software may be defined as software released under the terms of a license that basically allows the licensee to use, modify, and redistribute, either gratis or for a fee. Of particular interest in this research, however, is the recent phenomenon of opensourcing as a software development model. Similar to outsourcing, the OSS development model allows companies to subcontract development activities to another party. Since anyone (in principle) can join any open source project, the development community can reasonably be assumed to be global (Millar et al. 2005). The global nature of contributions to open source projects has been confirmed in several studies. For example, Lakhani and Wolf (2001) conducted a survey on a sample of almost 10,000 projects in SourceForge and found that 55 percent of respondents were from outside North America. In another large-scale survey of almost 3,000 developers from across a broad range of open source projects, Ghosh et al. (2002) estimate that almost 86 percent of open source developers are from outside North America. Another study, which focused specifically on Linux developers, also reports developers outside North America perform the majority of development (Dempsey et al. 2002). Another relevant study is the one conducted by Robles-Martinez et al. (2001) of a number of popular open source projects, which found that 62 percent of developers were from outside North America. This latter study is particularly noteworthy in that the authors performed a series of exhaustive checks to identify the country of origin of contributors. The global nature of the collaborative development found in all of these studies supports our classification of opensourcing as primarily an offshore outsourcing strategy (i.e., in the bottom-right quadrant of Figure 1).

It is important also to emphasize that opensourcing, as a global sourcing strategy, is not limited to releasing previously proprietary software under an open source license and nurturing a community around the product. This approach to opensourcing—as, for example, practiced by IONA Technologies who released their flagship product Artix as open source under the name Celtix—we refer to as the *liberation* approach in this study. The reverse of this approach is when a company evolves from an existing open source product, as exemplified by the MySQL database management system (subsequently acquired by Sun Microsystems) and the JBoss J2EE application server (subsequently acquired by RedHat). The latter can be referred to as the *commercialization* approach to opensourcing.

A PCT Perspective on Opensourcing

An early and influential contribution to psychological contract theory (PCT) was that of Argyris (1960). PCT has since been widely used in studies on employment relationships (Anderson and Chalk 1998) and has featured in several IS research studies (e.g., Ang and Slaughter 2001; Koh et al. 2004; Miranda and Kavan 2005; Pavlou and Gefen 2005; Piccoli and Ives 2003; Raghu et al. 2004). A psychological contract represents "the contractual parties' mental beliefs and expectations about their mutual obligations in a contractual relationship, based on perceived promises of a reciprocal exchange" (Koh et al. 2004, p. 357). Three aspects of the psychological contract are particularly important in an outsourcing context: First, the importance of mutuality and reciprocity of obligations in a social context. This is distinct from the usual one-sided perspective (customer *or* supplier) commonly adopted in outsourcing studies (Goles and Chin 2005; Koh et al. 2004).

	In-house	Outsourced
Onshore	In-house (traditional model)	Subcontractor in the same locale
Offshore	Foreign branch of the same company	Subcontractor in a foreign locale

Figure 1. Offshoring Versus Outsourcing

Second, psychological contracts are distinct from legal contracts. Specifically, they encompass people's beliefs about both written terms and unwritten implicit terms. Third, PCT promotes an individual level of analysis, focusing on the individual's beliefs and expectations in a social relationship. This is distinct from the more common interorganizational level of analysis.

These three aspects are also central to the OSS concept. First, mutuality and reciprocity are critical to the success of the OSS development model. These values are effectively enshrined in OSS development through the "copyleft" terms found in OSS licenses, which decree that software can be used, modified, and redistributed provided subsequent modifications are made freely available to others. Also, development is accomplished through the fulfillment of mutual obligations with respect to the activities of coding, debugging, testing, and documentation. One of the most significant threats for the OSS movement has been suggested to be the "free rider" phenomenon whereby someone profits from OSS without reciprocating, which thus contravenes these values of reciprocity and fulfillment of mutual obligations (Von Hippel and Von Krogh 2003).

Second, in relation to the PCT focus on psychological contracts (which differ from legal ones as they encompass both written and unwritten terms), much OSS development is done in the absence of any legal employment contract for developers. Also, the norms of how development is conducted are both written and unwritten. Developers are expected to be familiar with written rules and coding norms, for example, before they attempt to contribute (Feller and Fitzgerald 2002). However, the unwritten rules must also be learned by developers over time. New recruits to development ranks serve their apprenticeship in learning these unwritten rules and norms of expected behavior (Gorman 2003; Raymond 1999).

Finally, in relation to the PCT focus on the individual level of analysis, the signaling incentive identified by Lerner and Tirole (2002) as the basic motivation for developers to contribute to OSS projects (i.e., a combination of *career concerns* and *ego gratification*) apply primarily at the level of the individual. Furthermore, since many OSS developers are not affiliated with any formal organization, an individual level of analysis is to some extent required in any study of OSS developers.

Similar to this study, Koh et al. (2004) used an initial qualitative case study approach to derive a set of obligations that customers and suppliers need to fulfill in order to achieve a successful outsourcing relationship. They followed this with a quantitative survey analysis to validate and refine these factors. They recommend that their framework of obligations be applied in a context in which clear and traditional authority structures do not exist. In this study, we drew on their finalized set of validated obligations and sought to apply it in an open source context. In this case, we use the "customer" entity in the same sense as the Koh et al. study, that is, as the organization commissioning the opensourcing, seeking to grow a community around a product. However, the "supplier" entity becomes the open source community in our study. Below, we consider the final refined set of obligations identified by Koh et al., and discuss their possible relevance to an OSS context.

Customer Obligations

Koh et al. conclude that there are four specific obligations for which the customer must bear responsibility and which are associated with outsourcing success. These are

- Explicit and comprehensive requirements specifications for the services covered by the outsourcing project
- Prompt payment to suppliers and no unreasonable withholding of payments
- Close project monitoring with active overseeing of project progress, attending project meetings, and regular discussions
- Project ownership to ensure that senior management provides strong leadership, support, and commitment to the project

Explicit and Comprehensive Requirements Specifications

At first glance, explicit and comprehensive requirements specifications might seem to be at odds with the OSS development model which is predicated on the principle of a developer perceiving "an itch worth scratching," to use Raymond's (1999) memorable phrase, and thus not normally associated with comprehensive requirements specifications. Also, OSS developers have typically been users of the software being developed (Dinh-Trong and Bieman 2004; Gacek and Arief 2004; Mockus and Herbsleb 2002), and the software was often targeted to a horizontal domain (such as office desktop applications or software development tools). In such situations, clear requirements specifications are not necessary as these are widely understood and internalized by the individual developers. However, these aspects of the OSS development context are changing. Increasingly, OSS development is being purposively "steered" as customers seek to stimulate OSS development in specific vertical domains (such as automotive or telecoms) where a developer may not perceive an itch worth scratching. In these development situations, the specifications are not part of conventional software development knowledge and thus clear specifications are becoming more important. This increasingly explicit formalization of specifications is already evident in the commercially sponsored projects that are increasingly becoming a feature of the OSS landscape.

Prompt Payment to Suppliers and No Unreasonable Withholding of Payments

Again, prompt payment to suppliers and no unreasonable withholding of payments might seem at odds with the OSS model where actual monetary payment is often not a factor. However, recent studies show that a significant number of OSS developers are indeed employed within professional organizations (Lakhani and Wolf 2001). Also, the Lerner and Tirole (2002) study illustrates that "payment" can come in forms other than mere monetary compensation. Many OSS developers report the primary motivation as the rush they get from seeing their code in use and getting prompt feedback from peers they really respect (Feller and Fitzgerald 2002). This is in marked contrast to the proprietary software development model, where developers may wait months or even years to see their code in use. Thus, there is a sense in which prompt payment can arise, albeit in the form of surrogates such as peer feedback. When payments in the normal sense are not part of the equation, then the issue of unreasonable withholding of (monetary) payments is not likely to arise in the case of OSS.

Close Project Monitoring with Active Overseeing of Project Progress

Raymond's (1999) characterization of the cathedral versus the bazaar to differentiate OSS development from traditional development caused the perception that OSS development was merely about developers following their own agenda, developing in parallel in a spirit of optimistic concurrency. However, Raymond's characterization was based on a limited sample of OSS projects that didn't reflect the heterogeneity of the OSS development landscape. In recent times, OSS development has become more formalized. This is evident in the regular project meetings that are now a feature of a number of popular open source products, such as the Apache conferences in the United States and Europe, the Zope/Plone and PyPy development sprints, and the GNOME annual project conferences (German 2004) which bring together developers to coordinate and plan development.

Project Ownership and Senior Management Leadership and Support

The importance of strong management support has been verified in several studies of ICT adoption (e.g., Agarwal 2000; Chatterjee et al. 2002; Fichman 2004; Gallivan 2001). Project ownership and senior management championship is undoubtedly critical for radical, high-risk initiatives such as OSS deployment since it contravenes the traditional model where ongoing support is legally guaranteed by a vendor. Indeed, top management championship is likely to become even more important in the future as OSS adoption moves out of the domain of invisible infrastructure systems to more visible, high-profile desktop systems and IS applications.

Community Obligations

Koh et al. identify five specific obligations for which the supplier must bear responsibility and which are associated with outsourcing success. These are

- Clear authority structures which delineate the decision-making rights and reporting structures in the project, in terms of the roles and responsibilities of all parties involved
- Taking charge in terms of completing the job and solving problems independently, with minimal customer involvement
- Effective human capital management in assigning high-quality staff to work on the project, and seeking to minimize staff turnover during the project
- Building effective interorganizational teams—investing time and effort to foster a good working relationship among the team of customer and supplier staff working on the project
- Effective knowledge transfer in educating the customer in terms of the necessary skills, knowledge, and expertise associated with using the outsourced system or service

Clear Authority Structures

In the absence of traditional organizational sanctions, some form of structure is necessary to coordinate development. In many OSS projects, this takes the form of a benevolent dictatorship, as initially suggested by Raymond. Several studies of OSS development have detailed the complex authority structure that evolves over time to ensure that all code contributions are vetted and incorporated in a disciplined fashion (Mockus et al. 2002).

Taking Charge and Solving Problems Independently

OSS development has typically been characterized by the developers proactively taking charge, solving problems independently with minimal customer involvement, and without the need for traditional organizational sanctions to ensure development work is undertaken. Initially, OSS developers did not engage in formal requirements analysis with customers (Scacchi 2002), but took direct responsibility for development decisions. Even though the OSS development process is becoming more formalized (Fitzgerald 2006), OSS developers are still more likely to retain a strong sense of independence.

Effective Human Capital Management

Research has also focused on the human capital management aspects of OSS (e.g., Hann et al. 2002). This suggests that participation in OSS projects allows developers to gain highly marketable technical skills which in turn can lead to higher earnings in the future, which is also facilitated by the fact that the opportunity cost of participating in OSS development can be quite low as developers can choose the amount of work they do and organize it to fit their own personal timescale and agenda. Also, OSS developers have long been acknowledged as of high quality (Raymond 1999) and the loyalty of developers in the longevity of their commitment to their development projects has been remarkable (Feller and Fitzgerald 2002). Indeed, the cardinal sin of OSS, that of project forking (whereby a project is divided in two or more streams, each evolving the product in a different direction), is a strong community norm that acts against developer turnover on projects.

Building Effective Interorganizational Teams

The particular characteristics of OSS position it as a good exemplar of the "whole-product" concept of a market-driven business approach that seeks to deliver a complete solution to the customer in terms of products and services (Feller et al. 2008). In this scenario, developers do the coding while others complete the business model by adding sales and marketing services—necessary activities but ones in which developers may not be interested. The OSS whole-product approach is also larger than a single company or software product or service. Indeed, the network benefits of open source arise as a result of the size of the overall community and ecosystem. Thus, an interorganizational network of interested parties with complementary capabilities can form an ecosystem to offer a professional product and service in an agile, bazaar-friendly manner. Customer service requests can be routed to the most appropriate expert partner in the network, perhaps even to the developer who wrote the actual code.

Effective Knowledge Transfer in Educating the Customer

Again, this is a topic that resonates well in the case of OSS on a number of aspects. In the cases where the developer is also the user/customer, the issue of knowledge transfer does not

arise. However, the role of the user/customer is significantly elaborated in OSS as they can contribute to debugging, testing, documentation, etc (Feller and Fitzgerald 2002). Thus, a close working relationship can emerge between developer and user.

Qualitative Phase: Identifying Psychological Contract Obligations in Opensourcing

Research Method: Qualitative Phase

Given the relative newness of the opensourcing concept, it is unsurprising that there is not a solid research base to date on this phenomenon. Bearing this in mind, this study was concerned with initially achieving an increased understanding of this phenomenon. Thus, analysis of a small number of cases was deemed appropriate as such "revelatory cases" (Yin 1994) may provide the required rich insight. The primary case selected for the study was the Celtix project, an open source Java Enterprise Service Bus (ESB) sponsored by IONA Technologies. This was complemented with case study interviews of the DVTk (DICOM Validation Tool kit) project initiated by Philips Medical Systems and the Morfeo project initiated by Telefonica.

Background to Case Study Projects

IONA Technologies – The Celtix Project: IONA Technologies, a NASDAQ-quoted company, is headquartered in Dublin, Ireland, with U.S. headquarters in Waltham, Massachusetts, and offices worldwide. IONA was founded as a campus company at Trinity College Dublin in 1991, and provides products and services to help organizations build business-to-business enterprise portals. IONA is currently rated as the leading provider of standards-based platform middleware technology, with more than 4,500 blue-chip enterprise customers worldwide, who use IONA products to address large, complex application integration and achieve interoperability by means of a standards-based, service-oriented architecture. In June 2005, IONA extended its business model to incorporate open source by leading a community project to develop Celtix, an open source Java ESB that will coexist with Artix, the company's flagship integration product. The Celtix project is hosted by an established open source community, ObjectWeb, who specialize in developing open source middleware products. Most of ObjectWeb's members are small to medium-sized enterprises based in continental Europe. The Celtix project has achieved an impressive development productivity schedule, proceeding through four significant development milestones, a beta release to a fully stable 1.0 release in just over 10 months.

Philips Medical Systems – The DVTk Project: Philips Medical Systems specializes in medical devices and is part of the global technology organization Philips, headquartered in Eindhoven, The Netherlands. DICOM (Digital Imaging and Communication in Medicine), initiated in 1993, is a global standard for storing and processing medical images and is used in virtually all hospitals worldwide. The DVTk product provides extra functionality for problem solving and improving the quality of the DICOM interface. It began as a collaboration between Philips and AGFA, and was released as open source under the GNU LGPL license in June 2005. Since then a community has grown up around the product with contributions from independent developers worldwide as well as developers in AGFA and Philips.

Telefonica – The Morfeo Project: Telefonica I+D, the research and development division of Spanish telecom operator Telefonica, initiated the Morfeo project, which operates in the area of service oriented architectures, and aims to speed up the development of software standards in this area. Telefonica I+D is the customer "engine," releasing proprietary software components and injecting resources into the community. The project has set up its own Morfeo implementation of a SourceForge-like portal, where the development base of the subprojects integrated in Morfeo are hosted, including source repositories, binaries, mailing lists, bug and change trackers, and documentation.

Data Collection and Analysis

Data was gathered over a 12-month period from July 2005 to June 2006, and drew on a number of sources (see Table 1). These ranged from workshops to a series of interviews, both formal face-to-face and informal telephone interviews. In keeping with Patton (1990), an interview protocol guide was developed (see Appendix A) based on the customer and community obligations identified earlier. Marshall and Rossman (1989) identify the importance of being able to gain entry to a company and maintain continuity of presence for as long as necessary. We sought to achieve this by conducting initial interviews with the chief scientist at IONA (the customer in our study) and the chairman of ObjectWeb (the supplier community). Subsequently, in the Philips/DVTk and Telefonica/Morfeo projects, we conducted initial interviews with the lead customer and community figures. These interviews served to give a good strategic overview of the project and the high-

Table 1. Data Sources for Qualitative Phase

Workshops	Interviews	Supplementary Sources
September 2005: Presentation and discussion of Celtex business model and strategy. April 2006: Workshop presentation on opensourcing strategy and discussion of DVTk and Morfeo projects. July 2006: Debriefing presentation of findings.	July 2005 – June 2006: Interviews with • Chief Scientist, IONA • Chairman, ObjectWeb • Administrator, IONA • Open Source Program Director, IONA • Two project managers, IONA • Two developers, ObjectWeb • Two managers, Philips Medical Systems • Developer, DVTk • Manager, Telefonica • Developer, Morfeo	IONA AND ObjectWeb maintain detailed and comprehensive web portals for the Celtex project. We also had access to mailing lists and project development wiki pages. Also, detailed web portals and mailing lists for DVTk and Morfeo projects were available.

level obligations that were in place. These lead individuals identified key figures in the projects and facilitated access to these interviewees. Following this, as other key informants emerged during the interview process, support from leadership in both the customer and community entities greatly facilitated achieving access. Most comprehensive studies of open source developers up to now have relied on anonymous surveys (e.g., Ghosh et al. 2002; Lakhani and Wolf 2001), the studies by Hann et al. (2002) and Mockus et al. (2002) being notable exceptions. This is caused in part by the difficulty in getting personal access to key developers, but this study was notable in achieving such access.

The interview protocol guide approach was chosen as it is more comprehensive and systematic for data collection than the purely conversational interview, and more flexible than the standardized open-ended interview or the closed, fixed-response interview. The duration of interviews ranged between 30 minutes and 90 minutes. Interviews were recorded so as to minimize data loss due to note-taking, and these recordings were subsequently coded. The interview guide was e-mailed to interviewees in advance to allow them an insight into the overall issues on which we wished to focus. Informal follow-up interviews took place to clarify and refine issues that emerged following the interview transcription process. The interview transcripts comprised almost 60,000 words. These interviews were complemented by comprehensive reviews of documents and communications on the mailing lists, project wiki, and web sites. Also, the project findings were presented to the participants and other researchers over a series of workshops with in-depth discussions feeding back into the analysis process (see Table 1).

Qualitative analysis was undertaken using coding techniques proposed by Strauss and Corbin (1998), and exemplified by the research of Orlikowski (1993) and Baskerville and Pries-Heje (1999). Interviews were transcribed and then coded based on the seed categories (Calloway and Ariav 1991; Miles and Huberman 1994) represented by the customer and community obligations derived earlier, and analytical memos were written as patterns and themes emerged from these field notes. This was combined with an open-ended analysis in which we sought to break free from the initial obligations to allow for new categories to emerge. As a result, our revised set of obligations, presented below, deviates but yet incorporate constructs from the initially proposed set of obligations (cf. Klein and Myers 1999). The analytical memos provide a comprehensive grounding of these revised obligations in both existing theory (OSS and outsourcing) and in our gathered empirical data. The overall structure of this research process along with coding examples can be found in Appendix A.

The initial round of coding was done by the two authors—independently at first, followed by a joint effort to compare and contrast findings and interpretations. We did not attempt to capture inter-rater reliability. Instead, the results of this initial coding were then presented to two other members of the research team, and issues that were unclear or on which there was no unanimity between the initial two coders were discussed. Outstanding issues were clarified in follow-up telephone interviews. Also, the emerging results of the qualitative research were presented and discussed at three workshops attended by the authors and many of the interviewees.

A problem that has been identified in relation to qualitative research is what is termed multiple realities. This refers to the unavoidable fact that the understanding of reality is based on an individual interpretation of the data, and that different individuals may interpret the same data in different ways (Kaplan and Duchon 1988). This problem was addressed in

a number of ways. First, our approach to data analysis explicitly recognizes this problem of subjective data interpretation and, to address it, uses rigorous coding and memoing processes which provide a traceable, documented justification of the process by which research conclusions were reached, thereby providing an audit trail of the process (Guba 1981). Second, the method of venting was used. This is a process whereby results and interpretations are discussed with professional colleagues (Goetz and LeCompte 1984). The findings were formally presented and discussed with colleagues in detail on several occasions at practitioner/researcher workshops and conferences. Also, in this study, IONA, Object Web, Philips, Telefonica and Morfeo representatives were active participants in a EU-funded research project led by the authors. Thus, as findings were presented and discussed at the project workshops, quite detailed member-checking of our interpretation of the findings was possible.

Findings and Discussion: Qualitative Phase

Here we discuss the obligations that emerged from the interviews. In our approach, we asked both customer and community interviewees to discuss their perceptions of their own obligations, and also the obligations they would expect from each other. Although the interviews were based on the distinct obligations identified earlier, it quickly emerged that many of these obligations were symmetrical.

Here we present the refined set of obligations based on the interview evidence.

Refined Customer Obligations

Achieving Consensus on Development Roadmap: Initially we expected that the customer would provide an explicit requirements specification of required functionality, which would be in keeping with the evolution of open source toward more formalized commercial phenomenon (Fitzgerald 2006). However, this was not in fact the case. Interviewees stressed the manner in which requirements specification here differed from traditional outsourcing. Companies may have a clear idea of what functionality they would like to see the community adding to the product. However, there has to be consensus as to what functionality will be added. If a company pushes its own agenda too much in seeking to control development, there can be problems. The Celtix project manager within IONA expressed it well:

> *A company cannot just go onto the mailing list or the community, and say, "Can you guys build this?" When kicking off the project in the open source community, it's about stating the overall goal and the top-level requirements you are trying to achieve. Then it's driven by consensus. If people perceive you as driving your own agenda, then you will get pushback on having things accepted.*

This again emphasizes the delicate equilibrium that must be maintained between acceptable OSS community values and customer desire for value creation (Fitzgerald 2006). Interestingly, it was stressed that within the ecosystem formed around this mode of development, it was quite permissible for customers to engage in more traditional outsourcing relationship directly with some developers in the community, outside the strict remit of the opensourcing project.

Also, it was often the case that the initial development community contributions were to provide something that existed within their repertoire. Thus, their contributions were not based on an assessment of the most pressing commercial priority for the company; rather, they considered what they could offer based on their existing repertoire and expertise.

Project Ownership: The customer interviewees identified with the project ownership obligation. One customer interviewee stressed the radical change in mindset represented by opensourcing, suggesting that it represented a strategic initiative which differed from the normal business model where developers could see that their salary was pretty much directly derived from the sales of the commercial product which they developed. In the opensourcing model, their work could appear to be benefiting the open source community, and not leading to an obvious direct remuneration. Thus, top management championship was necessary to convince developers that the strategy made business sense. In fact, the Celtix project could also grow the market for other proprietary IONA products, enabling additional support contract revenue.

In keeping with demonstrating strong ownership and commitment to the project, IONA established a full time position: Open Source Program Director. This person is responsible for engaging with the community, and helps ensure that issues relevant to the open source community receive prompt attention, which helps ensure the quality of the software. Also, senior management commitment can help ensure that R&D resources are provided. Interestingly, however, there is a delicate equilibrium to be maintained here also. The Celtix project manager at IONA suggested that if the project is seen as too much an IONA project, the developer community may have less interest in getting involved. A manager at Philips Medical Systems summarized this dilemma:

> *This application is deeply used within Philips organization. In several processes we depend on it.*

And because it's part of our product release, we want to stay in control of it. So on the one hand you want to control the project, and then on the other hand you want to be an open project and provide freedom for independent developers to join. And there's a little bit of conflict between these positions.

Marketing Project to Increase Visibility: Several interviewees emphasized the need for the company to market the attractiveness of the project and improve its visibility. This has a two-fold purpose. First, the development community will perceive their reputation has been improved through involvement in a high-profile project with a high profile company. As a DVTk community interviewee expressed it, *"Working with them [the company] is almost a sort of professional honor, as it were."*

This helps ensure the vibrancy of the project, and can attract further developers to the community, which cannot be taken for granted in an open source project. OSS is an emergent phenomenon and very few projects to date have been deliberately started and nurtured to be successful; rather some extremely successful ones have emerged over time, whereas others have died off. Again, the Celtix project manager offered an interesting insight:

There are a lot of open source projects which don't go anywhere, even though they have built good code. It also needs to be pushed so that it gets noticed and used by other projects, documented, and marketed. This is a big overhead, and commercial companies have structures in place to help achieve that.

Interviewees also suggested that the customer company could provide their expertise in relation to software commercialization and productization in creating a professional OSS product and subsequently marketing that product. This would involve a holistic approach and proactive marketing to ensure that all who could usefully consume the product were made aware of it, and could contribute. It was felt that a commercial vendor could usefully complement the OSS community by providing this expertise. These efforts grow public awareness of the product, which can then create business-consulting opportunities for members of the development community. A DVTk community interviewee captured the essence of this well:

What I am looking for is a multiplier. I am providing a piece of my work. I know they [the company] already have a useful chunk of work. But adding my bit to their larger existing work in a cooperative way creates something of greater multiplied use to everybody, including myself.

Transparency and Close Project Monitoring: Given that there is a strong external element in opensourcing, customer interviewees stressed the necessity for clear project milestones and more visibility about product releases. This was contrasted with traditional proprietary development where internal milestones and actual release times are perhaps deliberately kept vague, whereas the need for the customer to be clear about future project plans was important to the community. More frequent product releases was identified as a by-product of the open source approach.

Also, it was suggested that the customer could not insist on a particular project-monitoring regime. Rather, different open source communities may have different norms and approaches in this area, and the customer has to be flexible and prepared to adapt to the particular regime in vogue in the particular open source community.

Related to transparency there was also a community expectation that the company have an open process for accepting community contributions. Interestingly, the company can have unrealistic expectations as to the level and significance of contributions. Indeed, one company manager suggested his experience was that of *"one significant contribution per month from the external community."*

Related to this is the issue of licensing and intellectual property (IP). A senior community interviewee with a background in commercial development suggested that it was sensible and pragmatic to have a clear IP policy. They had requested that IONA release the Celtix project under the Lesser General Public License (LGPL) and IONA agreed. The Celtix project manager suggested that IONA were keen to embrace open source and build trust within the community, and the choice of license is a key determinant for developers in deciding whether to participate in a particular OSS project, and also for companies to adopt. Thus, the development community tends to have a preference for a restrictive (so-called viral or reciprocal) GPL-style license that safeguards their contributions due to its reciprocal nature. However, companies may tend toward more permissive licenses that afford them greater control over the future of the software. IONA had perceived the need to be even more open to other companies and communities, and hence dual licensed Celtix under the less restrictive Eclipse Public License as well.

Related to this issue was the idea raised within Philips Medical Systems of creating a single legal entity or foundation to represent the project. This trend toward legal incor-

poration has been established in several OSS projects (O'Mahony 2005).

Creating a Sustainable Ecosystem – Customer Responsibilities: It was seen as important that both company and the community members strive toward the creation of a sustainable ecosystem around the product. A vital factor here is to create an atmosphere of trust. Creating an ecosystem and engendering trust is also facilitated by the fact that the project interaction tended to be *"very much techie to techie,"* as one interviewee put it. The Celtix project management committee is chaired by an IONA Distinguished Engineer who would garner respect from the technical OSS development community.

There was broad agreement from the community interviewees on this issue also. However, one community interviewee stressed the importance of meaningful content in the feedback. While promptness was appreciated, this evaporated if the content of the feedback was "empty." However, the techie-to-techie nature of the relationship helped ensure that feedback was meaningful.

Another issue with significant implications for project management practices within the customer company emerged in relation to development staff rotation. Normally, within proprietary software development companies, development staff are rotated after a few months onto other projects. However, because of the strong techie-to-techie interaction between developers in the company and community, there was a strong pressure from the community that developers would remain on the project for a lengthy period. This is in turn matched by the company expectation that community developers show strong loyalty and commitment to the project.

Refined Community Obligations

We now present the refined open source community obligations that emerged from the qualitative phase.

Clear and Democratic Authority Structure and Process Transparency: Community representatives identified clear and transparent authority structures as important—indeed, one stressed that these are *"not only important, but mandatory."* It was argued that since professional involvement in OSS is increasing, the OSS development community is expected to show the same level of quality and transparency as could be expected from any professional organization: people need to understand how decisions are made. Indeed, professionalism is permeating opensourcing projects since *"everybody knows there are business reasons why people are there,"* as pointed out by an IONA project manager, and also emphasized by a DVTk community member confessing that, *"Getting good karma is great, but it's not my primary objective."* When the opensourcing community involves a large number of traditional professionals, this is particularly emphasized.

The customer interviewees also agreed that clear authority structures are important. However, in opensourcing, authority structures are framed by a strong belief in democratic principles:

> *It would be good to ensure that the [democratic] process is working, but I'm not sure that it is possible to see any authority structures other than that. It will always be shared responsibility.* [Distinguished Engineer, IONA]

It was furthermore pointed out that such structures are important in two different respects. First, they provide consistency between projects, which means that developers can easily contribute to more than one project. Contributing to several projects is not uncommon in OSS, and with the increasing interest in the so-called whole product approach, which was discussed earlier, this is expected to be increasingly important, a point stressed by the Open Source Program Director at IONA. Second, they provide for consistent terminology within and across projects, which ensures people are *"on the same page, and really focus on innovation."*

As a complement to clear authority structures, the lack of a traditional, written outsourcing contract means that an open source community must be clear also about their actual work processes. Interestingly, this mirrors the "transparency and close project monitoring" obligation that is expected on the part of the customer.

Responsible and Innovative Attitude: Although community and customer interviewees alike believed it to be essential that the community takes responsibility and delivers on what has been committed to, this outsourcing obligation becomes somewhat blurred in the opensourcing context. In the Celtix project, for example, paid IONA employees do a large part of the overall development. Hence, the customer has the power to manage part of the development effort more directly than would be possible in a traditional offshore outsourcing context. Although this may change as the Celtix project matures and the number of external contributions increases, an IONA project manager explained that there currently seems to be a feeling that *"there will always be customer involvement."* However, he also suggested that a lot more community inde-

pendence was expected in the future and that responsibility follows naturally with independence. A manager at Philips Medical Systems captured this crisply, referring to a potential problem with lack of written contracts:

> *I think one of the biggest risks you have in open source development is when someone says, "Yes I'm going to do that," and he doesn't put any effort into it, nothing is done.*

This obligation also supports the customer obligation of "achieving consensus on the development roadmap," which assumes that the OSS community is innovative and contributes constructively to this roadmap. As this will have a positive impact on the project, the community is expected to help achieve a positive impact among (the customer's) customers.

Creating a Sustainable Ecosystem – Community Responsibilities: As mentioned earlier, both company and community members are expected to aim for a sustainable ecosystem.

Consequently, community members are expected to be loyal and committed to continued involvement in the project, which, as mentioned earlier, is a marked feature of OSS projects. This was confirmed by a Telefonica manager who explained that *"a benefit that open source brings is that people are more prepared to commit,"* which also transfers across to customer developers who *"even though they're working on the same technology now commit more personally."*

From a community perspective, it was suggested that the high-quality software associated with the OSS model is an indication that the human capital management is working. It was also perceived that the quality of the code is a way to attract more business, which is essential for the ecosystem to develop. As OSS is moving away from networks of individuals to networks of companies, if a contributor earns a reputation for producing high-quality code, customers will keep coming back for more. It is also the case that customers sometimes use the OSS model to identify the best suppliers, who are then approached directly and contracted in a traditional outsourcing model.

Opensourcing customers expect the OSS model to attract high calibre people who understand the project domain very well without requiring additional training. The Open Source Program Director of IONA even argued that it attracts a certain personality, with traits not necessarily those traditionally associated with a top-notch programmer. In her view, people are attracted by the OSS model because they want to build something better, they want to get involved, and they want to be part of a community—in summary, *"These are the kind of people that I would want on my team, whether I was doing open source or not."*

Interestingly, OSS community members do not necessarily see themselves as suppliers commissioned by a customer in the traditional sense. In fact, the open source community as represented by ObjectWeb sees its members not as OSS developers but as *ecosystem developers*. There seems to be a definite trend toward more organized open source communities, such as that of ObjectWeb. Hence, facilitating effective interorganizational teams is to a large extent an intrinsic property of the opensourcing model. As indicated earlier, this is due to the merging of customer and community into one ecosystem: *"I don't consider IONA as a customer. IONA is a member"* was how the ObjectWeb Chairman described the situation. Opensourcing is thus not just about building good working relationships between customer and supplier. It is about ecosystem development. Hence, although there are business reasons for people to participate, there is a lot more collaboration than in traditional outsourcing:

> *In a traditional market you don't call up your competitor and be like, oh, well tell me what your stuff does. But in open source you do.* [Open Source Program Director, IONA]

In opensourcing, the software developed is typically not aimed for end-users but is more likely to be tools and infrastructure components. Consequently, the customer and community participants typically share the same level of technical expertise: it is mostly developer-to-developer communication. Therefore, there is no need for formal training. Instead, knowledge transfer is happening continuously from one research lab to another within the ecosystem. This was emphasized by the ObjectWeb Chairman, who asserted that, *"I don't speak about education or anything like that, I speak about exchange between researchers."* The Open Source Program Director at IONA acknowledged this view, referring to it as "cross pollination." According to a project manager at IONA, knowledge transfer was also facilitated by an early and proactive focus on documentation as part of the work toward achieving consensus on a development roadmap discussed earlier (a customer obligation).

Summary of Identified Obligations

Table 2 summarizes the refined list of obligations in the context of opensourcing.

Table 2. Summary of Refined Customer and OSS Community Obligations in Opensourcing

Customer obligations (i.e., obligations for which the customer must bear responsibility):

(1) Achieving consensus on development roadmap:
- Not too forceful and dominant in pushing own agenda
- Accept a general roadmap (vision) of future functionality rather than seeking a precise requirements specification

(2 Project ownership:
- Provide senior management commitment to the project
- Provide R&D resources to further develop the project
- Help improve the quality of the software

(3) Marketing project to increase visibility
- Provide professional expertise in relation to marketing and productizing the software
- Seek to improve the reputation of the community of contributors
- Provide a business opportunity for the community to use the product

(4) Transparency and close project monitoring:
- Transparent in plans for the future of the project
- Open to outside contributions
- Use an appropriate license to safeguard community contributions

(5) Creating a sustainable ecosystem:
- Seek to create trust in the relationship with the community
- Engage in community-sustaining activities
- Behave as a responsible member of the opensourcing ecosystem
- Preserve continuity by keeping developers on projects for a longer period than the norm in proprietary software development

Community obligations (i.e., obligations for which the community must bear responsibility):

(1) Clear and democratic authority structure and process transparency:
- Provide a transparent authority structure to allow customer see the decision making process within the community
- Behave as a professional team

(2) Responsible and innovative attitude:
- Take responsibility and deliver on what is committed to
- Be creative and innovative in suggesting new functionality and directions for the project
- Help achieve a positive impact among customers

(3) Creating a sustainable ecosystem:
- Offer high quality people who understand the project domain very well without requiring additional training
- Exhibit loyalty and continued involvement in the project

Quantitative Phase: Exploring Effects of Fulfilled Obligations on Opensourcing Success

In the second phase of the study, we drew on the refined obligations from the qualitative phase to explore the opensourcing phenomenon further by means of a quantitative survey.

Opensourcing Survey

A survey questionnaire was constructed to operationalize the obligations summarized in Table 2, and also comprising factors related to open outsourcing success. Respondents were asked to consider a particular opensourcing project of which they had experience and, on a five-point Likert scale, assess to what extent these obligation factors were met and to what extent the project was perceived as successful. In order to improve instrument validity, we drew on the advice of Straub (1989) and incorporated relevant items that had previously been validated by Koh et al. (2004) in a similar survey.[2] This questionnaire was then pretested with four practitioners with experience of opensourcing—two from the customer side and two from the community side. Some minor modifications were made based on this feedback, and then we considered the issue of constructing a survey sample.

Given the newness of this area of research—to our knowledge this is the first comprehensive study of opensourcing—it was not possible to identify an overall population from which a random sample could be drawn, an issue which has been discussed in relation to open source survey research in the past (Ghosh et al. 2002). Hence, our study is by necessity quite exploratory, and a purposeful sampling strategy was employed which sought to identify information-rich cases using a form of operational construct sampling (Webb et al. 1966); that is, we sought to identify real world operational examples of the phenomenon of interest. Prior to the study we were aware of a number of possible exemplars of opensourcing in addition to the IONA/Celtix, Philips/DVTk and Telefonica/Morfeo projects. Two other exemplars of the liberation approach to opensourcing were Eiffel Studio and OpenAdaptor. In addition, to also capture the commercialization approach we included two well-known examples, MySQL and Canonical/Ubuntu. Overall, these projects represented good operational candidates of the opensourcing phenomenon.

[2]We would like to acknowledge gratefully the assistance of Christine Koh, Soon Ang, and Detmar Straub in providing a copy of their survey instrument.

Again, we chose to focus primarily on key figures in these companies and communities and sent them a web-link to the online questionnaire, requesting that they complete and forward to persons they considered relevant. Such sampling is known as *snowball* or *chain sampling* (Patton 1990), and is used as a tactic to try to ensure information-rich cases. This approach has also been used successfully in a previous survey of open source (Ghosh et al. 2002). Reminders were sent weekly until the cut-off date, resulting in a total of 218 responses. A small number of responses were incomplete, and these were eliminated. Also, a number of individuals who had participated in the interview phase responded to the survey. These too were eliminated so as to avoid any primacy bias (Koh et al. 2004). This left a total of 207 usable responses.

Given that the survey was online and that we asked the principal contact points to further distribute it to relevant partners in their networks, we cannot indicate the total number of survey questionnaires sent to the population, as usually can be done with a postal survey. Therefore, the issue of nonresponse bias was quite important. We investigated this through the use of late respondents as surrogates for nonrespondents (Wallace and Mellor 1988). We compared the responses of a random sample of 20 early respondents with a random sample of 20 late respondents. There were no statistically significant differences. On visual inspection, the most notable difference was that early respondents tended slightly more toward the extremes in reporting their levels of satisfaction and dissatisfaction with the opensourcing experience than late respondents. This is somewhat intuitive, perhaps suggesting that early respondents tended more toward the extremes of satisfaction and dissatisfaction with the opensourcing phenomenon, and may have felt more compelled to respond quickly, but again it must be stressed that this was not a statistically significant result.

Analysis of Survey Responses

Demographics

Respondents were quite evenly distributed with 56 percemt representing a company and 44 percent representing an open source community. Also, 53 percent of the respondents reported more than 5 years of experience with open source, with fewer than 10 percent reporting less than 1 year of experience with open source. An interesting demographic also arises in relation to gender. Previous studies (Ghosh et al. 2002; Lakhani and Wolf 2001) indicate that over 98 percent of OSS developers are male. In this study, just over 4 percent of the respondents were female, breaking down into

Table 3. Respondents by Country	
Percentage of Respondents	**Countries**
less than 1%	Argentina, Australia, Austria, Bangladesh, Chile, Costa Rica, Denmark, Hong Kong, Hungary, India, Iran, Lithuania, Mexico, New Zealand, Norway, Peru, Philippines, Poland, Portugal, South Africa, Venezuela
1 – 5%	Belgium, Brazil, Canada, Finland, Germany, Sweden, Switzerland
5 – 10%	France, Italy, Netherlands, Pakistan, United Kingdom, United States
more than 10%	Ireland, Spain

3 percent of community respondents and 5 percent of company respondents. While this is not a major breakthrough in redressing gender imbalance, it does suggest that the relative success of women in the broader IT industry in general may at last percolate into the open source arena.

Table 3 summarizes the primary work location of respondents. The fact that 37 countries across all continents are represented confirms the global offshoring nature of the opensourcing phenomenon. The high number of respondents from Ireland and Spain, and to a lesser extent France and the Netherlands, is indicative of our continued focus on the IONA/Celtix, Telefonica/Morfeo, and Philips/DVTk projects, which were the basis of the qualitative research phase, as the companies driving these projects are based in these countries.

Construct Validity and Reliability

To explore the validity of our proposed company and company obligations, we conducted principal component factor analysis (PCA). One of the assumptions for factor analysis is that data be measured on a continuous scale, but for exploratory factor analysis, ordered categorical data based on Likert scales, as in this study, is justifiable (Hutcheson and Sofroniou 1999). Also, factor analysis is based on correlation coefficients and thus is more reliable with larger sample sizes, estimated at 150 cases and above (Tabachnik and Fidell 1996). Our sample size of 207 cases exceeds this threshold considerably. It is also recommended to have a ratio of between 5 and 10 participants per variable (Hair et al. 1984; Kass and Tinsley 1979). Given that we had 22 variables in our initial analysis, we are at the safer upper-end of this estimate. However, these researchers suggest that these heuristic guidelines be complemented by other tests in the particular research context. One such alternative for investigating sampling adequacy for factor analysis is the KMO statistic (Kaiser 1974). The KMO statistic is based on the pattern of correlations among the variables and indicates whether the analysis can lead to distinct and reliable factors. Kaiser suggests that KMO statistic values in excess of 0.9 are "marvellous" while in excess of 0.8 is "meritorious." The KMO statistic value for our dataset was 0.86, which is clearly at the upper end of the sampling adequacy recommended value.

For the company obligations, PCA (with Varimax rotation, eigenvalues greater than 1.0, and 25 iterations) of the company obligations produced a set of three higher-level component factors for the customer obligations, which explained 49 percent of the variance in the company sample. Three of the hypothesized customer obligation items with inadequate loadings (below 0.5) were dropped and excluded from further analysis. Factor loadings for the remaining items were all quite high (0.57 to 0.75). PCA (again with Varimax rotation, eigenvalues greater than 1.0, and 25 iterations) of the community obligations confirmed our original hypothesized set of seven factors for the community obligations, which explained 46 percent of the variance in the community sample. Again, the factor loadings were quite high (0.58 to 0.76).

We also investigated the reliability of factors as outlined in Table 4 which shows descriptive statistics, correlations, and Cronbach's α. The Cronbach α values (in parentheses on the diagonal) were all above the threshold level of .6 for construct reliability recommended for an exploratory study (we do not report the Cronbach for one of the customer obligations—customer did not seek to dominate and control the process—as this factor was measured by just two items). In terms of discriminant validity, none of the off-diagonal correlations were above .80, which can indicate extreme cases of multicollinearity (Ghiselli et al. 1981).

Table 4. Means, Standard Deviations, Scale Reliabilities and Intercorrelations

	Variable†	Mean	SD	1	2	3	4	5
1	Create open company-community ecosystem	3.67	0.76	(.79)				
2	Provide professional business expertise	3.61	0.61	.42**	(.63)			
3	Did not seek to dominate and control process	3.37	0.80	.33**	.12	(–)		
4	Community professional obligations	3.74	0.61	.42**	.27**	.28**	(.79)	
5	Opensourcing success	3.95	0.81	.54**	.37**	.28**	.57**	(.84)

†Items measured on a five-point scale (1 = strongly disagree, 5 = strongly agree)

Reliability coefficients in parentheses on diagonal.

**Correlation is significant at the 0.01 level two-tailed.

Company Versus Community Differences

In our questionnaire, we asked both the company and community respondents to rate the importance of fulfillment of each other's obligations. We conducted Mann-Whitney tests to investigate this. The analysis reveals a number of obligations on which company and community differed to a statistically significant extent ($p < .05$). Community respondents were less positive than company respondents in relation to the customer obligation factor "Was open to outside contributions." This suggests that the community were not fully persuaded that companies were fulfilling their obligations in relation to accepting contributions. Typically, in open source projects, a meritocracy emerges over time as developers "prove" themselves and are allowed to commit contributions. The discrepancy between the company and community here suggests that companies are cautious about accepting outside contributions into the code base, but presumably a meritocratic structure could emerge in time, which would see companies overcoming this reluctance.

However, the company respondents also differed from community respondents ($p < 0.05$) in that they were less positive about two community obligation factors: "Provided a transparent authority structure to allow company see the decision making process within the community" and "Helped improve public perception of the project." This suggests that companies did not always perceive communities as being open in relation to development decisions, and also that the community might be remiss in relation to helping grow the public profile of the project. Given that companies see open outsourcing as a means of growing product awareness, this is an obvious point of potential conflict, as companies would expect the community to help in this regard.

We also measured respondent opinion as to the success of opensourcing. However, there were no statistically significant differences between the company and community respondents.

Regression Analysis of Obligation Fulfillment and Opensourcing Success

Finally, we conducted stepwise regression analysis to investigate the relationship between fulfillment of company and community obligations and the perceived success of opensourcing. One assumption behind the use of regression analysis is that the variables are measured on a continuous scale and are normally distributed. The PCA factor analysis allowed us to generate four composite factors for the company and community obligations which satisfied these assumptions. These were used in stepwise regression. Our model suggested that all four components were significantly associated with success, explaining over 47 percent of the variance ($F = 38.56$, $p < 0.001$). Table 5 provides the details of the regression analysis showing the standardized beta coefficients and significance levels.

Overall Conclusions and Implications

The aim of this study was to explore critical customer and community obligations that contribute to success in an opensourcing relationship of offshore sourcing. To achieve this, we adopted a two-phased research approach consisting of an initial qualitative multiple case study followed by a more quantitatively oriented large-scale survey study. Adopting a psychological contract perspective, the first phase used the outsourcing obligations identified by Koh et al. (2004), adapted to the opensourcing context, as a basis for identifying customer and community obligations that were associated with opensourcing success. These were then further explored by means of a survey among customer and community representatives with experience of opensourcing. Our results show that the fulfillment of certain customer and community obligations is significantly associated with opensourcing success. Interestingly, these customer and community obligations are

Table 5. Regression Analysis

Dependent Variable	Independent Variables	Model†
Opensourcing success	Company Obligations	
	Create open company–community ecosystem	.35***
	Provide professional business expertise	.21***
	Did not seek to dominate and control process	.12*
	Community professional obligations	.38***
	F	38.56***
	R^2	.47
	Adjusted R^2	.46

†Model statistics are standardized betas

*p < 0.05, **p < 0.01, ***p < 0.001

partly symmetrical and complementary and thus represent a tension between customer and community that needs to be managed for the opensourcing arrangement to be successful. Our final set of obligations is shown in Table 6 and discussed below where we consider the practical contributions of the study in terms of implications for practice and also theoretical contributions with implications for research.

Implications for Practice

Given that the opensourcing phenomenon has only recently emerged, success is not guaranteed. Our study, which sought to investigate the extent to which the fulfillment of customer and community obligations are linked to success, is thus useful in that it identifies those obligations which appear most critical, and which can act as a checklist of salient issues for customers and communities engaging in opensourcing. Also, the symmetrical and complementary nature of the obligations illustrates how these obligations need to be operationalized differently by the customer and community, but managed jointly, to ensure achievement of the mutual goal. Here we identify the potentially problematic issues that companies need to be aware of, and we also identify some new practices, or significant changes to traditional practices that are required.

Complementariness of Obligations: Product Lifecycle

While, the customer must be prepared to compromise at all stages and not seek to dominate and control the opensourcing process, the community must provide a transparent and democratic authority structure with shared responsibility. Hence, while the customer must be tactful and seek to embrace the OSS values of openness and democracy, the community must be able to show the same level of management capability as would be expected from a traditional outsourcing partner. It is important that the customer avoids pushing too hard its own agenda and accepts that requirements evolve throughout the project with active input from the community. To reciprocate, the community must make sure that its development process is communicated and accepted by the customer. Furthermore, the opensourcing customer has to provide complementary expertise in relation to product commercialization and marketing. In turn, the community is expected to help improve public perception and awareness of the product.

When viewed holistically, a product lifecycle is evident. First, the community is expected to take responsibility and deliver on what is committed to. In turn, the customer can provide R&D resources to complement and further develop innovative ideas, and also provide its professional expertise in relation to marketing and productization of software, and in improving the visibility of the product. Furthermore, the credibility provided by the company in marketing a product can create a business opportunity for the product which community developers can then leverage.

Differing Perceptions of Obligation Fulfillment

Our study suggests that customer and community can have different perceptions of the extent to which each has fulfilled its obligations. In particular, the community was significantly less likely to concur with the customer on the extent to which the customer is open to outside contributions. Further evidence of tension in this regard arose in the customer opinion

Table 6. Finalized Set of Customer and Community High-Level Obligations with Detailed Responsibilities

Customer Obligations	*Community Obligations*
Do not seek to dominate and control process • Not too forceful and dominant in pushing own agenda • Accept a general roadmap (vision) of future functionality rather than seeking a precise requirements specification	Clear and democratic authority structure and process transparency • Provide a transparent authority structure to allow customer see the decision making process within the community • Behave as a professional team
Provide professional management and business expertise • Preserve continuity by keeping developers on projects for a longer period than the norm in proprietary software development • Provide a business opportunity for the community to use the product • Provide professional expertise in relation to marketing and productizing the software • Provide R&D resources to further develop the project • Provide senior management commitment to the project	Responsible and innovative attitude • Take responsibility and deliver on what is committed to • Be creative and innovative in suggesting new functionality and directions for the project • Help achieve a positive impact among customers
Help establish an open and trusted ecosystem • Behave as a responsible member of the opensourcing ecosystem • Open to outside contributions • Transparent in plans for the future of the project • Seek to create trust in the relationship with the community • Engage in community-sustaining activities	Help establish a professional and sustainable ecosystem • Offer high quality people who understand the project domain very well without requiring additional training • Exhibit loyalty and continued involvement in the project

expressed above that there may be only one significant community contribution per month.

On the customer side, there was also significantly less satisfaction that the community were providing a transparent authority structure in relation to the decision making process, and also there was less satisfaction with the community efforts to help improve the public perception and awareness of the product. Given that for many companies one of the goals behind opensourcing is to grow the potential market for the product, this is an important issue. Both customer and community participants need to be aware of these sources of tension so they can be avoided if possible.

Achieving Balance between Value Creation and Community Values

Furthermore, the customer must seek to reach consensus on the project monitoring system that will be instituted, on trying to show leadership and ownership of the project but not so strongly as to deter the development community. Overall, the customer must achieve that delicate equilibrium between value-creation in terms of a successful business model for itself while not transgressing the OSS community values. This includes embracing OSS values of openness to be trustworthy in the eyes of the community. On the other hand, the community must adopt a professional attitude and approach in order to be taken seriously in the corporate world. Thus, the ethics of crowdsourcing has been questioned due to its taking advantage of the creativity of the user community for commercial gains (Bruns 2007). Successful opensourcing is rather characterized by reciprocity and symbiosis (see Dahlander and Magnusson 2005).

Changes to Standard Development Management Practices

We already referred to the need for management support on the company side to ensure buy-in to a risky initiative where it appears as if the "crown jewels," the company's software,

is being given away for free. Standard operating practices that tend to apply in the customer company may need to change. For example, more clarity is required in relation to software release milestones so the community can plan their own business opportunities. Also, more frequent product releases are likely rather than artificially separating product release functionality on the basis of the revenue that can be generated, which is often the case with proprietary software product releases. Furthermore, the policy in companies of rotating developers onto different projects after a period of several months may not be sustainable as company developers become associated with the project in the OSS community. The community develops a trusted relationship with a company developer and does not wish to see that relationship disturbed through the company moving developers on to other projects.

Global Recruitment: From Unknown to Known

The study also reveals that outsourcing to the OSS community provides a significant opportunity for companies to headhunt top developers, whereby community members become also employees of the customer—hence moving from outsourcing to a largely unknown OSS workforce toward recruitment of talented developers from the open source community. In this study we see a move from "unknown unknowns" as neither the customer nor the community are known to each other, to a scenario of "known knowns" as each gets to understand each other's position and build complementary skills. Essentially, the opensourcing model consists of a company (as opensourcing customer) and a community of individual developers and other companies with whom the customer interacts. This community provides potential for development cost savings, recruitment opportunities, and the capability of increasing innovation. Interestingly, while the community is expected to behave professionally and provide transparent authority structures (thus reducing the unknown factor), innovation potential is primarily to be expected from the unknown part of the workforce. Hence, there are forces in the ecosystem pulling in opposite directions: while cost-savings and innovation are facilitated by a large unknown workforce, trust-building and recruitment of community developers by the customer will tend to erode the unknown aspect.

Contribution to Research

We believe this is the first comprehensive study to focus on the opensourcing phenomenon. It thus represents an important step towards both (1) elaborating the IS offshoring research agenda to incorporate also this novel and unconventional approach to global sourcing and co-opetition, and (2) bringing the underexplored area of company-led OSS projects onto the OSS research agenda, in particular the ***liberation*** of hitherto proprietary software.

In terms of research method, the combination of a qualitative phase of in-depth case study interviews, complemented by a more quantitatively oriented large-scale survey of the phenomenon, has shown to be an effective way of triangulating from multiple sources to allow a comprehensive exploration of a novel phenomenon such as opensourcing. The rich understanding gained from in-depth qualitative data and analysis allowed us to propose a sophisticated initial model, and also to reinterpret and elaborate the results of the quantitative survey as in Table 6.

Our study is also notable in that most previous research on outsourcing has adopted a single perspective: the customer or the supplier (but most often focusing on the customer), the Koh et al. (2004) study being a notable exception, while this study considers both the customer and the community obligations. This is important since, although the obligations are symmetrical to a large extent, they are also complementary, and there are differing emphases from each perspective, both sides of which must be fulfilled to achieve a successful opensourcing arrangement.

While we contribute to a cumulative research tradition by building on the research of Koh et al., we also identify the significant ways in which opensourcing differs from conventional outsourcing—the lack of a formal contract and requirements specification driven by the customer as well as the absence of payment in the conventional sense, for example. Thus we position opensourcing as an exemplar of global sourcing (Carmel and Tjia 2005) in an offshoring context (i.e., as a setting where sourcing occurs from external collaborators as well as from offshore and onshore locations within the organization).

Open Source: From Replication to Innovation

Interestingly, open source is often assumed to be about replication rather than innovation. However, creativity and innovation is stimulated by multidisciplinary teams operating outside conventional organization structures (Garvin 1993; Goldman and Gabriel 2005; Inkpen 1996; Leonard-Barton 1995; Nonaka 1991). Certain characteristics have been identified, including ***autonomy*** (which forms the basis for self-organizing and widens the possibility that individuals will motivate themselves to form new knowledge), ***creative chaos*** (whereby individuals do not have to follow organizational rules, but are challenged to investigate alternatives and rethink assumptions), ***information redundancy*** (where indi-

viduals have information that goes beyond their immediate needs for a particular task), and ***requisite variety*** (whereby the individuals involved have the skill diversity to match the complexity and variety of the environment they face). Interestingly, these characteristics appear to be present in OSS communities and can thus be expected to play an important part in the success of opensourcing as a global sourcing strategy. Previous studies of OSS have shown that about 40 percent of OSS developers are employed within professional organizations,[3] suggesting an "open" community of about 60 percent. This latter cohort may provide the creative and innovative spark as OSS loses its image of being merely about imitation of proprietary products and innovation becomes the defining feature. In this study, the characteristics of the community facilitate innovation as community developers operating outside traditional organizational constraints can identify new functionality and develop the software in creative ways—innovation does indeed seem to happen elsewhere (Goldman and Gabriel 2005).

Open Source: From Individual to Company

The study reveals an ongoing shift from OSS as a community of individual developers to OSS as a community of commercial organizations, primarily small and medium-sized enterprises, operating as a symbiotic ecosystem in a spirit of co-opetition. Overall the goal seems to be to create such a sustainable open and trusted ecosystem where customers and community participants operate as equals with neither party dominating. Thus, in contrast to traditional outsourcing, opensourcing is not primarily about commissioning software development to a third party, but rather about engaging in long-term collaborative activities leading to a sustainable ecosystem. Since many of the collaborators in this ecosystem are likely to be the customer's competitors, the collaboration is necessarily done in a spirit of co-opetition (Brandenburger and Nalebuff 1996). Both customer and community members have a shared responsibility to actively contribute to the development and sustainability of the ecosystem. Thus, research on ecosystems (e.g., Schnase et al. 2003) might be usefully applied to elaborate this issue.

As mentioned earlier, much OSS research has focused inward on the OSS phenomenon itself and far less has been done on the organizational implications of OSS, and particularly on company-led OSS projects in an opensourcing strategy as here. It is our hope that this study will inspire researchers to investigate further this important area, both in terms of liberation and commercialization as well as other possible approaches and contexts.

[3]Lakhani and Wolf (2001) estimate 40 percent and Jorgensen (2001) estimates 43 percent of OSS developers are paid for their work.

Acknowledgments

This work has been financially supported by the Science Foundation Ireland Investigator Programmes B4-STEP (Building a Bi-Directional Bridge Between Software ThEory and Practice) and Lero – The Irish Software Engineering Research Centre, and by the EU FP6 projects CALIBRE (Coordination Action for Libre Software) and OPAALS (Open Philosophies for Associative Autopoietic Digital Ecosystems).

A paper covering part of the initial qualitative phase of this study was presented at ICIS 2006 in Milwaukee. The authors would like to thank Helena Holmström Olsson and Eoin Ó Conchúir for constructive collaboration on this initial part of the work.

References

Agarwal, R. 2000. "Individual Acceptance of Information Technologies," *Framing The Domains of IT Management: Projecting the Future Through the Past*, R. W. Zmud (ed.), Cincinnati, OH: Pinnaflex Press, pp. 85-104.

Anderson, N., and Chalk, R. 1998. "The Psychological Contract in Retrospect and Prospect," *Journal of Organizational Behavior* (19:S1), pp. 637-647.

Ang, S., and Slaughter, S. A. (2001) "Work Outcomes and Job Design for Contract versus Permanent Information Systems Professionals on Software Development Teams," *MIS Quarterly*, 25(3), pp. 321-350.

Argyris, C. (1960) *Understanding Organizational Behaviour*, London: Tavistock Publications.

Aubert, B. A., Patry, M., and Rivard, S. (2005) "A Framework for Information Technology Outsourcing Risk Management," *The DATA BASE for Advances in Information Systems*, 36(4), pp. 9-28.

Baskerville, R., and Pries-Heje, J. (1999) "Grounded Action Research: A Method for Understanding IT in Practice," *Accounting, Management and Information Technologies* (9:1), pp. 1-23.

Brandenburger, A. M., and Nalebuff, B. J. 1996. "*Co-Opetition: A Revolution Mindset That Combines Competition and Cooperation*," New York: Doubleday.

Bruns, A. 2007. "Produsage: Towards a Broader Framework for User-Led Content Creation," *Proceedings of the 6th ACM SIGCHI Conference on Creativity and Cognition*, Washington, DC, June 13-15.

Calloway, L., and Ariav, G. 1991. "Developing and Using a Qualitative Methodology to Study Relationships Among Designers and Tools," in H. Nissen, H. Klein, and R. Hirschheim (eds), *Information Systems Research: Contemporary Approaches and Emergent Traditions*, Amsterdam: Elsevier Publishers, pp. 175-194.

Carmel, E. 1999. *Global Teams: Collaborating Across Borders and Time Zones*, Upper Saddle River, NJ: Prentice-Hall.

Carmel, E. 2006. "Building Your Information Systems from the Other Side of the World: How Infosys Manages Time Zone Differences," *MISQ Executive* (5:1), pp. 43-53.

Carmel, E., and Agarwal, R. 2001. "Tactical Approaches for Alleviating Distance in Global Software Development," *IEEE Software* (18:2), pp. 22-29.

Carmel, E., and Tjia, P. 2005. *Offshoring Information Technology: Sourcing and Outsourcing to a Global Workforce*, Cambridge, NY: Cambridge University Press.

Chatterjee, D., Grewal, R., and Sambamurthy, V. 2002. "Shaping Up for E-Commerce: Institutional Enablers of the Organizational Assimilation of Web Technologies," *MIS Quarterly* (26:2), pp. 65-89.

Cheon, M. J., Grover, V., and Teng, J. T. C. 1995. "Theoretical Perspectives on the Outsourcing of Information Systems," *Journal of Information Technology* (10:4), pp. 209-219.

Dahlander, L., and Magnusson, M. G. 2005. "Relationships between Open Source Software Companies and Communities: Observations from Nordic Firms," *Research Policy* (34), pp. 481-493.

Davis, G. B., Ein-Dor, P., King, W. R., and Torkzadeh, R. 2004. "Information Technology Offshoring: Prospects, Challenges, Educational Requirements, and Curriculum Implications," *Proceedings of the 25th International Conference on Information Systems*, R. Agarwal, L. J. Kirsch, and J. I. DeGross (eds.), Washington, DC, December, pp. 1027-1038.

Dempsey, B., Weiss, D., Jones, P., and Greenberg, J. 2002. "Who Is an Open Source Software Developer?," *Communications of the ACM* (45:2), pp. 67-72.

Dinh-Trong, T., and Bieman, J. M. 2004. "Open Source Software Development: A Case Study of FreeBSD," in *Proceedings of the 10th International Symposium on Software Metrics*, IEEE Computer Society.

Ebert, C., and De Neve, P. 2001. "Surviving Global Software Development," *IEEE Software* (18:2), pp. 62-69.

Feller, J., and Fitzgerald, B. 2002. *Understanding Open Source Software Development*, London: Addison-Wesley.

Feller, J., Finnegan, P., Fitzgerald, B., and Hayes, J. 2008. "Bazaar by Design: Managing Inter-Firm Exchanges in an Open Source Service Network (OSSN)," in *Change in the Service Economy*, M. Barrett, E. Davidson, and J. I. DeGross (eds.), New York: Springer (forthcoming).

Fichman, R. G. 2004. "Going Beyond the Dominant Paradigm for IT Innovation Research: Emerging Concepts and Methods," *Journal of the Association for Information Systems* (5:8), pp. 314-355.

Fitzgerald, B. 2006. "The Transformation of Open Source Software," *MIS Quarterly* (30:3), pp. 587-598.

Gacek, C., and Arief, B. 2004. "The Many Meanings of Open Source," *IEEE Software* (21:1), pp. 34-40.

Gallivan, M. 2001. "Organizational Adoption and Assimilation of Complex Technological Innovations: Development and Application of a New Framework," *The DATA BASE for Advances in Information Systems* (32:3), pp. 51-85.

Garvin, D. 1993. "Building a Learning Organization." Harvard Business Review (71:4), pp. 78-91.

German, D. M. 2004. "The GNOME Project: A Case Study of Open Source, Global Software Development," *Software Process: Improvement and Practice* (8:4), pp. 201-215.

Ghiselli, E., Campbell, J., and Zedeck, S. 1981. *Measurement Theory for the Behavioral Sciences*, San Francisco: W. H. Freeman.

Ghosh, R., Glott, R., Kreiger, B., and Robles-Martinez, G. 2002. The Free/Libre/Open Source Software Developers Survey," available from http://www.infonomics.nl/FLOSS/report/.

Goetz, J., and LeCompte, D. 1984. *Ethnography and Qualitative Design in Educational Research*, Orlando, FL: Academic Press.

Goles, T., and Chin, W. W. 2005. "Information Systems Outsourcing Relationship Factors: Detailed Conceptualization and Initial Evidence," *The DATA BASE for Advances in Information Systems* (36:4), pp. 47-67.

Goldman, R., and Gabriel, R. P. 2005. *Innovation Happens Elsewhere: Open Source as Business Strategy*, San Francisco: Morgan Kauffman Publishers.

Gottschalk, P., and Solli-Sæther, H. 2005. "Critical Success Factors from IT Outsourcing Theories: An Empirical Study," *Industrial Management & Data Systems* (105:6), pp. 685-702.

Gorman, M. 2003. "A Design, Implementation and Algorithm Analysis of a Virtual Memory System for Linux," personal communication.

Grover, V., Teng, J. T. C., and Fiedler, K. D. 2002. "Investigating the Role of Information Technology in Building Buyer-Supplier Relationships," *Journal of the Association for Information Systems* (3:7), pp. 217-245

Guba, E. 1981. "Criteria for Assessing the Trustworthiness of Naturalistic Inquiries," *Educational Communication and Technology* (29), pp. 75-92.

Hair, J., Anderson, R., Tatham, R., and Grablowsky, B. 1984. *Multivariate Data Analysis with Readings*, New York: Macmillan Publishing.

Hann, I., Roberts, J., Slaughter, S., and Fielding, R. 2002. "Why Do Developers Contribute to Open Source Projects? First Evidence of Economic Incentives," paper presented at the Second Workshop on Open Source Software Engineering, Orlando, FL.

Herbsleb, J. D., and Grinter, R. E. 1999. "Splitting the Organization and Integrating the Code: Conway's Law Revisited," in *Proceedings of the 21st International Conference on Software Engineering*, Los Angeles, California.

Hutcheson, G., and Sofroniou, N. 1999. *The Multivariate Social Scientist*, London: Sage Publications.

Inkpen, A. C. 1996. "Creating Knowledge through Collaboration," *California Management Review* (39:1), pp. 123-140.

Jorgensen, N. 2001. "Putting it All in the Trunk: Incremental Software Development in the FreeBSD Open Source Project," *Information Systems Journal* (11:4), pp 142-157.

Kaiser, H. F. 1974. "An Index of Factorial Simplicity," *Psychometrika* (39), pp. 31-36.

Kaplan, B., and Duchon, D. 1988. "Combining Qualitative and Quantitative Methods in IS Research: A Case Study," *MIS Quarterly* (12:4), pp. 571-587.

Kass, R., and Tinsley, H. 1979. "Factor Analysis," *Journal of Leisure Research* (11), pp. 120-138

Klein, H. K., and Myers, M. D. 1999. "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems," *MIS Quarterly* (23:1), pp. 67-94.

Koh, C., Ang, S., and Straub, D. W. 2004. "IT Outsourcing Success: A Psychological Contract Perspective," *Information Systems Research* (15:4), pp. 356-373.

Lakhani, K., and Wolf, R. 2001. "Does Free Software Mean Free Labor? Characteristics of Participants in FOSS Communities," available at http://web.sourceforge.com/.

Leonard-Barton, D. 1995. *Wellsprings of Knowledge: Building and Sustaining the Sources of Innovation*, Boston: Harvard Business School Press.

Lerner, J., and Tirole, J. 2002. "Some Simple Economics of Open Source," *The Journal of Industrial Economics* (50:2), pp. 197-234.

Marshall, C., and Rossman, G. 1989. *Designing Qualitative Research*, Thousand Oaks, CA: Sage Publications.

Miles, M., and Huberman, A. 1994. *Qualitative Data Analysis: An Expanded Sourcebook* (2nd ed.), Thousand Oaks, CA: Sage Publications.

Millar, C., Choi, C. J., Russell, E. T., and Kim, J. B. 2005. "Open Source Communities: An Integrally Informed Approach," *Journal of Organizational Change Management* (18:3), pp. 259-268.

Miranda, S. M., and Kavan, C. B. 2005. "Moments of Governance in IS Outsourcing: Conceptualizing Effects of Contracts on Value Capture and Creation," *Journal of Information Technology* (20:3), pp. 152-169.

Mockus, A., Fielding, R. T., and Herbsleb, J. D. 2002. "Two Case Studies of Open Source Software Development: Apache and Mozilla," *Transactions on Software Engineering Methodology* (11:3), pp. 309-346.

Mockus, A., and Herbsleb, J. D. 2002. "Why Not Improve Coordination in Distributed Software Development by Stealing Good Ideas from Open Source?," in *Meeting Challenges and Surviving Success: The Second Workshop on Open Source Software Engineering*, pp. 19-25, available online at http://opensource.ucc.ie/icse2002/MockusHerbsleb.pdf

Nonaka, I. 1991. "The Knowledge-Creating Company," *Harvard Business Review* (69:6), pp. 96-104.

O'Mahony, S. 2005. "Non-Profit Foundations and Their Role in Community-Firm Software Collaboration," in *Perspectives on Free and Open Source Software*, J. Feller B. Fitzgerald, S. Hissam, and K. Lakhani (eds.), Cambridge, MA: MIT Press, pp. 393-414.

Orlikowski, W. J. 1993. "CASE Tools as Organizational Change: Investigating Incremental and Radical Changes in Systems," *MIS Quarterly* (17:3), pp. 309-340.

Patton, M. Q. 1990. *Qualitative Evaluation and Research Methods* (2nd ed.), Newbury Park, CA: Sage Publications.

Pavlou, P. A., and Gefen, D. 2005. "Psychological Contract Violation in Online Marketplaces: Antecedents, Consequences, and Moderating Role," *Information Systems Research* (16:4), pp. 372-399.

Piccoli, G., and Ives, B. 2003. "Trust and the Unintended Effects of Behavior Control in Virtual Teams," *MIS Quarterly* (27:3), pp. 365-395.

Raghu, T. S., Jayaraman, B., and Rao, H. R. 2004. "Toward an Integration of Agent- and Activity-Centric Approaches in Organizational Process Modeling: Incorporating Incentive Mechanisms," *Information Systems Research* (15:4), pp. 316-335.

Raymond, E. S. 1999. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, Sebastopol, CA: O'Reilly.

Robles-Martinez, G., Scheider, G., Tretkowski, I., and Weber, N. 2001. "Who Is Doing It?," available online at http://widi.berlios.de/paper/study.html.

Scacchi, W. 2002. "Understanding the Requirements for Developing Open Source Software Systems," *IEE Proceedings–Software* (149:1), pp. 24-39.

Schnase, J. L., Cushing, J., Frame, M., Frondorf, A., Landis, E., Maier, D., and Silberschatz, A. 2003. "Information Technology Challenges of Biodiversity and Ecosystems Informatics," *Information Systems* (28:4), pp. 339-345.

Straub, D. 1989. "Validating Instruments in MIS Research," *MIS Quarterly* (13:2), pp. 147-169.

Strauss, A., and Corbin, J. 1998. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques* (2nd ed.), Thousand Oaks, CA: Sage Publications.

Tabachnik, B., and Fidell, L. 1996. *Using Multivariate Statistics* (3rd ed.), New Yorik: Harper Collins.

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.

Wallace, R., and Mellor, C. 1988. "Non-Response Bias in Mail Accounting Surveys: A Pedagogical Note," *British Accounting Review* (20:2), pp. 131-139.

Wang, E. T. G. 2002. "Transaction Attributes and Software Outsourcing Success: An Empirical Investigation of Transaction Cost Theory," *Information Systems Journal* (12:2), pp. 153-181.

Webb, E., Campbell, D., Schartz, R., and Sechrest, L. 1996. *Unobtrusive Measures: Non-Reactive Research in the Social Sciences*, Chicago: Rand McNally.

Wheeler, D. 2004. "Why Open Source Software/Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!," available online at http://www.dwheeler.com/oss_fs_why.html.

Yin, R. K. 1994. *Case Study Research: Design and Methods* (2nd ed.), Thousand Oaks, CA: Sage Publications.

About the Authors

Pär J. Ågerfalk is a professor of Computer and Systems Science at Uppsala University, Sweden, where he holds the Chair in Computer Science in Intersection with Social Sciences. He received his Ph.D. in Information Systems Development from Linköping University and has held full-time positions at Örebro University, University of Limerick, Jönköping International Business School, and Lero – The Irish Software Engineering Research Centre, where he is also currently a senior research fellow. His work on open source software, global software development, method engineering, information systems design, and conceptual modeling has appeared in a number of leading information systems journals and conferences and he is currently an associate editor of the *European Journal of Information Systems*.

Brian Fitzgerald received his Ph.D. from University of London and currently holds the Frederick A Krehbiel II Chair in Innovation in Global Business and Technology at the University of Limerick, where he is a Science Foundation Ireland Principal Investigator. His research interests include open source software, agile methods, and distributed software development. Over the years he has published 10 books and his work has appeared in most leading IS journals and conferences. He has served as a guest senior editor for several prominent journals, including *Information Systems Research*, *Communications of the ACM*, *European Journal of Information Systems*, and *Information Systems Journal*.

Appendix A

Schematic Overview, Interview Guide, and Coding Examples

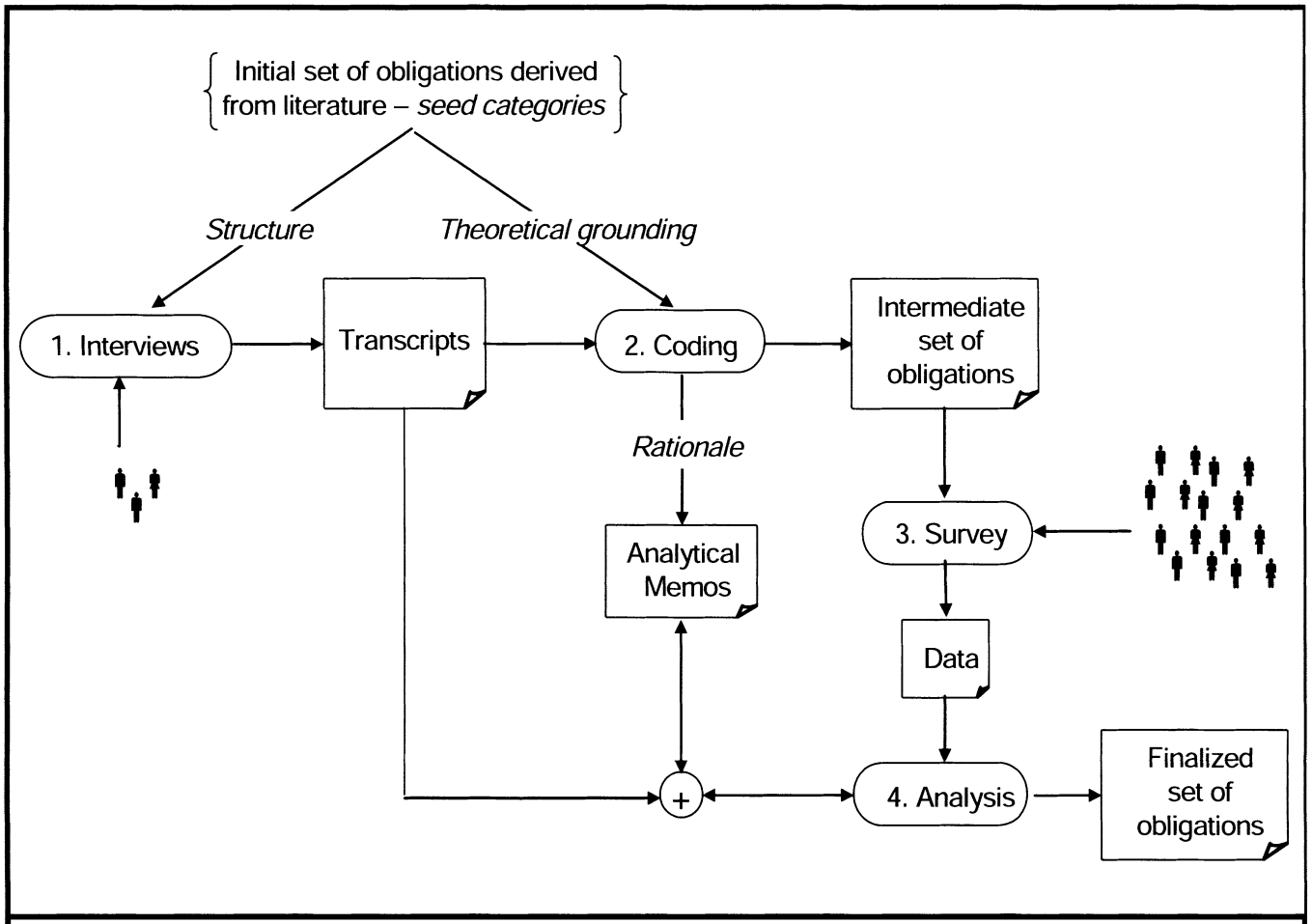

Figure A1. Schematic Overview of the Research Process

Table A1. Interview Guide Used in Qualitative Phase

Customer Obligations	***Community Obligations***
(1) *Explicit and comprehensive requirements specifications for the services covered by the outsourcing project* – Although initially, requirements specifications were not part of the OSS landscape, this appears to be increasingly the case.	(1) *Clear authority structures which delineate the decision-making rights and reporting structures in the project* – Given the absence of normal organizational authority, the "benevolent dictatorship," and meritocracy in OSS projects is necessary.
(2) *Prompt feedback to supplier community with no unreasonable delays* – Although payment in the monetary sense is usually (but not always) not a factor in OSS development, prompt feedback by peer developers and users is critical.	(2) *Taking charge in terms of completing the job and solving problems independently, with minimal customer involvement* – OSS development has traditionally been characterized by developer independence and prompt problem solving, although customer involvement in terms of user feedback has been a marked feature.
(3) *Close project monitoring with active overseeing of project progress, attending project meetings and regular discussions* – Again, project monitoring is increasingly a part of the more commercially focused OSS development process.	(3) *Effective human capital management in assigning high-quality staff to work on the project, and seeking to minimize staff turnover during the project* – OSS developers are acknowledged to be high quality, and exhibit strong loyalty to projects due in part to avoidance of project forking and the freedom to choose what development tasks to work on.
(4) *Project ownership to ensure that senior management provides strong leadership, support, and commitment toward the project* – Given the high risk, radical initiative that OSS deployment represents, strong project ownership and management championship may be necessary.	(4) *Building effective inter-organizational teams – investing time and effort to foster a good working relationship in the customer and community project team* – Community networks of OSS companies are becoming a common mode of delivering whole product OSS offerings to customers.
(…) *Other possible obligations.*	(5) *Effective knowledge transfer in educating the customer in the skills, knowledge, and expertise associated with using the outsourced system or service* – The user/developer relationship is very close in OSS thus facilitating knowledge transfer.
	(…) *Other possible obligations.*

Table A2. Coding Examples

Initial Set of Obligations	***Transcripts (Excerpts)***	***Analytical Memos***	***Emerging Set of Obligations***
...	...	...	...
Explicit and comprehensive requirements specification	When kicking off the project in the open source community, it is about stating the overall goal and *the top-level requirements* you are trying to achieve. Then it is *driven by consensus.*	Although initially, requirements specifications were not part of OSS, this appeared to be increasingly the case. High-level requirements evolve.	Achieving consensus on development roadmap.
...	...	...	...
N/A[†]	There are a lot of open source projects that don't go anywhere, even though they would have built good code. It also *needs to be pushed* so that it gets noticed and used by other projects, documented, and marketed. This is a big overhead, and *commercial companies have structures in place to help achieve that.* What I am looking for is a multiplier. I am providing a piece of my work. I know they already have a useful chunk of work. But adding my bit to their larger existing work in *cooperative ways creates something of greater multiplied use* to everybody, including myself.	Companies have marketing experience and resources that OSS communities typically lack.	Marketing project to increase visibility.
	...	...	...

[†]Note that there is not a one-to-one mapping between the two sets of obligations, which is a natural consequence of the open-ended analysis in which new obligations were also allowed to emerge.