'Letting go of Control' to Embrace Open Source: Implications for Company and Community

Maha Shaikh
Information Systems and Innovation Group,
London School of Economics and Political Science
m.i.shaikh@lse.ac.uk

Tony Cornford
Information Systems and Innovation Group,
London School of Economics and Political Science
t.cornford@lse.ac.uk

Abstract

It is increasingly understood across the information technology and services sector that engagement with the open source software model can serve as a means for firms to capture intellectual energy, learn about productive software processes, access relevant technical skills, identify and recruit staff, as well as obtain valuable resources including code. This paper reports a study undertaken within two large global IT companies that have been actively involved with open source for more than ten years. The study involved over 30 semi-structured interviews with employees of the companies drawn from top, middle, and lower level management, and included active and experienced developer as well as open source community members. Our findings indicate how these companies have adapted their day-to-day management practices to take into account the need for flexibility and freedom expected by open source communities. This paper focuses on how they 'let go of control' and what the implications of this are for both companies and the communities involved. Our data reveals a number of themes and in this paper we focus on three principal ones; issues of requirements, total cost of adoption, and alignment of open source engagement with long term company strategy.

1. Introduction

This paper builds on Agerfalk and Fitzgerald's discussion of open-sourcing [1]. However, while they discuss open-sourcing as 'outsourcing to an unknown workforce', we consider here large technology service companies that adopt open source code and actively engage with open source communities, often well established and of high repute, and do so in large part because this 'workforce', and its qualities, are very much known. Indeed, it is increasingly understood across the IT sector that engagement with the open source software model can serve as a means for such firms to capture intellectual energy, learn about productive software processes, access relevant technical skills, identify and recruit staff, as well as obtain valuable resources including code. Fitzgerald has coined the term OSS 2.0 to stand for this closer integration of open source activity with corporate interests and business users [2]. The opportunity offered by such engagement is evident for both small niche businesses (SME), and within the largest corporations. This movement to acquire code and associated resources from open source communities has been described under the term 'open-sourcing' [1-3], but also under other terms such as corporate source [4]. There are other terms in use that imply borrowing the ideology of sharing from open source but are not usually explicitly linked to open source such as insourcing [5], cosourcing [6], netsourcing [7], inner source [8] and crowdsourcing [9, 10].

The move towards open-sourcing has led to the emergence of a number of new, and innovative business models. Increased commercial interest in open source has created new opportunities for both companies and open source projects and communities. As Fitzgerald notes, open-sourcing has encouraged studies to evaluate where value is created in software processes, and how it can be sustained as companies engage with these communities [2].

Here we report data drawn from a part of a larger EU funded study looking to open source models for guidance on how to create and sustain digital business ecosystems[1]. We present our initial findings from one part of this work, a field study undertaken within two large global technology services companies who use open source code and work with open source communities. These two companies were chosen because of their experience with this type of relationship - more than ten years in each case. Our goal was to achieve a nuanced and in-depth understanding of what the move from wholly proprietary to substantial use of open source code entails for such technology and business leaders.

[1] **OPAALS** (Open Philosophies for Associative Autopoietic Digital Ecosystems)

IEEE computer society

In this research we have sought insight into the attitudes and strategies of managers in large technology and information services companies as they accommodate this style of open-sourcing within their corporate context and harness and exploit open source activity. This engagement can be challenging for both parties – the corporation and the open source community – as one manager told us, from the perspective of the company, 'it is about letting go of control' in order to keep some control. The analysis presented here focuses on three primary areas that emerge from our data and which are seen to require particular attention by those we interviewed; issues of requirements processes, total cost of adoption, and alignment to long term company strategy.

2. Literature review

The literature on open source adoption by commercial enterprises (large, medium and small) [11-18] covers a range of issues, some of which include governance and the implications of open licenses, business model innovations, and value adding strategies. The increasingly significant scale of the use of open source code within corporate environments, and in particular in the technology and IT services sector was until recently a rather under-researched area [19] though, some focused studies have recently emerged [20-23]. This growing body of work looks at how companies are adopting open source [20], what form this adoption and engagement with open source software and communities takes [22], and the blurring of company and community boundaries through expert exchange [21].

Systems development and management, and software processes in particular, are fundamentally knowledge based activities that in the contemporary environment often need to be undertaken as distributed work and on a substantial scale. As knowledge based activities, one key to success is sourcing talent and gaining access to appropriate and highly skilled knowledge communities. Doing this well can often require innovations that will challenge traditional practices [24, 25]. In other areas of business that depend critically on ideas and talent open innovation models, often linked to the internet, have attracted increasing attention [26-28]. For example, Proctor & Gamble (P&G) in part base their R&D strategy on an open model, named as the "Connect and Develop" innovation model. Huston and Sakkab [29] describe it as a process to "leverage external assets and capabilities…. [in a] relationship of co-invention-based interaction with outside resources". This model with its connections focus, seeks to tap into multiple knowledgeable communities across the globe [29-31]. These authors are clear that this is not a conventional method of 'outsourcing' R&D, but rather "in-sourcing creativity", and as in open source processes, aims to tap into a large pool of people, ideas, developers and testers who can offer the vital diversity needed by a global company such as P&G.

Drawing from another strand in the literature we might understand the use of open source code and engagement with open source communities as a new form of organizing or the building of a novel type of virtual organization appropriate to serve new knowledge needs. Metiu and Kogut [32, 33] through a study of software companies in four different countries identified two distinct forms of organizing for innovation and creativity in globally distributed work. The established model they term the 'global project model', and is based on conventional ideas of specification and control. They then identify a new model emerging –the 'open development model'. The 'global project model' implies that companies are able to take advantage of lower cost of labour by passing some work, essentially routine, specifiable, tasks to offshore low cost sites. This demands requirements specifications up front, and high degrees of control. In their analysis this model begins to transform over time into the 'open development model' as offshore firms and developers begin to not only follow specified requirements but also build their own skills, innovate and promote their own requirements to client companies. This stage is similarly identified by Carmel and Agarwal [34] in Stage 4 of their SITO model of offshore outsourcing.

Kogut and Metiu question to what extent offshore developers (those bound up in conventional offshore outsourcing) are able to move far from simple specification following [33]. The 'open development model' may come to push at the boundaries of the 'global project model', but only when or if the motivation of contributors change, actively seeking new experience, knowledge and skills. This analysis may be a real challenge to conventional offshore outsourcing companies, but it also echoes strongly the general understanding of the important motivations in open source communities; concerned with innovation and creativity, building human capital and with participation driven by the explicit purpose to learn and enhance skills [35, 36]. Metiu and Kogut's analysis may suggest that there are limits to what a company can expect from conventional outsourcing relationships, restricted by specification limitations and by the motivation and capability of people and the offshore developers. However, the open source context we study here is different. Empirically we know that large multinational businesses in the technology sector can and do exploit and even depend on the creative and innovative possibilities opened up by engagement

with open source communities. Further they recognize that this suggests that they can do many positive things to enhance and support their open source partners [37].

Drawing from the literatures outlined above, we then approach this study with a concern for both issues of demand and issues of supply. That is demand for technology products (code etc.) is found within large companies as they struggle to sustain the 'innovation journey' [38] and manage the interorganizational relationships that innovation in this sector almost always implies. Set against this the view of the supply side, leads us to consider the world of open source projects, open source 'workers', and open source cultures and ideologies, as well as the pragmatics of getting an open source project launched and maintaining it over time. In this work we therefore present out findings in terms of the concerns and strategies of corporate managers, as well as the issues that such relationships raise for the community.

3. Methodology

For this study two large global technology companies with substantial markets in systems development and information services were chosen. The larger research project within which this work is taking place also encompasses data collection in small and medium enterprises and directly in open source projects, but the main focus in this paper is on managers in these large companies.

Our access to both companies was agreed upon on the condition of anonymity. Both companies have moved towards greater use of open source software, ideas and development methods over the last ten years and the two companies are in many ways very similar in respect of their engagement with open source. Both have at the core of their business a focus on a mix of software, hardware, managed services and consulting. Their use of open source software is both internal, in their own systems, but also embedded in their software and service offerings sold in the market. The number of discrete open source projects led by Company A are reported by the company's own presentation as over 80, a figure which pales in comparison to Company B's figure of over 3000 - a figure quoted on Company B's website. Company A also acknowledges that it contributes to over a 150 open source projects, while both companies claim to dedicate over a 1000 developers each to open source development.

For the purposes of this work these similarities between the two companies confirms that our interviews are drawing on opinions from a common population. This is not to say that Company A and Company B are similar in all respects, indeed they have substantially different corporate cultures and history as one might expect. However in this paper we do not emphasize any such contrasts between the two, and indeed we have found most of our findings to be generally robust across the two sub-populations showing many common concerns and attitudes.

To give some contrast to the company interviews we also undertook 5 interviews with people who were primarily identified as open source project members – though they may have some paid position but not in our two companies. These people were chosen as people who work in projects that these two companies are also active in. In this aspect we also drew upon work in the wider study which has interviewed a number of open source developers.

Table 1: Interviews

Interviewee Details	Number of Interviews
Company A	
• Directors	2
• Managers - Technology - Marketing - Strategy	11
• Developers	5
Company B	
• Directors	3
• Managers - Technology - Marketing - Strategy	7
• Developers	3
Community	5
	36

3.1. Data Collection

Our research method was primarily based on semi-structured interviews, some carried out in person, but mostly via telephone. Interviewees were based in Europe and the USA. Each interview lasted an hour or more. The interviewees belonged to top and middle management, and included active software developers with a mix of responsibilities in corporate and open source development (see Table 1). In Company A access was negotiated in a

traditional manner by approaching a senior manager and asking the contact to suggest key personnel who could prove fruitful interviewees, keeping in mind the focus of our study. For Company B we did not have a similar senior contact, and our access resulted from a search of the Internet for names of personnel in open source related positions in large technology companies. We used project websites that large companies were linked too and searched mailing lists for possible interviewee names. We found our first interviewees at Company B in this manner. Subsequently we successfully adopted the snowballing method whereby each interviewee was asked to offer a few more potential contacts. Most unhesitatingly offered two or more names, people in their own company or colleagues in similar large technology based firms.

3.2. Data Analysis

Interviews were transcribed and coded using *Atlas.ti* content analysis software. Using the tools of Grounded Theory [39, 40], though not the full ontology, yielded a code book of forty-nine initial codes (reference withheld for anonymous reviewing purposes). Along with codes we added memos in the form of notes, concepts and broader emerging themes.

Table 2: Open Codes

Open Codes
After sales services
Best practices
Coping strategies
Governance model
Hybrid workers
Incentive schemes
Innovation
Liaison with community
License
Resource capture
Requirements
Value creation

Our theoretical inclinations provided us with a lens through which to understand our data and offered the first 11 codes (see Table 2) in the open coding step. However, our theoretical ideas need to be understood more as meta guidance to our analysis rather than offering concepts for micro analysis. The three main themes that our coding and memos gave rise to and which we explore here include: issues of requirements processes; total cost of adoption; and alignment with long-term company strategy.

4. Findings and discussion

4.1. Requirements processes

We found, perhaps not very surprisingly, that companies and open source communities had a somewhat different understanding of requirements process both in the traditional form (expressions of desired functionality), and as implicit in the flow of submitted code patches. Employees from both companies clearly stated how little they tried to exercise any pressure on the open source community they worked with to accept their own specific requirements for the product. As a manager from Company B put it, “You can’t bully and you certain don’t even want to try to bully an open source project”. A peer in Company A was a bit less emphatic and spoke about a “balancing act” because “It’s about the world sharing your vision and explaining why you think it is better. […] Hopefully, the reasons are so good and then others are going to think the same”.

On the other hand, community members that we interviewed expressed frustration at times with company employees. Community members felt that they were at times coerced to take a product in a specific direction that suited the needs of a company or that company’s customers. Coping strategies employed by community developers (Table 3) ranged from simple disregard of such directives to more active strategies like enhanced scrutinizing of the patch submission process and establishing company and community interaction avenues to ensure stronger peer review.

Many, if not most, company coping strategies involve placing their own employees in the midst of the community so as to become ‘one of them’. This will entail making more various and substantial contributions to the community and the product, but it could also support greater influence and control by a company (see also discussion below of total cost of adoption). Such a logic of reward for efforts is indeed in keeping with general open source practices where the more one contributes, and does so with impact, delivering quality code and showing sound judgment, the more rights and authority any developer gains.

Companies also foster their chosen open source communities by offering other incentives such as hosting services or sponsorship of meetings. The concern expressed by community members was that the community, once accustomed to company support in the form of sponsorship, hosting or free hardware might find it awkward to deny company employees

some extra margin of influence over the direction a project takes (see also the discussion below of strategy and alignment).

Indeed, in the context studied here both companies are eminently able in theory to take over any such code if need be. The larger concern for them is the

Table 3: Requirements processes

Concern around	Issue for Managers	Company Coping Strategies	Issue for Community	Community Coping Strategies
Requirements Processes	Losing control over day to day activities and development	Funneling resources to company developers to participate in community work to take active part in community discussions, offer bug-fixes etc	How to counter bullying tactics of big companies?	Ensuring that the community keeps a rigorous review process for patch submission
	Product requirement needs are distinctive from those of the developer community	Building in-house team to take on work not covered by the community	Maintaining a more emergent form of requirements gathering without losing company support	Allowing the emergence of a trajectory of software that keeps more than one version of the software active
	Tricky to match company long-term strategy with an emergent form of product development practiced by the community	Participate in (or encourage the establishment of) a steering committee to influence project development	Need to counter the more rigid development practices assumed/imposed by some companies	Making sure that the community is adequately represented on the steering committee
	Threat of the community losing interest in the product which is crucial to the company	Creating in-house expertise to counter the threat of the community losing critical mass of developers	Losing critical mass of developers because of company interference	Maintaining more than one version of the software, where there is always a beta or experimental version available for hackers to contribute to

4.2. Total cost of adoption

Lower total cost of ownership (TCO) is often explored in research studies as a key factor in company adoption of open source products (or not) [41]. It is recognized that TCO is not necessarily lower for open source, and in any case is not easy to measure [42, 43]. Given the context of this study the conventional concept of TCO as related to an end user product such as a desk-top suite is not directly relevant (these are not just 'users' of the code, but participants in an open source process), but still our interviewees used the term to convey the question of the relative costs of writing and supporting some code in house, or taking it from open source. While our respondents spoke of TCO we have chosen to slightly rephrase this as Total Cost of Adoption (TCA), to reflect their specific context and concerns.

Table 4 provides some indication of the factors that respondents considered need to be taken into account when trying to understand TCA for open source code. The factors 'look' familiar to the TCO debate but, for example, the implication of lock-in is different for an open source product where source is by definition available and there is potentially the ability to find alternative maintainers/suppliers. dependence on the expertise and knowledge held within the community which created the code. This may be harder or more expensive to replicate, and as open source development is usually thought not to document code as rigorously as closed source software companies, it may make sustaining a code base problematic.

For the community itself TCA is not a direct concern, though they do understand that there is a negative impact on the viability of any project if companies do not adopt or if in time they withdraw their interest and funding. For example, while open source projects often struggle to sustain documentation, and test schedules, these may be important components from the view of an adopting company and may be exactly the kind of activity which an adopting company can and will resource with its own staff, as we were told.

One feature identified by our respondents, which has not been focused on in the literature, was the concept of a 'healthy community' established around an open source product. We probed managers further about this idea and found it to be a major deciding factor when in a situation of choosing one open source product over another. As an interviewee told us, "One of the decision criteria that a development team needs to [use] when they are choosing open source software, [is] how viable and vital is the

community that they are drawing from […] product teams and users in general will try to stay with projects that are very robust and have a long term future and that's why, you know, (Company x) invests in the various communities that we participate in."

Another manager was happy to define the characteristics considered when assessing a healthy community; "It's how many people are in the community, how many people contribute, how often do they release, how many bugs do they have on a given release. How many days does it typically take for a bug to be resolved [by] the community, how has it grown or shrunk overtime". A link to the company strategy is also important, "Is the direction of the community […] in line with the direction that the given [in-house] product team need over the long term. […] Those are the things that you evaluate to begin with, and then monitor overtime to understand the viability of the project".

In interviews we asked managers if TCA of open source was in their experience less for open source projects. They explained that though in the short-run the cost of open source development might be slightly higher than proprietary in the long–run it will (or should) fall. One manager stated it quite clearly, "At first, yes, you have to hire in a lot of people and you have to build that link to the community, so they will have to spend a lot of time being active in that community, so they won't work maybe 100% of their time on your software and they will have to spend 20-30% of their time just reading the forums and answering and proving that they know the software and having their name known by the major developers of that software. It will cost time and money, but after that, [TCA] will drop, drastically".

Companies reported that they found various training and retraining costs more straightforward to evaluate. Gradually, in both companies, we found that they considered TCA less problematic (better understood) The companies we were told had

Table 4: Total Cost of Adoption

Concern around	Issue for Managers	Company Coping Strategies	Issue for Community	Community Coping Strategies
Total Cost of Adoption	Difficult to estimate the cost of creating, and sustaining a community around a project	Counter the issue of cost estimation of community and product sustainability through the use of past maintenance contract experience.	Communities need to be aware of license differences, possibility of a fork, and somehow manage to hold the community around a shared goal.	Using company interest to boost their profile and project attractiveness to keep developer interest.
	Switching costs to OS can be high in terms of staff commitment	Working around incompatibility issues by choosing OS projects that are a closer match with legacy system	How to make their open source product attractive to companies without being taken over	Creating more user friendly code, better, and more documentation, engineering compatibility features.
	Training in the use of OS code can be significant – e.g. new skills and new practices to be learned.	Re-training expenses can be modeled on usual in-house training outlays	Balancing the status and 'promotion' of maintainers etc if some people are paid to do such work – and sustaining the position requires constant work	Developing wider understanding with company partners.
	Risk management is an issue (uncertainties to cope with like license differences, security, lack of support, and documentation).	Using risk analysis methods that are better suited for fast changing environments, and choosing the OS license wisely (if there is a choice)	How to improve the level of documentation to sustain company interest	Create foundations that form a bridge between commercial companies and the community, and provide some of these services
	Scalability and modularity of the product to allow integration into own services and products	Assessing OS in terms of modular, reusable and maintainable code	Duplication or distraction of effort	Adopting, for some areas, a more company centric approach of delegating work.
	Locked in to a community that may not survive, or may fork (another form of vendor lock-in?)	Holding onto the source (as it is OS) and building an in-house team	Overly dependent on financial and marketing support of the company	View companies as potential trainers and future employment
	Integration can become an issue as code evolves	Use of open standards	May be conflicting choice of standards	Draw upon wider open source traditions

managed to convert most of these costs if not into figures, into something understandable to top management and were able to formulate reasonably accurate and credible estimations of TCA.

4.3. Strategy and alignment

Managers across both companies were clear that a company wide open source strategy was essential so as to present a coherent and consistent face to the open source communities, and equally so internally. For example, a consistent set of understandings on the use of various open source licenses was needed and an appropriate review process as elements of proprietary code are released as open source. When we asked about the nature of such a strategy we received two distinct accounts, depending perhaps more on the respondent's views than on the particular company. By one senior manager we were told about the strategy in terms of sales; "The basic strategy is simple and that's to sell hardware, software and services. Open source is an enabler for that. Open source allows us to get products out there to meet market demand." We were also told that "It's quicker to develop with them. It opens up a lot of service opportunities for people who are using it or want to use open source software. And, virtually, everybody does use open source software."

Another equally senior manager, when asked directly what was the strategy for open source use and adoption, replied more in terms of the desire to tap into and sustain a process of innovation. "At the centre of it, right, is innovation, community innovation. We think that community innovation is a good thing. We think that it's one of these things like with the Internet that, if you can get the tide to rise … it's a good thing." This metaphor of the rising tide, where all the boats float free, indicates that open source is seen here in terms of the wider industry and its customers. "We want to accelerate community innovation, because it's going to help the whole process move forward. We want to also harness that innovation or harness the invention, so it becomes innovation." From this desire to access innovation and inventions comes a further element of the strategy; "The second thing is that, we think, if you are going to […] derive business from it, you need to be a contributor. You need to be active in the communities, certainly as a computer company".

Significantly, these two companies have consultancy and outsourcing relationships with their own customers, and at times need to offer them advice on the use of open source. As one manager from Company A pointed out, "what we tell others is that you basically need an open source strategy and you need to have some kind of process in place […] some companies are willing to take more risks and that might influence [their] policies. But, basically, [they] need some kind of strategy that open source can be used." The interviewee followed up with another message to such companies, "You simply can't live without open source, I mean. It's just unavoidable".

Table 5: Strategy and Alignment

Concern around	Issue for Managers	Company Coping Strategies	Issue for Community	Community Coping Strategies
Strategy and Alignment	Constant tussle with differing aims and goals of the company from the community	Attempts to persuade the community through greater interaction, mailing list communication and bug fixes	Constant tussle with differing aims and goals of the community from the company	Giving privileges and positions of control and maintainership to company developers based on trust earned over time
	Customer satisfaction problems with OS	Mediating between the OS community and the customer with their own line of support, but using the community mailing forums too	More users or customers usually leads to more developer interest and thus a more sustainable community and product	Creating more user friendly products to increase the user base; and aiming for less aggressive manner of replying to forum queries
	Countering customer mistrust/lack of knowledge of OS	Using an OS license that is less restrictive; showing cost advantages to the customer; promising higher levels of support	More company interest implies greater levels of sponsorship	Creating a product that provides an OS alternative to a proprietary one
	Seen as parasitic on the community and thus in danger of losing their support	Indulging in more sponsorship schemes for the community; support through free hardware provision; holding joint conferences to encourage greater communication.	To find ways to ensure that the company contributes back to the community somehow	Using the company as a platform for possible jobs; controlling the level of participation allowed to company members

In both establishing and operationalizing a strategy a main concern was finding ways to keep customers happy and sell more services and products while holding onto and being receptive to the innovation, ideas and expertise coming from the open source communities. While the companies, as seen above, were fairly clear on their open source strategy, they found a number of challenges as they attempted their implementation (Table 5).

A core element of this strategy is to manage the relationship with chosen open source projects in a way that serves both short term and long term interests of the company. In doing this the companies face a willing, if at times hesitant, partner in open source projects. Open source community developers explained that they were eager to maintain company interest in their work and their product. Company interest meant that they would be able to attract more developers to their project, and developers willing to undertake some of the less attractive work. Gaining this critical mass of developers for an open source project is often a serious dilemma so company interest is welcomed, and a major company can be seen as an important fellow collaborator. Existing members may also believe that there might be, for example, job opportunities from greater exposure and experience. Indeed, some open source communities approach and invite company interest in their work, a significant change from the outright suspicion of parasitic behaviour that might have been common some years ago. This is an example of the shift that Fitzgerald identifies with his concept of OSS 2.0.

5. Conclusion

Both company and community perspectives are distinct, yet we have seen in our data compelling evidence to suggest that both can live side-by side in a symbiotic relationship. The balance of power is delicate and needs constant maintenance and management in order to survive. Large companies understand that if they want to preserve a long term relationship with open source communities and harness the expertise and products they offer, then they must loosen up and relax, avoiding to much concern about their level of control. Companies may not 'lose' control, but they do need to let go of it. This demands a different mindset The long-term implications of such collaboration unfold over time, and we see a greater hybridization apparent in communities, open source products, and development process as business learns to work with open source.

Whether we speak of losing or letting go of control what is notable in the two companies studied here is that they are also contributing to each others' work through open source projects, still with the objective of self-gain. Indeed, one interviewee described open source as just one option to consider in any context where a collaborative effort was needed, to be judged against such alternatives as a formal joint venture, a standards making body, or a sub-contracted bit of work. These companies, in the end, engage in open source not for the greater good, but to increase their profit margin. The open source communities, though perhaps never quite as altruistic as painted by Raymond [44, 45] are having to respond and showing signs of increased awareness of the significance of the commercial facets of their work.

6. References

[1] P. Agerfalk and B. Fitzgerald, "Outsourcing to an Unknown Workforce: Exploring Opensourcing as a Global Sourcing Strategy," *MIS Quarterly,* vol. 32, pp. 385-400, 2008.

[2] B. Fitzgerald, "The Transformation of Open Source Software," *MIS Quarterly,* vol. 30, pp. 587-598, September, 2006.

[3] M. Shaikh and T. Cornford, "Open-Sourcing: On the Road to the Ultimate Global Source?," in *2nd Global Sourcing, Services, Knowledge and Innovation Workshop* Val d'Isere, France, 2008.

[4] J. Dinkelacker, P. Garg, R. Miller, and D. Nelson, "Progressive Open Source," in *Proceedings of the 2002 ACM International Conference on Software Engineering (ICSE'02)*, 2002, pp. 177-184.

[5] R. Hirschheim and M. C. Lacity, "The Myths and Realities of Information Technology Insourcing," *Communications of the ACM,* vol. 43, pp. 99-107, February, 2000.

[6] K. M. Kaiser and S. Hawk, "Evolution of Offshore Software Development: From Outsourcing to Cosourcing," *MIS Quarterly Executive,* vol. 3, pp. 69-81, June 2004.

[7] T. Kern, L. Willcocks, and M. C. Lacity, *Netsourcing: Renting Your Business Applications and Services over a Network*: Financial Times/Prentice Hall, 2002.

[8] G. Gaughan, B. Fitzgerald, L. Morgan, and M. Shaikh, "An Examination of the Use of Inner Source in Multinational Corporations," in *1st OPAALS Workshop*, Rome, Italy, 2007.

[9] D. Brabham, "Crowdsourcing as a Model for Problem Solving: An Introduction and Cases," *Convergence: The International Journal of Research into New Media Technologies,* vol. 14, pp. 75-90, July 7th 2008.

[10] D. Brabham, "Moving the crowd at iStockphoto: The composition of the crowd and motivations for participation in a crowdsourcing application," *FirstMonday,* vol. 13, 2nd June 2008.

[11] S. Butler, D. Adebanjo, and H. Ismail, "Open source software and leveraging of business effectiveness in SMEs - A case study," *Ice-B 2008: Proceedings of the International Conference on E-Business,* pp. 93-100, 2008.

[12] E. Capra and A. I. Wasserman, "A framework for evaluating managerial styles in open source projects," *Open Source Development, Communities and Quality,* vol. 275, pp. 1-14, 2008.

[13] M. Pulkkinen, O. Mazhelis, P. Marttiin, and J. Meriluoto, "Support for knowledge and innovations in software development - Community within company: Inner source environment," *WEBIST 2007: Proceedings of the Third International Conference on Web Information Systems and Technologies, Vol SeBeG/eL,* pp. 141-150, 2007.

[14] K. Martin and B. Hoffman, "An open source approach to developing software in a small organization," *Ieee Software,* vol. 24, pp. 46-+, 2007.

[15] J. Holck, M. H. Larsen, and M. K. Pedersen, "Managerial and technical barriers to the adoption of open source software," *Cots-Based Software Systems, Proceedings,* vol. 3412, pp. 289-300, 2005.

[16] J. Lindman, M. Rossi, and P. Marttiin, "Applying open source development practices inside a company," *Open Source Development, Communities and Quality,* vol. 275, pp. 381-387, 2008.

[17] C. Melian and M. Mahring, "Lost and gained in translation: Adoption of open source software development at Hewlett-Packard," *Open Source Development, Communities and Quality,* vol. 275, pp. 93-104, 2008.

[18] K. Ven and H. Mannaert, "Challenges and strategies in the use of Open Source Software by Independent Software Vendors," *Information and Software Technology,* vol. 50, pp. 991-1002, 2008.

[19] Economist, "Born free: Open-source software firms are flourishing, but are also becoming less distinctive," in *The Economist*. vol. May 28th, 2009.

[20] L. Dahlander, "Penguin in a newsuit: a tale of how de novo entrants emerged to harness free and open source software communities," *Industrial and Corporate Change* vol. 16, pp. 913-943, 2007.

[21] L. Dahlander and M. W. Wallin, "A Man on the Inside: Unlocking Communities as Complementary Assets," *Research Policy,* vol. 35, pp. 1243-1259, 2006.

[22] L. Dahlander and M. G. Magnusson, "Relationships between open source software companies and communities: observations from nordic firms," *Research Policy,* vol. 34, pp. 481-493, 2005.

[23] W. Stam, "When does community participation enhance the performance of open source software companies?," *Research Policy,* vol. 38, pp. 1288-1299, 2009.

[24] H. W. Chesbrough, *Open Business Models* Boston, MA: Harvard Business School 2006.

[25] H. W. Chesbrough, W. Vanhaverbeke, and J. West, "Open Innovation: Researching a New Paradigm " London: Oxford University Press 2007.

[26] E. von Hippel, *Democratizing Innovation*. Cambridge, MA: MIT Press, 2005.

[27] E. von Hippel, "Innovation by user communities: Learning from open-source software," *MIT Sloan Management Review,* vol. 42, pp. 82-86, Sum 2001.

[28] E. von Hippel and G. v. Krogh, "Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science," *Organization Science,* vol. 14, pp. 209-223, March-April 2003.

[29] L. Huston and N. Sakkab, "Connect and Develop: Inside Procter & Gamble's New Model for Innovation," *Harvard Business Review,* vol. 84, pp. 58-66, March, 2006.

[30] N. Sakkab, "Connect & Develop Complements Research & Develop at P&G," *Research - Technology Management,* vol. 45, pp. 38-45, 1 March 2002.

[31] L. Huston and N. Sakkab, "Implementing Open Innovation," *Research - Technology Management,* vol. 50, pp. 21-25, March-April 2007.

[32] A. Metiu and B. Kogut, "Distributed Knowledge and the Global Organization of Software Development," Wharton School, University of Pennsylvania, Philadelphia February, 2001.

[33] B. Kogut and A. Metiu, "Open-source software development and distributed innovation," *Oxford Review of Economic Policy,* vol. 17, pp. 248-264, Sum 2001.

[34] E. Carmel and R. Agarwal, "The Maturation of Offshore Sourcing of Information Technology Work," *MIS Quarterly Executive,* vol. 1, pp. 65-77, June 2002.

[35] M. Shaikh, "Version Control Software in the Open Source Process: A Performative View of Learning and Organizing in the Linux Collectif, Doctoral Thesis in Information Systems, London School of Economics and Political Science," in *Unpublished Doctoral Thesis in Information Systems* London: London School of Economics and Political Science, University of London, 2007.

[36] K. R. Lakhani and R. G. Wolf, "Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects," in *Perspectives on Free and Open Source Software*, J. Feller, B. Fitzgerald, S. Hissam, and K. R. Lakhani, Eds.: MIT Press, 2005.

[37] R. T. Watson, D. Wynn, and M.-C. Boudreau, "JBOSS: The Evolution of Professional Open Source Software," *MIS Quarterly Executive,* vol. 4, pp. 325-341, 2005.

[38] van de Ven, D. E. Polley, R. Garud, and S. Venkataraman, *The Innovation Journey*: Oxford, 2008.

[39] Strauss and J. Corbin, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*: Sage Publications, 1999.

[40] B. G. Glaser and A. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Chicago: Aldine, 1967.

[41] J. Stoller, "Open Source: assessing the TCO picture," *CMA Management,* vol. 78, pp. p18-22, 2004.

[42] A. MacCormack, "Evaluating Total Cost of Ownership for Software Platforms: Comparing Apples, Oranges and Cucumbers.," mimeo 2003.

[43] R. M. Sauer, "Why develop open-source software? The role of non-pecuniary benefits, monetary rewards, and open-source licence type," *Oxford Review of Economic Policy* vol. 23, pp. 605-619, 2007.

[44] E. S. Raymond, *The Cathedral & the Bazaar*, 2 ed. Sebastapol, CA: O'Reilly, 2001.

[45] E. Raymond, "A Brief History of Hackerdom," in *Open Sources: Voices from the Open Source Revolution*, C. DiBona, S. Ockman, and M. Stone, Eds. California: O'Reilly, 1999.