Improving Packaged Software through Online Community Knowledge

Helena Holmström

University of Limerick, Ireland
helena.holmstrom@ul.ie

Ola Henfridsson

Viktoria Institute and Halmstad University, Sweden
ola.henfridsson@viktoria.se

Abstract. Packaged software development (PSD) is largely a knowledge-intense activity. Thus, it depends on the organizational capability of developing and combining market and technical knowledge into timely and competitive software products. Given customers' situated knowledge of the software, software firms increasingly seek new ways to involve customers in their software development activities. As highlighted in the literature, one path for doing this is to use online communities. However, there exists little empirical research that examines the role that communities can play in the commercial endeavor of PSD. To address this omission, this paper examines the benefits and limits of online community use in PSD as it unfolds at the intersection between commercial software firm practices and voluntary community participation. On the basis of this examination, the paper presents implications for both research and practice.

Key words: packaged software, online communities, community participation

1 Introduction

Packaged software is an increasingly important form of information technology. Already in 1998, packaged software was the fifth largest industry in the US (Sawyer 2000), and it is now widely used by both business organizations and consumers. Sold by vendors, distributors or stores, packaged software (also known as shrink-wrapped, commercial off-the-shelf, or commercial software) can be distinguished as tradable software products that are designed to be easily installed and to interoperate with existing system components (Abts 2002).

Developing packaged software is largely a knowledge-intense activity (Clegg et al. 1996), driven by time-to-market demands in that breaking new ground is conceived critical for competitive advantage (Zachary 1994). Such development is conducted by developers that typically hold line positions making them central to firm performance (Sawyer 2000). Rather than holding supportive staff functions, packaged software developers are more center-stage than most IT staff in that they often possess the core competence central to the software firm's competitiveness (cf. Hamel and Prahalad 1994).

Given the centrality of packaged software development (PSD), software firms continuously seek new ways to improve their development processes (Humphrey 1989; Mathiassen et al. 2002). Recently, customer involvement has been recognized as a means to leverage PSD (see e.g., Keil and Carmel 1995; Sawyer 2000). While its equivalent in custom IS development—user involvement—is a well established ingredient in both literature and practice (see e.g., Carmel and Sawyer 1998; Greenbaum and Kyng 1991), however, customer involvement in PSD is yet to gain momentum. Such involvement is fairly uncommon and often based upon indirect links such as intermediaries or customer surrogates (Keil and Carmel 1995).

The relative rareness of customer involvement in PSD can be associated with at least two customer involvement barriers. First, PSD is characterized by geographically distant customers (Sawyer 2000), making face-to-face interaction difficult to achieve. Second, packaged software customers are unknown (Grudin 1991) in that they are not part of any coherent use context. The development context of packaged software is clearly separated from the use context, i.e., there are no consistent organizational processes or structures at hand when representing customers. Instead, there are multiple, sometimes conflicting, individual needs and requirements to take into consideration when developing packaged software for a mass market.

In view of these barriers, software firms increasingly develop and deploy online communities for aligning customers with their PSD processes. In such

communities, customers can engage in beta testing of nearly finished software as well as in more comprehensive activities including customer-driven software development and maintenance (Holmström 2001). In this regard, online communities can facilitate the engagement of competent but distributed customers in developing, testing, and modifying software (Lee and Cole 2003). Such communities can also support the development of a better comprehension of customer perspectives on the software. In other words, online customer communities can be understood as a means of overcoming geographical distance and the lack of a coherent use context characterizing packaged software customers.

As documented in the literature on distributed product development, development efforts involving loosely coupled individuals and groups rely on a capability to convert the collective knowledge possessed into appropriate action for improving the product (Nambisan 2002; Orlikowski 2002). For example, in successful open source communities, this capability is a natural and important element in successful development of software (Lee and Cole 2003; Ljungberg 2000). In PSD, however, firms face a number of challenges as they attempt to benefit from customer communities. The main problem is the inherent tension between the motivational structures of commercial software firms and those of voluntary community participation. Indeed, our previous case study research of a computer game firm documents attempts to benefit from customer communities in developing packaged software (Holmström, 2001; Henfridsson and Holmström, 2002). However, our research also highlights sensemaking and customer role difficulties with community use.

On the basis of a longitudinal case study at Daydream Software, this paper explores the role of online communities in PSD. In particular, we focus on the inherent tension that unfolds at the intersection between commercial software firm practices and voluntary community participation. To this end, we adapt Newman and Robey's (1992) social process model of user-analyst relationships to the PSD context. We then use this model for better understanding the prospects and limits of developer-customer relationships in community-enabled PSD.

2 Customer Community Knowledge in PSD

Developing packaged software is largely a knowledge-intense activity (Clegg et al. 1996). Thus, it depends on the organizational capability of developing and combining market and technical knowledge into timely and competitive products (Andreu and Ciborra 1996; Prahalad and Hamel 1990). Indeed, the

packaged software industry is dominated by time pressures (Sawyer 2000) in that breaking new ground is often conceived critical for generating return on investment (Zachary 1994). Contrary to custom IS development, the success of the packaged software industry's products is measured by profit and market share, underscoring the challenge of either developing a large installed base or creating new market opportunities (Sawyer 2000).

In quest for competitive advantage, software firms continuously seek new ways for improving their PSD processes (Mathiassen et al 2002). Apart from efficiency measures applied to increase staff effectiveness and lower rework time (Little 2004), PSD can benefit from leveraging customer involvement and input (Keil and Carmel 1995). Indeed, many of the best ideas for product improvements come from customers (Finch 1999; Von Hippel 1986). In possessing situated knowledge of the software, customers constitute an important resource for improving PSD in that they can co-produce the software. Nambisan (2002) portrays this as 'knowledge co-creation', highlighting the centrality of knowledge produced in developer-customer relationships for gaining competitive advantage over time (see also e.g., Keil and Carmel 1995).

Customer knowledge about software is characterized by its situational nature. As recognized by Brown and Duguid (2001), what individuals know reflects the social context in which knowledge is acquired and applied. Within the scope of this paper, customer knowledge emerges from everyday software use. Hence, it reflects not only the particular circumstances and different purposes of software use but it also conveys necessary technical competence such as hardware and software configuration skills. As such, this type of knowledge is enacted in the moment (Orlikowski 2002), reflecting the capability of individuals or groups to transform their situated software knowledge into meaningful action. Such transformation is achieved through continuous reconstitution (Orlikowski 2002) and renegotiation (Wenger 1998) of meaning. Thus, customer knowledge is something that is achieved, not given, and it emerges from people's ongoing reflection, experimentation, and improvisation within the (software) practice of which they are part (Orlikowski 2002).

Online communities can be considered as important mediators of software customer practices (Lee and Cole 2003). Whether online or not, communities work as repositories for the development, maintenance, and reproduction of knowledge (Brown and Duguid 2001). In this vein, we refer to community knowledge as the situated software knowledge (e.g., design, graphics, and hardware/software configuration skills) manifested in the practice of a group of distributed software customers. While a community's knowledge is not held equally by all members but shared across the community, participation gives access to the collective knowledge generated within the community (Brown and Duguid 2001).

In order to integrate community knowledge in the PSD process, however, software firms need to initiate actions intended to use community knowledge. Throughout this paper, we refer to such actions as community interventions. First, knowledge building denotes interventions made by the software firm intended to support software customers' creation and sharing of situated software knowledge. Such interventions can consist of setting up websites and active participation in the forums of such sites. In this context, however, it might be noted that creation and sharing of situated software knowledge is relatively independent of what a software firm does to promote it. As long as the boundary object—the software—attracts enough committed customers there typically will be customers internalizing situated knowledge from software use, which a significant portion of them will share with others (Ljungberg 2000). In other words, the willingness and capacity of creating and sharing situated software knowledge is something that resides naturally within a popular online community.

Second, improving PSD cannot merely come from knowledge building. It also depends on a capability of the software firm (1) to stimulate customers' willingness to express their knowledge, and (2) to incorporate the knowledge expressed in the PSD process. This knowledge elicitation capability can be referred to as interventions made by the software firm for making sense of customer-generated input. In this regard, the software firm needs to develop organizational arrangements with which they are able to transform customer input into meaningful improvements of the software (Henfridsson and Holmström 2002). These arrangements involve a sensemaking capability (Weick 1979) attentive to the knowledge built in light of perceived business needs.

Finally, knowledge exploitation refers to the firm's implementation of software improvements on the basis of elicited customer knowledge. As in any product development process, this process is influenced not only by organizational factors such as firm culture, but also environmental factors including risk capital availability and market forecasts. For example, the individualistic and entrepreneurially-oriented cultural milieu of packaged software development can pose a challenge for implementing product suggestions generated by customers (Carmel 1997; Sawyer 2000).

3 The PSD Challenge: Business Meets Volutarism

There are different ways to access customer knowledge in PSD. Traditionally, customer knowledge is collected through indirect links such as intermediaries

or customer surrogates (Keil and Carmel 1995). Popular developer-customer links noted by Keil and Carmel include support lines, marketing and sales, trade shows, and focus groups. In this regard, Microsoft has invested considerable energy in making customer support be part of its software improvement (Cusumano and Shelby 1995). Indeed, the world-leading software firm has developed a whole set of customer input types covering the product development cycle from requirements gathering to product refinement.

One central finding in Keil and Carmel's (1995) study is the importance of reducing the proportion of indirect developer-customer links in favor of more direct links. While most of Microsoft's links documented in the study by Cusumano and Shelby are indirect links, online communities are a type of direct developer-customer link with potential value for software firms. One important source of inspiration for software firms that consider community-based customer links is the tremendous success for open source communities (see e.g., Feller and Fitzgerald 2001; Lee and Cole 2003; Ljungberg 2000). The prospect of involving thousands of committed and competent volunteers in the PSD process is tempting. Given that online communities were pioneered as virtual places in which people meet to socialize, exchange experiences, and enjoy the possibility to establish relationships without any exposure to commercial interests (see e.g., Markham 1998; Rheingold 1994); a key challenge in such efforts is the tension between the organizing principles of commercial software firms and those of voluntary community participation.

Indeed, there exist significant differences between firm-based and community-based models of knowledge creation. These differences are manifested in organizing principles such as intellectual property ownership, membership restriction, authority and incentives, knowledge distribution, and mode of communications (Lee and Cole 2003). An important source of these differences is the locus of action. For instance, the firm-based model of knowledge creation assumes the firm to be the boundary of efforts to stimulate learning. This assumption plays out as a firm-centered knowledge ownership view that fosters a culture in which knowledge distribution is restricted outside firm boundaries.

Given the gift-culture and its associated motivational structures that drives open source communities (Bergquist and Ljungberg 2001), attempts to benefit commercially from community use are associated with paradoxes. As idealized models of knowledge creation (see Lee and Cole 2003), these paradoxes seem virtually irresolvable. In this regard, we take heart from recent research suggesting that so-called hybrid models can be a plausible way forward for commercial software firms to foster community practices similar to those found in open source (Sharma et al. 2002). Although we take this suggestion as an inspiration for our research, however, we find Sharma et al.'s view

unsatisfying in that they do not explore the possibility of such practices in an online community context. The next section outlines a social process model that function as our basis for such an exploration.

4 Towards a Process Model of Community Use in PSD

In using online community knowledge, software firms seek to stimulate collective idea generation and product conceptualization among geographically distant customers (Nambisan 2002). Given the tension between the organizing principles of commercial software firms and those of voluntary community participation, such use can be seen as a process marked by varying degrees of customer acceptance of the community interventions (knowledge building, elicitation, and exploitation) made by the software firm. Sustained customer acceptance of community interventions over time can be assumed to be essential in successful community use in PSD. In investigating the benefits and limits of community use in PSD, we therefore need to explore relationships between software developer intervention and customer acceptance of such intervention over time. In doing this, we adapt Newman and Robey's (1992) social process model of user-analyst relationships as a lens through which to understand how such relationships play out in the process of community use in PSD.

Encouraged by Newman and Robey's (1992, p. 252) remark that the model's original organizational role differentiation should not be limited to traditional systems development, we differentiate between software developers and software customers (rather than analysts and users) for our purposes. Thus, the social process model of developer-customer relationships specifies that established relationships between developers and customers persist over time as long as critical encounters do not change the current trajectory of a project (Newman and Robey 1992). Viewing software development as a sequence of events consisting of antecedent conditions, encounters, episodes, and outcomes; the model is useful for studying not only the relationship between preceding events and their consequences but also for analyzing possible directions of changing existing developer-customer relationships.

Antecedent conditions are simply the relationship between developers and customers that existed before the community under consideration was established. These conditions are usually governed by relationships that emerged in prior endeavors. In cases in which a software firm initiates a community for supporting their PSD process for the first time, it can be assumed that both the

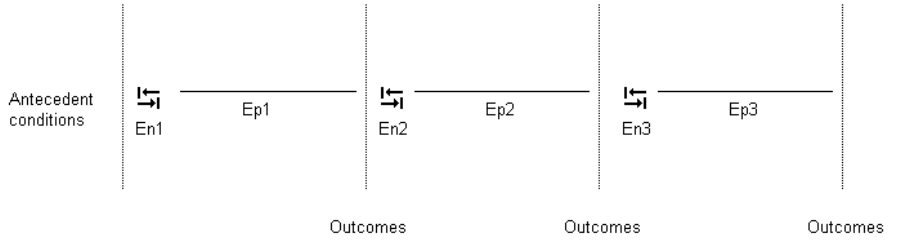

Figure 1. A process model of community use in PSD

software firm and the customers participating in the community will appropriate practices from related contexts. Software developers are likely to relate to developer-customer link types that they have previously encountered in their professional career. Customers are likely to relate to their experience of participating in other online communities.

Drawing on punctuated equilibrium theory (Gersick 1991), the model depicts a process characterized by periods of stable relationships punctuated by short periods of change. In this regard, the sequence of events that follows from initiating a project (in our case, online community) is divided into encounters and episodes. An episode conceptualizes the beginning and end of a set of events that are related to each other in that they stand apart from other events. Encounters between developers and customers mark breaks in episodes of incremental and fairly stable relationships. These encounters can be described as 'windows of opportunities' that are critical to the project's trajectory (Tyre and Orlikowski 1994).

In view of Newman and Robey's model, community use in PSD can be analyzed as a social process characterized by episodes and encounters in the relationship between developers and customers (see Figure 1 and Table 1). These episodes include the three different community interventions outlined earlier (knowledge building, elicitation, and exploitation). Given the tension between a software firm's commercial interest and customers' voluntary community participation, the software firm's challenge is to maintain customer acceptance of its attempts to use the knowledge developed, maintained, and reproduced in communities (cf. Brown and Duguid 2001). Successful handling of this relationship on behalf of the software firm can be hypothesized to be central to the possibility of community use in PSD.

Given that our own work (Henfridsson and Holmström 2002; Holmström 2001) and that of others (Keil and Carmel 1995; Sawyer 2000) recognize the promise of customer involvement in PSD, we view our adaptation of Newman and Robey's (1992) social process model as a way of focusing on the different

episodes and encounters of community use in PSD. In the next section, the research methodology and the empirical context for our study are outlined.

Concepts	*Definitions in the PSD context*
Antecedent conditions	The relationship between developers and customers that existed before the community under consideration was established.
Encounters (En)	Short periods of changed relationships between developers and customers. The resulting relationship sets the agenda for the coming episode of community interventions.
Episodes (Ep)	Characterized by a period of a stable and specific set of community interventions intended to support: • Knowledge building: The process of supporting software customers' creation and sharing of situated product knowledge. • Knowledge elicitation: The process of making sense of customer-generated input and transforming this into software improvements. • Knowledge exploitation: The implementation of elicited customer knowledge into software improvements.
Outcomes	Two dimensions: • PSD contribution: The extent to which the community use is valuable to the PSD process in terms of software improvements. • Customer acceptance: The extent to which customers accept the community interventions made by the commercial software firm.

Table 1: Central concepts

5 Research Methodology

5.1 Research Site

Daydream Software is a small computer game developer with its headquarters in Umeå, Sweden. At the beginning of this study, the firm employed around 65 people and had developed two computer games: Safecracker and Traitors Gate. The relative success of these two games had attracted a significant international customer base, making an early stock exchange quotation in 1996 possible.

Our selection of Daydream as the case for this research can be traced to a number of factors. Daydream is a software firm that focuses on a type of packaged software, computer games, that must be up-to-date with technological and societal trends. This does not only imply that software developers must be competent in a particular domain but also willing to acquire new knowledge over time. Interaction with software customers in an online customer community can therefore be seen as an important source for acquiring such new knowledge. Moreover, the development and introduction of Daydream's third computer game—Clusterball—involved a commitment to improve its PSD processes by means of an online customer community—the Clusterball community. This fact coincided well with our intentions to study the role of community use for improving PSD.

The technical basis for the Clusterball community was a web application (www.clusterball.com, see also figure 2) and the game itself. The web application provided electronic discussion forums, fan website links (links to unofficial Clusterball websites), product information, and software downloads. The game included a pre-game chat allowing for players to meet before each gaming session to discuss tactics and to share experiences from previous gaming sessions. Given the ambition to improve Daydream's PSD process, the Clusterball community represented an attempt to cater for situated customer knowledge.

6 Research Design

The research reported in this paper builds on a 17 month (January 2000 – May 2001) interpretive case study (Klein and Myers 1999). Interpretive researchers examine research phenomena through investigating the different meanings people assign to them (Orlikowski and Baroudi 1991). This procedure builds on that people act-in-the-world on the basis of their subjective and inter-subjective creation of meaning. Whether the meanings associated with the phenomena are factual descriptions of the world is not an issue for the interpretive researcher. What matters is rather the extent to which these meanings can help the researcher understand why people act as they do.

The interpretive case study is particularly suited for studying research contexts in which different actor groups' views of the research phenomenon can be expected to be divergent. In fact, examining multiple interpretations of the research phenomenon is at the heart of interpretive research (Klein and Myers 1999). In our case, we suspected that the gap between Daydream's commer-

Figure 2. The Clusterball website

cial interest and the non-commercial interest of community participants would be central to understand benefits and limits of community use in PSD. Therefore, an interpretive frame of reference would be useful for exploring the prospects of community use in PSD, as these would be reflected in the assumptions, beliefs, and knowledge held by both parties.

The research process consisted of three inter-related phases (see Table 2). First, an exploratory study was conducted. In this, we aimed at getting an initial understanding of Daydream and the context of which the firm was part. During this period we attended company meetings and discussions, reviewed early design documents and spent time observing the work practices of Daydream employees. In this, our aim was to get a comprehension of the setting, culture, and study topic. During this period, we also presented our research interests and the way in which Daydream provided an interesting empirical setting for our work. As a result of this phase, we got an initial understanding of Daydream and its PSD process, as well as a comprehension of their aim of using an online community for involving customers in this process.

Second, and as a significant starting-point for acquiring an insider view of the research phenomenon, the first author of this paper spent most of her

working time during 6 months at the Daydream site. Supporting engagement between researchers and research subjects (Nandhakumar and Jones 1997), this in-depth study, and the observational studies that were carried out as part of this, was important impetus for developing a tentative understanding of the research setting. While observational studies, and the daily documentation of these, were the main activity during this phase, data sources such as meeting minutes (formal company meetings and project group meetings) and organizational documents (technical and design documents, shareholders documents, press releases, company presentations etc.) were also used to get an enhanced understanding of Daydream and its PSD process. Furthermore, to gain an understanding of the community and the relationship between developers and community members, website data, i.e. community postings, was considered an important data source throughout the study. On a daily basis, the community forum was monitored and postings concerning the game and improvement of this were printed and archived. Especially, postings revealing developer-customer relationships were of interest. These were used to help us in our understanding of how knowledge was acquired and exchanged among developers and customers. Also, the postings revealed the extent to which developers were active in the community and in what situations there was developer-customer interaction. In combination with the observations, the postings admitted for a detailed understanding of the interventions made in the community and the way in which community knowledge was catered for in Daydream's PSD processes. Also, patch specifications, i.e. documents revealing improvements made to the software, were studied in order to see to what extent community knowledge resulted in actual software improvements. In total, six patches were included in the study and each of these was documented on 1-3 written pages.

Finally, a complementary data collection phase was conducted as a rounding-off on our active presence at Daydream. Here, the pre-understanding gained during the two earlier research phases was used as a basis for performing qualitative interviews with Daydream employees as well as for designing an online survey with community members. In all, 11 interviews were carried out with Daydream employees ranging from managers, marketing people, and administrators to web designers and developers. All interviews were of an open character, i.e., we did not direct the interview too closely but instead allowed the respondents to express their own views in order to gain as much richness as possible for further interpretation and analysis. The interviews lasted for about 90 minutes and they were all recorded and transcribed. In addition to the interviews, the complementary data collection phase consisted of an online survey that was sent out to 200 Clusterball players ranging from 'newbies', i.e., not very experienced players, to 'grand masters', i.e., very

experienced players who had registered in the customer database and played the game the month before the survey was sent out (i.e., October 2000). The survey, consisting of multiple choice questions as well as free text alternatives, was distributed in November 2000 and the final answers were collected in February 2001. With a response rate of 52 percent (104 respondents), the survey helped us in (1) our understanding of the customers and their apprehension and application of the customer community for the purpose of knowledge sharing, and (2) our understanding of the customers' view on how their knowledge was elicited and exploited by Daydream in the PSD process. However, despite a relatively high response rate indicating a very rich data material, the result of the survey was a bit of a disappointment. While the multiple choice questions were carefully answered by a majority of respondents, the free text alternatives—which were intended to provide us with individual reflections in similar to those attained when conducting qualitative interviews with Daydream employees—were only briefly answered by a minority of the respondents, providing us with limited data material reflecting the view of individual customers. As a result of this, the online survey was used only as a complement to other data sources such as observations, interviews and website data, i.e. community postings. As a result of the complementary data collection phase we attained an increased understanding of the developer-customer relationship and the perception of community use as useful for software improvements.

6.1 Data Analysis

The findings generated in this research have emerged as an iterative process between theoretical conceptions and empirical data (Klein and Myers 1999). As in most long-term interpretive research, the initial conceptual apparatus—encompassing certain assumptions, beliefs, and rationale—consequently transformed over time. Our preconceptions, empirical data, and progressive interpretations of the research phenomenon worked as intertwined elements in the process of analyzing the case.

The collection and analysis of empirical data were undertaken as parallel activities, even though the analysis prolonged after ending the empirical work. Transitions between parts, such as community postings and Daydream's software development context, and wholes such as the knowing-in-practice literature and the social process model of user-analyst relationships (Newman and Robey 1992) were necessary to successively formulate our understanding of community use in PSD. Used as broad categories in our analysis, the knowing-in-practice literature (see e.g., Boland and Tenkasi 1995; Tsoukas 1996;

Research phase	*Data material*	*Research goals*
January – March 2000; Exploratory study	• *Observational data* (weekly written personal notes) • *Meeting minutes* (project meetings 1/week: duration 1-2 hours)) • *Technical/design documents* (specs. of Clusterball software and network configuration) • *Website data* (daily community forum print-outs)	Initial understanding of Daydream's PSD process and their intention of involving its customers in this
April – September 2000; In-depth study	• *Observational data* (daily written personal notes) • *Meeting minutes* (project meetings 2-3/week: duration 1-2 hours) • *Technical/design documents* (specs. of Clusterball software and network configuration) • *Website data* (daily community forum print-outs) • *Patch specifications* (specs. of six software patches)	Enhanced understanding of Daydream's PSD process and the developer-customer relationships evident in the online community
October 2000 – May 2001; Complementary data collection	• *Qualitative interviews* (11 interview respondents: duration 1,5 - 2 hours) • *Online survey data* (104 respondents) • *Website data* (daily community forum print-outs)	Reveal the role that developers and customers attributed to the online community in terms of leveraging community knowledge in the PSD process

Table 2: Overview of research phases and data material in the Daydream study

Brown and Duguid 2001; Orlikowski 2002) provided us with an apparatus for our practice-based understanding of concepts such as knowledge, practice, and community. Similarly emerging from iterative transitions between the theoretical model and the data, the selection of episodes and encounters highlighted in the case description was guided by the social process model of user-analyst relationships (Newman and Robey 1992). In this selection, customer

knowledge was of primary interest and the literature helped us articulating a plausible understanding of this complex phenomenon.

7 The Clusterball Case

7.1 Antecedent Conditions

Daydream's plan to develop a new computer game was formed in 1998. Since this game was envisioned as the first online game that would be distributed, paid, and played over the Internet, considerable time was invested in developing a new network protocol, micro-payment functionality and a customer relationship management (CRM) application interface. The development of these new and, at that time, immature technologies was both time-consuming and complex, and already at an early stage the game was delayed. At this stage, the PSD process was basically an undertaking between in-house software developers and external technology vendors and consultants. The developer-customer links used at Daydream in the development of previous software packages had been e-mail, focus groups, and trade shows.

7.2 Encounter 1: The Clusterball Website

As the first interface towards customers, the Clusterball website was set-up in early 2000. Since the game was still under development, the website merely contained game information and development progress reports. In other words, the website was primarily an advertising tool, intended to stimulate interest among earlier Daydream customers as well as to provide feedback to impatient investors in the small stock-listed firm. In parallel with the website development, the software was beta-tested by 200 players. These were recruited via the Clusterball website where customers could register in the customer database, as well as via personal contacts among Daydream employees. Approaching the official game release in July 2000, the website was continuously developed to include payment functionality and advanced graphics. At this stage, there were already hundreds of registered members in the Clusterball community.

7.3 Episode 1: Bug-Reporting

Following the game release, customers had immediate concerns with getting the software to work with their various technical configurations. Only during the first month, two patches dealing with installation, start-up/ configuration problems and host errors were released. These patches were direct responses to community postings, revealing the kind of problems typical for software that suffers from tight deadlines (e.g., little time for comprehensive testing). Illustrative postings from this period were:

> "Right when the loading screen appears I get an illegal operation and I have to close it. This happens every single time. I have a diamond ViperV550, running 1280x1024 32bit, and win 98. I have had some problems with other games not switching to direct 3d mode but nothing like this. Please help, I'm very annoyed".

> "Hi, My crashes end with: CLUSTERBALL caused an invalid page fault in module CLUSTERBALL.EXE at 015f:0054e7e4. It crashed mid-game. I have a 450 Athlon, 256mb, GeeForce256. Any suggestions? Thanks".

7.4 Encounter 2: The Community Manager Initiative

Given the customer involvement vision, the early use of the Clusterball community came as a little disappointment. Even though the community was helpful for corrective bug-fixing, this type of customer input reflected immediate customer concerns rather than the developer-customer relationships envisioned in setting up the Clusterball website. Seeking such relationships, Daydream intensified the efforts to keep customers updated with Clusterball information. In complementing general information such as technical specifications, screenshots, and game descriptions; both software developers and marketing people intensified publication of up-dated information including Clusterball news, technical support, and FAQ additions. Indeed, such information was considered necessary for supporting community knowledge building. The manager at the time commented:

> "You have to encourage the players, have a dialogue with them and make sure that they can contact us and communicate with us as well as with all the other players…this can be done in electronic forums, on chat-lines or through fan websites developed by individual players and promoted by us."

To handle the overwhelming amount of customer feedback during the month following the software release, Daydream appointed a 'community manager' in August 2000. As a link between developers and customers, the community

manager was responsible for stimulating customer feedback as well as to make sure that this was implemented in the PSD process. This activity was considered important for building a sense of trust among customers, whose suggestions then would be better catered for in Daydream's PSD process.

The community manager viewed himself as a Daydream representative in the community, directed at encouraging customer feedback as well as for feeding customer suggestions into the PSD process:

> "…whenever I enter the forum I do it as a Daydream employee, which is very important to remember. Thus, it is not my personal opinions—but instead 'the Daydream view'—that I am supposed to deliver. (…) As a manager, I try to collect all the ideas and turn them into software improvements by handing them over to the developers. It is easier if this is done by one person so that the developers don't have to keep track of all ideas themselves. I know what they are doing and what people to ask for certain things and in that way I think we get a smoother and faster process in implementing new features of the game."

In other words, the community manager played an important role in encouraging customer suggestions, making sense of these suggestions, and translating their meaning into product suggestions that could be handed over to software developers. As can be seen, the community manager viewed customer suggestions as a key component in the PSD process:

> "One very important thing in this is to remember that if you once invited the customers to be part of the development process…to make them aware of that they have impact…then you must also see to it that the suggestions they come up with are implemented. Things must happen on the basis of community discussions and it must be evident that they are able to influence the software development process in the way we told them that they would."

Besides monitoring community postings, the community manager actively stimulated new suggestions for software improvement. Typically, the manager intervened with a clear goal in mind, e.g., a new patch release. Consider the community manager's posting in relation to the third Clusterball patch:

> "Hi! In developing the next Clusterball patch we would like to know what features you most of all would like to see in the game…are there anything missing in the game right now and what is it that you all really want us to implement? Please, post your suggestions to the community forum…we are looking forward to seeing them."

7.5 Episode 2: Enriched Developer-Customer Relationships

As a response to this invitation, numerous postings were submitted to the community. Typical examples, which later were transformed into software improvements, were:

> "My experience of DNF is that I send data, but don't receive anything, and when enough time has passed, my computer evaluates this as the server having gone down (not closed, that is a different message), and ends the session."
>
> "I would like to improve the match making—to allow the possibility to find other players closer to my skill level".
>
> "I would like to be able to set a minimum and maximum player ranking when I host a game."
>
> "I would like to have the possibility to talk to other players while waiting for a game".

The third Clusterball patch was released in mid October, 2000. At that time, the peak of corrective bug-fixes seemed to have passed, and consequently, Daydream could pay more attention to functional improvements and additional features of Clusterball. In response to customer suggestions as exemplified above, additional functionality such as replay and recording, a DNF feature (players still get points even if they 'did not finish' the game), improved match-making capabilities (to lock a game server on a min/max player ranking basis), a pre-game chat, and team-play ranking were implemented. The community manager noted:

> "The first patches were developed almost exclusively on the basis of community discussions in which customer needs were evident. For example, there was the replay function so that people can record their games and look at them afterwards, the team play ranking in which every individual team gets points and are ranked in a system and a lot of different search and sorting possibilities so that it is easier to keep track of different players, different venues and different hosting servers."

The direct link between customer suggestions and Clusterball patches was also pointed at by one of the software developers:

> "The pre-game chat was a direct result of customer suggestions. Very soon many players felt that they wanted a place were they could meet and talk before joining a gaming session."

As a direct developer-customer link, the community served many purposes and the community manager role was appreciated by customers. One of them expressed the following in our online customer survey:

> "It is a good idea since we now know that someone is taking care of all the suggestions…hopefully in a systematic way. Even though they might not always be implemented anyway…but it feels better".

Also, the overall effort by Daydream to strengthen the developer-customer link was indeed appreciated by customers:

> "I think I can influence almost all things…Daydream has been working very closely in cooperation with the community".
>
> "I think the programmers of Clusterball will listen to peoples' voices because they really want this game to be as good as possible".
>
> "I think peoples' suggestions to new features are definitely taken into consideration and I think that's a great idea. The people at Daydream seem to be open to suggestions from us players".
>
> "The community helps me influence the people at Daydream. I talk to them there and I think they listen to us players and try to make the game as good as we want it to be".

7.6 Encounter 3: Clusterball Ambassadors and School

In November 2000, Daydream appointed what they called 'Clusterball ambassadors' as to further strengthen their relationship with customers. These ambassadors were customers playing Clusterball on a daily basis and contributing extensively to the community in terms of postings. Since the ambassadors would be regarded as any ordinary player by the rest of the community, Daydream reasoned they could obtain information that would not be presented to, or appreciated by, Daydream employees. By the end of November 2000, there were five ambassadors—three in the US, one in Germany and one in Italy. The ambassadors' role in supporting knowledge building and elicitation was recognized by the community manager:

> "Even if my job is to keep track on the community I am not an ordinary player who enters the game whenever I like to or with the same prerequisites as any other player…the ambassadors are ordinary players…in that sense I work as a link between the customers and Daydream while the ambassadors work as a link between the customers and me."

The role of the ambassadors was further explained by the community manager:

> "…their [the ambassadors'] duty is to be active participants in the forum discussion, help 'newbies' in the game and see to it that people get answers on their questions when entering the Clusterball world"

Furthermore, the ambassadors helped Daydream in administrating different community events—something that was appreciated by the community manager:

> "To this day, the ambassadors have arranged their own tournaments and helped us in the administration of different events, they have guided new players and helped them in learning the game and so on…"

In addition to knowledge building and elicitation activities initiated by Daydream, Daydream also sanctioned the initiative to start the 'Clusterball School'. This initiative originated from one of the most well-known and frequent players in the community. The aim of the school was to help new players to learn the game faster and thereby making them better prepared for gaming sessions. Daydream viewed the initiative as a way to attract new players to the game as well as for players to further develop their gaming skills. The initiative was appreciated by the community manager:

> "The school is great…all people seem to like it. People get to know each other without always having Daydream people around. They can learn by themselves and ask each other when they need help".

In other words, the Clusterball School seemed to increase customer-customer links. One of Daydream's software developers noted that:

> "I think the school helps people to get to know each other. Experienced players meet with new players and they can develop relationships that probably will last even after they have played the game. Also, I think they enjoy the game more when they know the opponents…it is good for the overall community atmosphere".

7.7 Episode 3: Ambiguity following Intensified Community Engagement

Customer-customer links, as manifested by the Clusterball School and the community forum, were appreciated by customers too:

> "In the community I can learn from more experienced players and from the developers".

> "The community is good when you need help. And it is a great way of sharing your own experiences as well as for learning from other players. I know I have learnt a lot".

> "It gives people a chance to give suggestions about the game, about other players and get advice for the game. I think it is a great place for 'newbies' [not very experienced players] too, to give them an idea of what's going on and how to play the game".

After a series of knowledge building and elicitation activities including the community manager role, the ambassador role and the Clusterball School, Daydream could relax somewhat at the turn of the year. Accordingly, two Clusterball patches dealing with only minor technical problems were released. For example, the forth patch solved graphics- and sound problems as pointed out by several customers:

> "I got an ATI Rage 128 graphics card... but when I play Clusterball, all the ships and balls are black. I need some help."

> "…the sound is still a problem in XP with SB Live sound card".

Following this, the fifth patch was released in February 2001. This patch—the GL Setup patch—could detect whatever graphic card customers used and hence, install the latest drivers for that particular graphic card.

During the spring 2001, lots of customer suggestions on new features were posted to the community. This may be traced to the fact that at that time the game had been around for some time and there was the opportunity for players to reflect upon major changes and improvements instead of only minor bug fixes for solving immediate operability problems. In response to these postings, a sixth patch was released in April, 2001. This patch included improved joystick support (such as 'twist handle' functions), LAN-play without restriction of Internet access for host and improved artificial intelligence in the training (offline) mode. In addition, corrective development was also done. These corrections handled, e.g., a crash bug in the pre-game chat, a freeze bug when viewing replays, a throttle bug on joysticks, and a DNF-bug in match history.

While the sixth patch included new functionality triggered by customer suggestions, however, there were also several suggestions that were ignored by the developers at Daydream. Examples of such suggestions were:

> "Make more then just the ship playable, most other games have more than one model to choose from... Maybe there could be a model editor where players could make their own ships…".

> "Could there be a 'viewer system' so that my friends could watch other people play before they participate themselves?".

"I would like to see a password for the server when hosting a match. Setting the number of games or how long time the dedicated server should run. It would also be great if it was possible to send messages to the players when running a dedicated server".

While reflecting customer concerns, these suggestions were never put attention to. Clearly, there were limitations to what the customer community could influence in terms of development, despite the fact that they had been encouraged to participate in Daydream's PSD process. These limitations were recognized and explained by the software developers:

"There were requirements that were too costly or too technically difficult to realize. Some suggestions we simply didn't implement because we didn't agree with them...for example some of the suggestions about the gameplay we didn't like and therefore never implemented."

"The features that will influence us the most are those that players have difficulty with and that Daydream themselves are not 100% happy with. I don't think Daydream would be happy with changing the gameplay too much, quite correctly, because it would affect existing players, but they have shown that they want to implement suggestions and will add new modes and features to expand the game according to player suggestions without changing the playability of the game".

"Much of what was implemented we already recognized ourselves, since we of course played the game in the same way as all other players. However, we prioritized all input we got from the community and we always chose features that the majority of customers wanted and that we thought would be the most appreciated by the community."

As suggested above, there existed contradictory views on the actual role that the community played in the development process of Clusterball. While Daydream viewed the community as a resource for improving Clusterball, however, they were clearly selective in their assessment of what customer suggestions to implement.

This selectiveness was also recognized by customers. Some of the comments in our online customer survey were negative:

"I think the community can be used for reporting bugs and make small suggestions. I don't think I can influence the game itself, more like little improvements for a new patch—that is what they [Daydream] seem to listen to".

"The forum is made for people to chat about anything relating to Clusterball. What people do is that they post messages about what they like and dislike about the game. For Daydream—taking a look at the forum once in a while

would be a good idea as it would give them ideas on how to improve the game and make it more enjoyable".

"I feel that it would be wise for Daydream to read through the user and player comments in the forums, and try to implement features based on these suggestions. Ultimately, the users will determine what they like and don't like about the game and if it meets the expectations they are looking for. If not, they will continue looking for something else".

As indicated above, our case study at Daydream pinpoints benefits as well as limitations associated with community use for improving customer involvement in PSD.

7.8 Developer-Customer Links in the Clusterball Case

In sum, we identified three encounters that played a significant role in the further structuring of episodes of relatively stable developer-customer relationships. These encounters were: the Clusterball website, the community manager initiative, and the Clusterball ambassadors and school. Each of these encounters was followed by episodes characterized by a specific mode of community use in the form of knowledge building, elicitation, and exploitation. Each of these encounters were also associated with certain forms of PSD contributions and characterized by a level of customer acceptance. Table 3 summarizes the community use noted in our study of the Clusterball case.

The development and release of the Clusterball website initiated the first developer-customer encounter in the Clusterball case. The main issue at this point was to set up the technical infrastructure necessary for establishing a community. The episode following this encounter was characterized by a relatively detached relationship between Daydream and its customers. Mainly, the community was used for bug-reporting, and hence the suggestions elicited primarily concerned installation and configuration problems. Reflecting immediate software operability concerns rather than commitment to software improvement, the suggestions posted were straight-forward and unambiguous in character. Daydream monitored the community forum for bug-reports that could be useful for corrective maintenance of the software. While this was a rather smooth episode of community use, it failed to leverage other lessons learned in the community. For instance, the importance of customer-to-customer relationships was virtually unnoticed by Daydream and hence, no activities for supporting these were undertaken. In fact, it can be noted that without

	Encounters	*Episodes*	*Outcomes*
#1: Jan-Aug 2000	The Clusterball website	Knowledge building: • Game and development progress reports • Recruitment of beta testers Knowledge elicitation: • Monitoring of community postings for bug-reports Knowledge exploitation: • Software patches (#1&2) consisting of corrective bug-fixes and solutions for host error problems	PSD Contribution: • Community functioned as a mere bug-reporting system Tension: • Non-existent

Table 3: Community use in Daydream's PSD process

#2: Aug- Oct 2000	The community manager initiative	Knowledge building: • Website updates on Clusterball news and FAQ's • Community manager stimulation Knowledge elicitation: • Monitoring of community postings for bug-reports • Community manager participation in community activities and game playing ('insider view') • Community manager reports of customer needs and requirements as well as customer behavior to software developers Knowledge exploitation: • Software patch (#3) including corrective bug-fixes and functionality additions	PSD Contribution: • Community enriched developer-customer relations, which resulted in a set of functionality additions. Tension: • Minor

Table 3: Community use in Daydream's PSD process

#3: Nov 2000- May 2001	Clusterball ambassadors and school	Knowledge building: • Appointment of Clusterball ambassadors • Start-up of Clusterball School Knowledge elicitation: • Monitoring of community postings for bug-reports • Community manager participation in community activities and game playing • Community manager reports of customer needs and requirements as well as customer behavior to software developers • Clusterball ambassadors as links between customers and community manager and software firm Knowledge exploitation: • Software patches (#4&5) including corrective bug fixes • Software patch (#6) including functionality additions	PSD Contribution: • Community contributed to enriched developer-customer relationships, but also occasioned requirements conflicting with each other as well as with Daydream's commercial intentions. Tension: • Tension was occasioned by Daydream's selective assessment of customer suggestions.

Table 3: Community use in Daydream's PSD process

any substantial community interventions, the community worked as any ordinary bug-reporting system for Daydream's software developers.

Noting the simplistic community use during episode 1, Daydream attempted to develop a richer and more enduring relationship with its customers at the end of August 2000. In this regard, the appointment of a community manager was a step towards taking more active part in the community's situated knowledge building. This appointment was intended to stimulate knowledge building, as well as to improve Daydream's capability to elicit the knowledge that was built among community members. While community use was somewhat restricted in episode 1, episode 2 was characterized by improved developer-to-customer relationships. As a result of this, the knowledge elicitation not only included bug-report monitoring but also efforts to build an 'insider's view' of community activities. The community manager's daily participation in forum discussions and game playing contributed to such a view, making him part of the skills and practices enacted by community members. At this point, customers viewed the community manager's active community participation as a legitimate attempt to improve developer-customer relationships and to involve community knowledge in the PSD process.

The third developer-customer encounter consisted of recruitment of Clusterball ambassadors and establishment of a Clusterball School. Clusterball ambassadors were recruited for improving customer-to-customer relationships with the potential to support the knowledge building and elicitation processes and hence, improvement of the software. Complemented by the initiative to start the Clusterball School, the ambassadors augmented earlier knowledge building and elicitation activities as conducted by the community manager. Since the capacity of the community manager was limited in terms of workload and the degree to which he could act as a true 'insider', these community-driven knowledge building and elicitation activities targeted a broader range of community members. Triggered by this intertwined community relationship, more profound suggestions were posted to the community forum. However, few of these suggestions were implemented. Indeed, cycle 3 demonstrated limits to community use in PSD. Concurring with the individualistic culture of software firms (Sawyer 2000), many suggestions were rejected due to developers' strong images of what the game would be like. In addition, more costly or technically difficult changes were often rejected because management feared that they would not pay off in increased sales or resonate well with strict time-to-market deadlines. In particular, this reflected a fear of not representing the majority of customers' needs and, in this way, leaving a large customer segment outside the negotiations of intentions (cf. Schwen and Hara 2003). Despite Daydream's intimate community use, episode 3 can be considered incomplete in that intensive activities for knowledge building and elicitation lead to limited exploitation in terms of software improvements.

8 Implications

Earlier research highlights the specific firm and firm environment conditions associated with PSD (Sawyer 2000). It also outlines how these conditions make direct customer involvement difficult in PSD. Typically, customer involvement is based on indirect links such as intermediaries or customer surrogates (Keil and Carmel 1995). Given the central role that customers can play in successful product development (von Hippel 1986; Nambisan 2002), we therefore investigated how online customer communities can be used to improve customer involvement in PSD. Given the tension between organizing principles of commercial software firms and those of voluntary community participation, we were particularly interested in relationships between software developer intervention and customer acceptance of such intervention over time.

In order to do this, we adapted Newman and Robey's (1992) social process model of user-analyst relationships to the PSD context. We used this model for analyzing the process by which Daydream Software established and used a customer community for developing and refining its online computer game Clusterball. The findings yielded through this analysis illustrate the feasibility of the model for analyzing the role of community knowledge in PSD. The research described in this paper has important implications for research and practice.

8.1 Research Contributions

We believe that our adaptation of Newman and Robey's (1992) social process model extends earlier work on developer-customer links in PSD (Keil and Carmel 1995) by providing a lens with which to analyze community use for such linking. Our study conveys challenges residing at the boundary between commercial firm interests and the voluntary nature of online community participation. We believe that this fact has consequences for the different processes of community use in PSD.

In particular, our distinction between different community intervention types facilitates a detailed analysis of the delicate balance between substantial community-enabled software improvement and customer acceptance. In applying the social process model, we found differences between types of community interventions with regard to customer acceptance. Indeed, software firms can more easily take knowledge building initiatives without jeopardizing customer acceptance. This can be traced to the fact that software customers' knowledge creation and sharing naturally takes place within a

community of successful software. While the software firm can support the initiation of this process, the creation and sharing processes themselves unfold through software customers' mutual engagement in a specific software product. In contrast, community knowledge exploitation represents the most contentious community intervention. Since the software firm operates in a commercial environment, issues related to business environment and firm culture naturally become ingredients in software improvement decisions. Partly outside the scope of what takes place in the community, commercial interests will almost always be prioritized in cases where these contradict community input. Such prioritization is typically difficult to explain to devoted community members sharing a different opinion. While such prioritizing always exist in software development projects, it can be harder to maintain trust in prioritizing based on commercial appropriateness rather than software excellence. This is a core difference between community use in PSD and open source software development.

Our research also suggests that the richer developer-customer relations that the software firm manages to establish, the higher are the stakes. While the two last episodes were the most valuable to Daydream in terms of PSD contribution, this increased value coincided with greater tensions between commercial and community interests. The enriched developer-customer relationships following the Clusterball ambassadors and school initiatives resulted in productive software improvement suggestions. However, it also created higher expectations among customers about subsequent software changes. Since some of these suggestions were perceived not to pay off in increased sales or were technically complex to implement, most customer suggestions were omitted. This caused ambiguity about the role and impact of the customer community.

With regard to the community literature, it must be emphasized that community use in PSD can never fully embrace the situated learning taking place in an online community-of-knowing. As highlighted by Schwen and Hara (2003), the online community literature tends to romanticize the notion of community. Looking at our case, the mere application of the notion of 'use' in 'community use' suggests a kind of detached relationship to the complicated and situated learning characterizing customer-to-customer relationships and identities produced and reproduced over time. In this regard, communities can never be controlled or exploited in conventional terms. In the spirit of Wenger's (2000) discussion on community of practice design, however, the investigation presented in this paper should be seen as an attempt to describe what actions software firms can take to recognize, support, encourage, and nurture customer communities in order to improve their software development through community knowledge.

8.2 Implications for Practice

Understanding the linkages between community interventions and customer acceptance of such intervention may reduce the likelihood of using communities in PSD in a way that fails to pay off in form of software improvement. In highlighting that successful community knowledge use can be a challenging task, our research is therefore useful for commercial software firms that plan to establish and maintain communities around their software products. As long as community knowledge is merely exploited for software operability concerns, such completion is relatively unproblematic. Here, community knowledge use is a straight-forward process consisting of the setting up of a web infrastructure for stimulating, monitoring, and implementing customer suggestions. Community use in PSD is also useful for minor functionality additions. However, such software improvement is more complex in that it requires organizational efforts for deepening the relationship with customers.

In this regard, our research suggests that software firms committed to enrich their developer-customer links must match their proactive knowledge building activities with well-chosen elicitation and exploitation activities. This is particularly true in relation to major software changes. While community members may be encouraged to devote time and effort to software improvement, they are likely to be disregarded when commercial interests are contradicted. In such cases, the omission to exploit community knowledge can undermine the trust needed for successful community use. Indeed, while the knowledge building and elicitation reflect altruistic ideals and mutual engagement grounded in a common interest, knowledge exploitation is located within the realm of the software firm. Steered by commercial ideals, this makes vibrant community input subject to firm prioritization. Given the risk of causing ambiguity by overseeing customer suggestions, it is therefore vital that software improvement policies are communicated to the community. While long-term directions for such policies are questions for management and lead developers to handle, its establishment, enactment, and maintenance at the operational level may be a key task for community managers and closely firm-associated customers.

9 Conclusions

This paper sets out to explore the benefits and limits of online communities for improving developer-customer links in PSD. The paper adapts Newman and Robey's (1992) model for analyzing community use in PSD as an undertaking

between a software firm and its customer community. The model depicts community knowledge use as a social process consisting of antecedent conditions, encounters, episodes, as well as outcomes in forms of PSD contribution and customer acceptance. Episodes consist of a set of community interventions (knowledge building, knowledge elicitation, and knowledge exploitation) maintaining a stable mode of developer-customer relationship. Such episodes are punctuated by encounters, i.e., changes in the software firm's community interventions and/or customer acceptance of such intervention, leading to a new episode of stable developer-customer relationship. On the basis of an application of this model to the Clusterball case, we outline important implications for both research and practice.

On a final note, our research was conducted within a computer game setting. Due to the extraordinary motivation found in such communities, this limits our ability to generalize the model to other contexts. As in all interpretive case studies, our findings should be seen as tendencies rather than predictions (Walsham 1995). However, we do believe that the model and hence, the opportunity for community knowledge use, is applicable to settings in which the software concerned triggers customer motivation and engagement equivalent to that found in computer gaming. Since "theory may never be scientifically generalized to a setting where it has not yet been empirically tested and confirmed" (Lee and Baskerville 2003, p. 240), further research is however needed to validate and refine our findings for new research settings.

Acknowledgments

Thanks are due to the SJIS editor, Matti Rossi, three anonymous reviewers, and Rikard Lindgren for useful comments on earlier versions of this manuscript.

References

Abts, C., "COTS-Based Systems (CBS) Functional Density—A Heuristic for Better CBS Design," In *Proceedings of COTS-Based Software Systems*, ICCBSS 2002, Orlando, USA, February 4-6, 2002.

Andreu, R., and Ciborra, C., "Organizational learning and core capabilities development: the role of IT," *Journal of Strategic Information Systems* (5), 1996, pp. 111-127.

Bergquist, M., and Ljungberg, J., "On The Power of Gifts: Organizing Social Relationships in Open Source Communities," *Information Systems Journal*, (11), 2001, pp. 305-320.

Boland, R.J., and Tenkasi, R.V., "Perspective Making and Perspective Taking in Communities of Knowing," *Organization Science*, Vol. (6:4), 1995, pp. 350-372.

Brown, J, and Duguid, P., "Knowledge and Organization: A Social-Practice Perspective," *Organization Science* (12:2), 2001, pp. 198-213.

Carmel, E., "American Hegemony in Packaged Software Trade and the 'Culture of Software'," *The Information Society*, (13), 1997, pp. 125-142.

Carmel, E., and Sawyer, S., "Packaged software development teams: what makes them different?" *Information Technology and People*, (11:1), 1998, pp. 7-19.

Clegg, C. W., Waterson, P. E., and Axtell, C. M., "Software development: knowledge-intensive work organizations," *Behaviour & Information Technology*, (15:4), 1996, pp. 237-249.

Cusumano, M. A., and Shelby, R. W., *Microsoft Secrets: How the Worlds Most Powerful Software Company Creates Technology, Shapes Markets and Manages People*, The Free Press, New York, 1995.

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

Finch, B.J., "Internet discussions as a source for consumer product customer involvement and quality information: an exploratory study," *Journal of Operations Management*, (17), 1999, pp. 535-556.

Gersick, C.J.G., "Revolutionary Change Theories: A Multilevel Exploration of the Punctuated Equilibrium Paradigm," *Academy of Management Review*, (16:1) 1991, pp. 10-36.

Greenbaum, J., and Kyng, M., (eds.), *Design at Work: Cooperative Design of Computer Systems*, Erlbaum, Hillsdale, 1991.

Grudin, J., "Interactive systems: Bridging the gaps between developers and users," *IEEE Computer*, (24:4), 1991, pp. 59-69.

Hagel, J., and Armstrong, A., *Net Gain - expanding markets through virtual communities*, Harvard Business School Press, Boston, 1997.

Hamel, G., and Prahalad, C.K., *Competing for the future*, Harvard Business School Publishing, New York, UK, 1994.

Henfridsson, O., and Holmström, H., "Developing E-commerce in Internet-worked Organizations—A case of customer involvement throughout the computer gaming value chain," *DATA BASE*, (33:4), 2002, pp. 38-50.

von Hippel, E., "Lead Users: A Source of Novel Product Concepts". *Management Science*, (32:7), 1986, pp. 791-805.

Holmström, H., "Virtual Communities as Platforms for Product Development—an interpretive case study of Customer Involvement in Online Game Development," in *Proceedings of 22nd International Conference on Information Systems (ICIS)*, December 16-19, 2001, New Orleans, LA, USA.

Holmström, H., and Fitzgerald, B., (forthcoming). "Using Virtual Communities for Software Maintenance," *Journal of Organizational Computing and Electronic Commerce - Special Issue on "Collaborative Internet Applications"*.

Holmström, H., and Henfridsson, O., "Customer Role Ambiguity in Community Management," in *Proceedings of 35th Hawaii International Conference on System Sciences (HICSS 35)*, January 7-10, 2002, Big Island, Hawaii.

Humphrey, W.S., *Managing the Software Process*, Addison-Wesley, Reading, 1989.

Keil, M., and Carmel, E., "Customer-Developer Links in Software Development," *Communications of the ACM*, (38:5), 1995, pp. 33-44.

Kim, A. J. *Community Building on the Web*, Peachpit Press, Berkeley, 2000.

Klein, H. K., and Myers, M. D., "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems," *MIS Quarterly* (23:1), 1999, pp. 67-93.

Lee, A. S., and Baskerville, R. L., "Generalizing Generalizability in Information Systems Research," *Information Systems Research* (14:3), 2003, pp. 221-243.

Lee, G. K., and Cole, R. E., "From a Firm-Based to a Community-Based Model of Knowledge-Creation: The Case of the Linux Kernel Development," *Organization Science*, (14:6), 2003, pp. 633-649.

Little, T., "Value Creation and Capture: A Model of the Software Development Process," *IEEE Software* (21:3), 2004, pp. 48-53.

Ljungberg, J., "Open Source Movements as a Model for Organizing," *European Journal of Information Systems*, (9:3), 2000, pp. 208-216.

Markham, A. N., *Life Online: Researching real experience in virtual space*, Altamira Press, London, 1998.

Mathiassen, L., Pries-Heje, J., and Ngwenyama, O., (eds.), *Improving Software Organizations: From Principles to Practice*, Addison-Wesley, Boston, 2002.

Nambisan, S., "Designing virtual customer environments for new product development: toward a theory," *Academy of Management Review*, (27:3), 2002, pp. 392-413.

Nandhakumar, J., and Jones, M., "Too close for comfort? Distance and angagement in interpretive information systems research," *Information Systems Journal*, (7), 1997, pp. 109-131.

Newman, M., and Robey, D., "A Social Process Model of User-Analyst Relationships," *MIS Quarterly*, (16:2), 1992, pp. 249-266.

Orlikowski, W. J. "Knowing in Practice: Enacting a Collective Capability in Distributed Organizing," *Organization Science*, (13:3), 2002, pp. 249-273.

Orlikowski, W. J., and Baroudi, J. J., "Studying information technology in organizations: Research approaches and assumptions," *Information Systems Research*, (2:1), 1991, pp. 1-28.

Prahalad, C.K., and Hamel, G., "The Core Competency of a Corporation," *Harvard Business Review*, (68:3), 1990, pp. 79-91.

Rheingold, H., *The virtual community: Homesteading on the electronic frontier*, Addison-Wesley, Reading, 1994.

Sawyer, S., "Packaged software: Implications of the differences from custom approaches to software development," *European Journal of Information Systems*, (9), 2000, pp. 47-58.

Schwen, T. M., and Hara, N., "Community of Practice: A Metaphor for Online Design?" *Information Society*, (19), 2003, pp. 257-270.

Sharma, S., Sugumaran, V., and Rajagopalan, B., "A framework for creating hybrid-open source software communities," *Information Systems Journal*, (12:1), 2002, pp. 7-25.

Tyre, M. J., and Orlikowski, W. J., "Windows of Opportunity: Temporal Patterns of Technological Adaptation in Organizations," *Organization Science*, (5:1), 1994, pp 98-118.

Walsham, G., "Interpretive Case studies in IS Research: Nature and Method," *European Journal of Information Systems*, (4:2), 1995, pp. 74-81.

Weick, K. E., *The Social Psychology of Organizing*, McGraw-Hill, New York, 1979.

Wenger, E., *Communities of Practice: Learning, Meaning and Identity*, Cambridge University Press, Cambridge, 1998.

Wenger, E., "Communities of Practice and Social Learning Systems," *Organization*, (7:2), 2000, pp. 225-246.

Zachary, G., *Showstopper: The Breakneck Race to Create Windows NT and the Next Generation at Microsoft*, The Free Press, New York, 1994.