Open-Sourcing as Offshore Outsourcing Strategy

Pär J Ågerfalk[1,2], Brian Fitzgerald[1], Helena Holmström[1,3], Eoin Ó Conchúir[1]
[1]University of Limerick, Ireland
[2]Örebro University, Sweden
[3]IT University of Gothenburg, Sweden
{par.agerfalk, bf, helena.holmstrom, eoin.oconchuir}@ul.ie

Abstract. This paper presents a psychological contract perspective on the use of open source as an offshore outsourcing strategy – open-sourcing as we term it here. Building on previous research on IS outsourcing, a theoretical framework for understanding commercial software organizations involvement in open source software (OSS) is derived. The framework is used in a qualitative case study involving a commercial organization, the Irish-based global middleware company IONA, as the customer, and representatives from the open source community, as suppliers of services to IONA. The study reveals an ongoing shift from OSS as community of individual developers to OSS as community of commercial organizations, primarily SMEs. It also reveals that outsourcing to the OSS community provides ample opportunity for companies to headhunt top developers – hence moving from outsourcing to a largely unknown OSS workforce towards recruitment of talented developers from the open source community. In a similar fashion, the process allows the development community to get to know the customer organization better. Overall, the key watchwords for open-sourcing are partnership, building of mutual trust, flexibility, tact and complementariness: The customer and community need to establish a trusted partnership of shared responsibility in building an overall ecosystem to deliver the product. The customer has to be flexible in accepting consensus on the development priorities and the functionality that will be built in. The community must be flexible in affording more transparency into the development process. Also, the complementary skills offered by each stakeholder are key to successfully nurturing the ecosystem.

Introduction

While manufacturing industries have been offshoring manufacturing processes to lower-cost destinations for thirty years or more, it is really only since the mid 1990's that a significant portion of software development work is being performed offshore (Carmel, 1999; Minevich et al. 2005). There are many potential advantages to be gained in offshoring software development, including reduced development 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 (Ågerfalk et al., 2005). Of these, most emphasis has been placed on potential cost savings, which accrue largely due to the availability of a trained pool of development staff at a lower salary base. To take an example, the average annual base salary of a software development engineer in India in 2004 was less than one-seventh of the median annual salary of a US software engineer. Such potential savings have helped fuel the amount of work being offshored from high-cost countries such as the US, UK and Scandinavian countries to lower cost economies such as India, China and Malaysia. The U.N. World Investment Report 2004 predicted that offshore outsourcing of IT-enabled business processes will increase 18-fold to almost $25 billion by 2007 (United Nations, 2004).

Given that the primary force driving offshore outsourcing appears to be cost savings, it is perhaps natural that companies might eventually focus on open source software (OSS) as a potentially even cheaper alternative, as there are significant cost savings associated with the acquisition of open source software (Fitzgerald and Kenny, 2003; Wheeler, 2004). Carmel and Tija (2005) have characterized offshore outsourcing as 'outsourcing to a global workforce', in this study we investigate the emerging trend towards open-sourcing, that is, outsourcing to a global but largely *unknown* workforce. To do this we 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 outsourcing organizations and members of the open source community. Building on Koh et al.'s (2004) results, our research question is thus: *What are the critical customer-community obligations in an open-sourcing relationship?*

Koh et al. (2004) 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 inwards 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.

This study focuses on the implications of open-sourcing, as we term it – that is, adopting open source as an offshore outsourcing strategy for the software devel-

opment process in organizations. One of the novel features of the study is that it considers both sides of the process – that is, the organization which is 'commissioning' the open-sourcing, 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. (2004) in two significant ways. First, it considers the unexplored concept of open-sourcing, as opposed to 'traditional' outsourcing. Second, Koh et al. (2004) focused on onshore outsourcing while this work considers offshore outsourcing, as implicated by the open-sourcing strategy.

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 open-sourcing. We then focus on PCT, explaining why we have selected it for this study, and how it has been used. Following this, our research approach is discussed, and the case study findings presented and analysed. Finally, the overall conclusions are presented.

Basic Terminology

While the terms offshoring and outsourcing are often used almost as synonyms, here 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 and any particular activity can thus 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.

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

Figure 1: Offshoring versus outsourcing.

Open source software may be defined as software released under the terms of a license which 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 open-sourcing as a software development model. Similar to outsourcing, the open source software 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 be assumed to be global (Millar et al., 2005). Hence, open-sourcing, by definition, falls in the offshore outsourcing category of Figure 1.

A PCT Perspective on Open-Sourcing

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; Piccoli and Ives, 2003; Koh et al., 2004; Raghu et al., 2004; Miranda and Kavan, 2005; Pavlou and Gefen, 2005). 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., 2004:357) 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 (vendor's or outsourcer's) commonly adopted in outsourcing studies (Koh et al., 2004). Second, psychological contracts are distinct from legal contracts. Specifically they encompass people's beliefs about both written terms and unwritten implicit terms. Third, it promotes an individual level of analysis, rather than the more common inter-organizational level of analysis.

These three issues are also central to the OSS concept. Firstly, 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 fulfilment 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 which contravenes these values of reciprocity and mutual obligations (Von Hippel and Von Krogh, 2003).

Secondly, in relation to the PCT focus on psychological contracts (which differ from legal ones as they encompass both written and unwritten terms), most 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 standards, 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 behaviour (Gorman, 2003; Raymond, 1999).

Finally, in relation to the focus on the individual level of analysis, research suggests that OSS developers display more loyalty to the OSS phenomenon than to the organizations where they may be employed (Feller and Fitzgerald, 2002). Also, the signalling incentives identified by Lerner and Tirole (2001) as the basic motivation for developers to contribute to OSS projects apply primarily at the level of the individual.

In their study, Koh et al. (2004) used an initial qualitative case study approach to derive a set of obligations that customers and suppliers need to fulfil 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. However, the 'supplier' entity becomes the open source community in our study. Below, we consider the obligations and discuss their specific relevance to an OSS context.

Customer Obligations

Koh et al (2004) 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 toward 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. 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 vertical domains where a developer may not perceive an itch worth scratching (Fitzgerald, 2006). 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, the Lerner and Tirole (2001) 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. Also given that payments in the normal sense are not part of the equation, then the issue of unreasonable withholding of 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 v. 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 which didn't reflect the heterogeneity of the OSS development landscape even at the time. In recent times, OSS development has become more formalized. This is evident in the regular project meetings which are now a feature of a number of popular open source products, such as the Apache conferences in the US and Europe, the Zope/Plone development project meetings, and the GNOME annual project conferences (German, 2003) 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.

Supplier Obligations

Koh et al (2004) 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 inter-organizational teams – invest 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

Again, these are discussed below in terms of their specific relevance in an OSS context.

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 “benevolent dictatorship” as initially suggested by Raymond (1999). 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. 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, is a strong community norm which acts against developer turnover on projects.

Building Effective Inter-Organizational 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 (Moore 1999). 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 inter-organizational 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, docu-

mentation etc. Thus, a close working relationship can emerge between developer and user.

Summary of Obligations

As illustrated above, the obligations identified and verified by Koh et al. (2004) map well to both the customer and the community in an OSS context. The manner in which they have been refined for an OSS context is summarized in Table I.

Customer obligation to provide:
(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.
(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.
(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.
(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.
OSS community obligation to provide:
(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) *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) *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) *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.
(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.

Table I. Summary of Customer and Community Obligations in an OSS Context.

Research Approach

Much of the research on OSS to date has focused inward on the phenomenon itself, studying the motivations of individual developers to contribute to OSS projects, or investigating the characteristics of specific OSS products and projects, for example. Far less has been done in looking outward at the organizational use and leverage of OSS in practice. Given this emphasis and the relative newness of the open-sourcing 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, an interpretivist approach which sought to inductively develop a richer understanding based on a deep analysis of a single case was deemed appropriate, as this "revelatory case" (Yin, 1994) may provide such rich insight. The case selected for the study was the Celtix project, an open source Java Enterprise Service Bus (ESB) sponsored by IONA Technologies.

IONA Technologies – The Celtix Project

IONA Technologies was founded as a campus company at Trinity College Dublin in 1991, and provides products and services to help organizations build B2B enterprise portals. IONA, a NASDAQ-quoted company, is headquartered in Dublin, Ireland, with U.S. headquarters in Waltham, Massachusetts and offices worldwide. 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 co-exist 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 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.

Data Collection and Analysis

Data was gathered over a 10-month period from July 2005 to April 2006, and a number of sources were drawn on (see Table II). These ranged from workshops, to a series of interviews, both formal face-to-face and informal telephone interviews. An interview protocol guide was developed based on the obligations iden-

tified above. 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'). These interviews served to give a good strategic overview of the project and the high level obligations that were in place. Both these individuals initially identified key figures in the project 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 studies of open source developers up to now have relied on anonymous surveys, 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 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. An interview protocol guide was prepared, both to act as an *aide memoire* during interviews, and also to act as a backup if interviewees were unwilling to be recorded. This was emailed to interviewees in advance to allow them an insight into the overall issues we wished to focus on. Informal follow-up interviews took place to clarify and refine issues that emerged following the interview transcription process. Interview transcription generated a total of 63 pages comprising 28,787 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 community participants and other researchers over a series of workshops.

Workshops	**Interviews**	**Supplementary Sources**
09/05 Presentation and discussion of Celtix business model and strategy. 04/06 Workshop presentation on open-sourcing strategy.	• 07/05–04/06 Multiple interviews with: o SB, Chief Scientist, IONA o JPL, Chairman, ObjectWeb o RB, Admin IONA o DM, Open Source Program Director, IONA o MMcM, Manager, STP o AS, Celtix project manager o CS, ObjectWeb developer	IONA and ObjectWeb maintain detailed and comprehensive web portals for the Celtic project. We also had access to mailing lists and project development wiki pages.

Table II. Summary of Customer and Community Obligations in an OSS Context.

For data analysis, a primarily qualitative grounded theory (GT) approach was adopted (cf. Corbin & Strauss, 1990; Miles & Huberman, 1994). The GT approach recognizes that social phenomena are complex and seeks to develop theory systematically in an intimate relationship with the data. Interviews were transcribed and then coded according to the constructs represented by the customer and community obligations derived earlier, and analytical memos were written as patterns and themes emerged from these field notes.

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 & Duchon, 1988). This problem was addressed in a number of ways. Firstly, the grounded theory method of data analysis explicitly recognizes this problem of subjective data interpretation, and to address it, prescribes 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). Secondly, 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 and ObjectWeb were active participants in an 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.

Research Findings and Discussion

Here we discuss the obligations raised by the interviewees. 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.

Customer Obligations

We begin with the customer obligations. These include the four that we initially derived and the additional ones identified during interviews. We begin the discussion of each of these customer obligations by focusing on the customer (i.e IONA) interviewee responses, and then consider the community interviewee views on the same obligation.

Explicit and Comprehensive Requirements Specifications

Explicit and comprehensive requirements specifications was identified as important by all IONA interviewees. It was suggested that more formalized specifica-

tions were increasingly the norm as open source "evolved towards more vendor-led projects". However, interviewees also stressed the manner in which requirements specification here differed from traditional development. Vendors 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 vendor pushes their own agenda too much in driving the development agenda, 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 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 open-sourcing project.

Several interviewees emphasized the need for marketing the attractiveness of the project and the desired functionality. This has a two-fold purpose. Firstly, it helps achieve consensus on the required functionality as discussed above. But more importantly, this vision helps attract developers and ensure the vibrancy of the project, which cannot be taken for granted in an open source project. After all, 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 vendors have structures in place to help achieve that."

The community interviewees also agreed that more formalized specifications and documentation were a critical part of the open-sourcing mode of work. However, one interviewee identified the cost of achieving this as a significant problem for the community, as such practices have been somewhat alien to the OSS community in the past.

Overall, the customer obligation to provide a high-level requirements specification is important in open-sourcing. However, it clearly differs from the process that arises within a conventional outsourcing relationship where the customer can unilaterally dictate the required functionality.

Prompt Feedback to Supplier Community

While payment in the OSS world is quite different from payment in traditional development, the issue of prompt feedback was raised. This was generally accomplished through mailing lists and the project wiki pages. The kudos which could arise was mentioned. Those who were active in giving feedback would be listed as contributors without having contributed actual code. Customer interviewees stressed the importance of not being perceived as free-riding, in taking advantage of the open source community effort without providing as much as possible in return. This is especially important in open-sourcing as the customer goal of creating value, while obviously quite legitimate, must at the same time be achieved without transgressing the values of the community.

Also, in order to more fully engage with the community, IONA established a full time position – Open Source Program Director – who would ensure that issues relevant to the open source community would receive prompt attention. Also, given that the interaction on development tended to be "very much techie to techie" as one interviewee put it, the project management committee is chaired by a Distinguished Engineer at IONA 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.

Close Project Monitoring

Given that there is a strong external element in open-sourcing, 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 (think of the Microsoft Vista and Longhorn projects!). Also, the frequency of 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 had different norms and approaches in this area, and the customer had to be flexible and prepared to adapt to the particular regime in vogue in the open source community.

The OSS community interviewees also stressed the importance of clarifying the governance of the project. Open source development is usually characterized by a clear project authority structure based on a meritocracy. Thus, it is hardly surprising that the community would expect this of the customer in leading the project, in that it does not depart from the usual norms for OSS development.

Project Ownership

The project ownership obligation is closely related to the previous project monitoring obligation. Again, the customer interviewees identified with this. The fact that a full-time position as Open Source Program Director had been created is evidence of strong leadership by the customer. This person is responsible for engaging with the community. One customer interviewee stressed the radical change in mindset represented by open-sourcing, 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 that they developed. In the open-sourcing model, their work could appear to be benefiting the open source community, and not leading to an obvious direct remuneration. Thus, as it was more a strategic initiative with a different business model, top management championship was necessary. However, the Celtix project also grows the market for IONA, enabling additional support contract revenue.

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.

As strong project ownership is very much part of OSS development anyway, community interviewees expected this to be the case, and talked about the manner in which project ownership could be manifest, suggesting that the project leaders "have to show example, have to prove that they are the best".

Further Customer Obligations

We also asked interviewees to identify any additional customer obligations that they felt might arise. Most suggestions elaborated the obligations we already had identified and discussed. However, some additional candidates were suggested, including the following:

Expertise to Create a Commercial Product Offering – Several customer interviewees identified the need to be able to create a professional OSS product and subsequently market 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. This was mentioned to some extent above in relation to the initial requirements specification obligation.

Licensing and Clear Intellectual Property Policy – 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 (GPL) 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. Over time, IONA had perceived the need to be even more open to other companies and communities, and hence dual licensed Celtix under the Eclipse Public License also.

Community Obligations

We now turn towards the open source community obligations. Again, these include the five that were initially derived and the additional ones identified during interviews.

Clear Authority Structures

Community representatives identified clear authority structures as important – indeed, one stressed that clear authority structures are "not only important, but mandatory". It was argued that since more and more professional people are involved in OSS, the community is expected to show the same level of quality and transparency as could be expected from any professional organization.

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

> "It would be good to ensure that 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. Firstly, 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 (Feller and Fitzgerald, 2002), and with the increasing interest in the so-called 'whole product approach' (Fitzgerald 2006), this is expected to be increasingly important, as pointed out by the Open Source Program Director at IONA. Secondly, they provide for consistent terminology within and across projects which makes sure people are "on the same page, and really focus on innovation".

Taking Charge in Terms of Completing the Job

Although the community taking charge and living up to expectations was believed to be essential by community and customer interviewees alike, this obligation becomes somewhat blurred in the open-sourcing context. Since part of the development community in our study are paid IONA employees, the customer does have the power to manage part of the development effort more directly than would be possible in a traditional offshore outsourcing context. Currently, there seems to be a feeling that "there will always be customer involvement". However,

as the open-sourcing phenomenon matures, "there will be a lot more of developer independence".

Interestingly, OSS community members do not necessarily see themselves as vendors commissioned by a customer in the traditional sense. Rather, customer and community are seen as "part of the same ecosystem". In fact, Object Web sees its members not as OSS developers but as "ecosystem developers".

Effective Human Capital Management

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 OSS "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.

IONA, as the open-sourcing customer, expects the OSS model to attract "high calibre people". The Open Source Program Director 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."

Building Effective Inter-Organizational Teams

There seems to be a definite trend towards more organized open source communities, such as that of Object Web. Hence, building effective inter-organizational teams is to a large extent what open-sourcing is all about. As indicated above, part of this trend is the merging of customer and community into an "ecosystem": "I don't consider IONA as a customer. Iona is a member" was how the situation was described by the Chairman Object Web. Open-sourcing is thus not just about building good working relationships between customer and vendor. It is about "ecosystem development". Hence, although "everybody knows there are business reasons why people are there", 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.

Effective Knowledge Transfer

In open-sourcing, the software developed is typically not aimed for end-users but is more likely to be tools and infrastructure components. Consequently, the cus-

tomer 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". This was emphasized by the Object Web Chairman who asserted that "I don't speak about education or anything like that, I speak about exchange between researchers". This view was acknowledged by Open Source Program Director at IONA who referred 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.

Further Community Obligations

We asked both community and customer interviewees about further community obligations that they felt had not been covered. The most significant one was identified by a Project Manager at IONA:

Process Transparency – As a complement to clear authority structures, the lack of a traditional written outsourcing contract means that an open source community must be clear about what they are doing. Interestingly, this mirrors the project monitoring and transparency obligation that is expected on the part of the customer.

Conclusion

Table III summarizes the refined list of obligations in the context of open-sourcing. Overall the key watchwords for customers are flexibility, tact, partnership, building of mutual trust, and complementariness. The customer must be prepared to compromise at all stages: For example, rather than just providing a requirements specification for desired functionality, the process requires that development priorities be consensually agreed upon by the customer and community. Also, the vendor has to provide complementary expertise in relation to product commercialization and marketing. Hence we have modified the obligation to reflect these issues. Furthermore, the customer must 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 creating a successful business model for itself while not transgressing the community values which seek to benefit the overall community. Also, the standard practices that tend to apply in the customer company may need to change. For example, more clarity is required in relation to release milestones, also more frequent product releases are likely rather that artificially separating functionality on the basis of the price that can be charged. Furthermore, the policy in vendor companies of rotating developers onto different projects after a period of several

months may not be sustainable as developers become associated with the project in the OSS community.

Although the OSS community indeed differ from a traditional outsourcing partner, many of the supplier obligations seem to apply also in an open-sourcing context. Particularly, clear authority structures that make the democratic decision making process and development model transparent are vital. In general, the OSS development model is assumed to attract high-quality developers, and there is evidence that this is also the case. Given the community spirit associated with OSS development, it is not surprising that the OSS community is expected to contribute actively to the creation of an ecosystem manifested in deep collaboration with the customer. Since open-sourcing means not just commissioning to the OSS community but also to contribute back to that community, the ecosystem exists within a context of 'co-opetition'. This also means that ongoing knowledge transfer is greatly facilitated, partly due to the "techie-to-techie" nature of collaboration.

Customer obligation to provide:

(1) *Holistic approach to requirements specification and product commercialization and marketing.* High-level requirements identified and consensus achieved with the community on implementation of these requirements. Customer also providing overall marketing to attract other developers and adopters to ensure a vibrant and successful project.

(2) *Prompt feedback to supplier community.* Meaningful feedback required, facilitated by the "techie-to-techie" nature of interaction, explicit acknowledgement of those active in feedback as contributors.

(3) *Close project monitoring with active overseeing of project progress.* Vital given the external and public nature of OSS. However, again the customer has to be prepared to adapt to the particular project monitoring regime in use in the OSS community.

(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 is necessary. However, a delicate equilibrium must be maintained as the development community may be reluctant to become involved if it is perceived as too much a customer-led initiative.

(5) *Clear IP/Licensing Policies*. Given the importance of licensing in open source and patents and IP in proprietary software companies, these issues come to the fore and must be resolved to the satisfaction of both customer and open source community.

OSS community obligation to provide:

(1) *Clear and democratic authority structure and process transparency*. Given the lack of a written contract, this was seen as important, and also facilitated by the increased

involvement of traditional professionals in open-sourcing. Democracy and shared responsibility are also emphasized.
(2) *Taking responsibility and living up to expectations.* Again, the initial characterization that the community would solve problems independently became blurred as it is very much a case f collective responsibility with much input from the customer side. However, the OSS community has to deliver according to its capabilities.
(3) *Effective human capital management in assigning high-quality staff to work on the project, and seeking to minimize staff turnover during the project.* The model allows customers to identify high quality developers who may be employed by the customer. Staff turnover is interesting in that the community exercises a strong pressure on the customer not to rotate its own developers across projects.
(4) *Building and maintaining an effective inter-organizational ecosystem.* Community networks of OSS companies operating in an overall ecosystem in a spirit of co-opetition are becoming a common mode of delivering 'whole product' OSS offerings to customers.
(5) *Effective knowledge transfer*. Given the "techie-to-techie" nature of the interaction, knowledge transfer is greatly facilitated.

Table III. Summary of Refined Customer and OSS Community Obligations in an Open-Sourcing Context.

The study reveals an ongoing shift from OSS as community of individual developers to OSS as community of commercial organizations, primarily SMEs, operating as a symbiotic ecosystem in a spirit of co-opetition. At the beginning of the project, the open source community represented an 'unknown' to the vendor, and indeed vice-versa, as IONA were quite 'unknown' to the community, and needed to achieve a position of being trusted by the open source community as capable of successfully sponsoring an open source project. The study also reveals that outsourcing to the OSS community provides ample opportunity for companies to headhunt top developers – hence moving from outsourcing to a largely unknown OSS workforce towards recruitment of talented developers from the open source community. To paraphrase Donald Rumsfeld's torturing of the Johari Window – there are known knowns, known unknowns, and unknown unknowns. 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 builds complementary skills.

Acknowledgements

This research was supported by grants from Science Foundation Ireland for the B4-STEP (Building a Bi-directional Bridge Between Software ThEory and Prac-

tice) and Lero – The Irish Software Engineering Research Centre projects, and from the EU to the CALIBRE project.

References

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

Ågerfalk P J, Fitzgerald B, Holmström H, Lings B, Lundell B and Ó Conchúir E (2005) A Framework for Considering Opportunities and Threats in Distributed Software Development, In Proceedings of the International Workshop on Distributed Software Development (DiSD 2005), Austrian Computer Society, Paris, 29 August 2005, pp. 47–61.

Anderson N and Chalk R (1998) The Psychological Contract in Retrospect and Prospect, Journal of Organizational Behavior, 19(S1), 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), 321-350.

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

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

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): 65-89.

Feller J and Fitzgerald B (2002) Understanding Open Source Software Development, Addison-Wesley, London, UK.

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)

Fitzgerald, B. (2006) The Transformation of Open Source Software, MIS Quarterly, forthcoming.

Fitzgerald, B. and Kenny, T. (2003) Open Source Software in the Trenches: Lessons from a Large Scale Implementation, Proceedings of 24th International Conference on Information Systems (ICIS), Seattle, December 2003

Gallivan, M (2001) Organizational adoption and assimilation of complex technological innovations: development and application of a new framework, Data Base, Vol 32, No 3, pp. 51-85.

Gorman, M. (2003) A Design, Implementation and Algorithm Analysis of a Virtual Memory System for Linux, (Unpublished PhD Thesis proposal).

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

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), 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), 152-169.

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

Piccoli G and Ives B (2003) Trust and the Unintended Effects of Behavior Control in Virtual Teams, MIS Quarterly, 27(3), 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), 316-335.

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

United Nations (2004) World Investment Report 2004 - the Shift Towards Services, New York and Geneva, pp. 468p.

Von Hippel, E. and von Krogh, G. (2003) Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science, Organization Science, Vol. 14, No. 2, pp. 209-223.