FROM OPEN SOURCE TO COMMERCIAL SOFTWARE DEVELOPMENT - THE COMMUNITY BASED SOFTWARE DEVELOPMENT MODEL

Completed Research Paper

Jie Yan
Associate Professor
Grenoble Ecole de Management
12, Rue Pierre Semard - BP 127
38003 Grenoble Cedex 01, France
Jie.yan@grenoble-em.com

Xinqiao Wang
Juling Infotech Inc.
21F, Kaipeng International
No.425, Gonghexin Road
200070, Shanghai, China
joe@huazisoft.com

Abstract

The successful practice of OSS leads to the intuition that integrating online software engineering community into the value chain of software company may be a solution to access qualified workforce and to reduce product cost. The emerging practice of crowdsourcing offers a potential solution for this attempt. Adopting an action research approach, the researchers collaborated with a software company in China and developed a crowdsourcing based software community development model, which consists of 3 elements: 1. online communities, providing the abundant low cost software developers with diverse technical capability and background; 2. crowdsourcing, providing incentive for developers' participation and also motivating competition; and 3. process management and quality control mechanism, borrowed from in-house software development practice, guaranteeing the product quality and fulfillment of project schedule. This Crowdsourcing Based Community Development (CBCD) model, as a new business model and a new method of organizing software development, was tested with real-life commercial software projects and proved to be effective.

Keywords: software development, open source software, crowdsourcing, action research, online communities, open innovation

Introduction

In recent years companies are increasingly making efforts to leverage external intelligent forces to create new products and services into market. The open innovation model shifts from the traditional inward focused R&D model to more outward looking management that abstracts ideas, knowledge, technologies and resources from external networks comprised of lead users, suppliers, new startups, competitors, universities and other social communities or organizations. Internet online community has emerged as one of these important external resources for innovation. The big number community members generally share some common amateurish interest and demographic characteristics, and therefore are often treated as sources of innovation by companies who seek intelligent input in designing new products or improving existing ones which target at these specific groups. The final users often have knowledge and ideas for innovation because they experienced the novel needs to improve the product in their daily life practice (Von Hippel 1998). Companies purposely release their demand for ideas, design and contribution in the Internet platform and motivate the community members to participate the innovation. Some companies have already made good success thanks to this community based innovation process (Lakhani and Panetta 2007).

Open source software (OSS) is a classic case of community based innovation (von Hippel and von Krogh 2003) and has been studied by extensive studies. An OSS project is generally developed voluntarily by a group of geographically distributed technical experts, who participate and contribute to the software project completely based on communication via Internet. OSS communities have developed a set of well-structured project management methods to initiate, design, coordinate, control and improve the process of software development by means of distant collaboration among unknown online community members. It was regarded as a successful way of knowledge and innovation creation (Lee and Cole 2003). This leads to the intuition that involving community developers in commercial software projects may be a possible solution to lessen the pressure software companies facing for searching qualified technical force and reducing cost. However, this is challenging given the significant difference between the altruism oriented OSS projects and the profit-chasing commercial projects.

The emerging practice of Internet crowdsourcing offers a potential direction for this solution. Crowdsourcing commercial software projects in online software engineering communities may offer company advantage by accessing the abundant low-cost intelligent and technical resources. However, different from the popular crowdsourcing projects which are generally based on idea contest and on individual work, software projects require collaboration among a group of developers and have tough restriction of budget, timetable and quality control. A strong mechanism integrating the merits of OSS development, crowdsourcing and in-house software engineering must be designed and implemented in order to make it possible to develop commercial software projects in online communities.

This research was initiated with the objective to develop a new software development model which is able to integrate the technical force of online communities into software companies' value chain with crowdsourcing practice. Taking an action research approach, the researchers worked collaboratively with a software company Juling Infotech Inc. in Shanghai, China (referred as Huazisoft, the company's product brand, in the following text). The researchers and the company management team worked together and developed the crowdsourcing based community software development model as well as the supporting IS platform. The model and platform were further improved by the prototyping approach (Naumann and Jenkins 1982; Connell and Shafer 1989; Baskerville and Wood-Harper 1998) by 4 real-life commercial projects. Huazisoft converted its business model to this new software development practice since 2011 and over 30 commercial projects were developed with the proposed model up to December 2012. The Crowdsourcing Based Community Development (CBCD) model, as a new business model and a new method of software development, is proved to be feasible, controllable and productive. It adds value both to the software company by technical capability improvement and cost cutting, and to community developers by an additional income.

Community based innovation, OSS and crowdsourcing

Community based innovation

Since the early 1990s, the Internet has been used as an enabling technology for long-distance communication and interaction, connecting people and generating a plethora of online communities.

Schubert and Ginsburg (2000) describe online communities as the union between individuals or organizations who share common values and interests using electronic media to communicate within a shared semantic space on a regular basis. Besides the communities for social and commercial purpose, many online communities are professional practices related, for example, scholars in academia (Koku and Wellman 2002), lawyers (Hara 2000), computer professionals (Wasko and Faraj 2000), and open source software developers (Lakhani and von Hippel 2003; O'Mahony and Ferraro 2004). These online communities provide opportunities, channels, and venues for professionals to share everyday work related resources, not just information, but also innovative ideas, solutions to specific problems, professional knowledge, and the latest thinking in their field of interest. Many participants treat such online communities as a place for learning and professional problem solving (Assimakopoulos and Yan 2006). Participants benefit from these communities by creating, accessing and exchanging new knowledge, expertise and innovative ideas not available in their local community of practice and working environment (Brown and Duguid 2001).

Online community has been regarded as a key venue of open innovation (Chesbrough 2006). By involving online communities into value chain, companies achieve a deeper understanding of consumer behavior and an earlier awareness of the upcoming market trends and changes without increase of the expenses on marketing research. It helps to speed up the innovation process by making the fuzzy front end manageable and reduces the failure rate of new product development. Many companies realize the value of online communities as a source of innovation, instead of using it solely as a channel of marketing. Lead users, who experience the needs for a new product or new function of old products, and therefore create novel ideas for innovation ahead of manufacturers (von Hippel 1998), are generally small in number, distributed among the large group of ordinary users, and often geographically dispersed. Online communities provide the convenience for them to gather together in the Internet platform. It has been found that there are many active lead users in online communities showing great enthusiasm in discussing innovation related issues (Sawhney and Prandelli 2000; Franke and Shah 2003; Fuller et al. 2006).

Among a wide range of topics in online communities, members actively exchange information, knowledge and ideas about products and share their thoughts about the opportunity to have a new product or to improve the functionality of old products. From manufacturers' point of view, online communities are convenient channel to access and gather such creative information. Many companies build up their own product community or organize brainstorming and idea competition in existing communities in order to exploit the intelligent resources (Franz and Wolkinger 2003). Moreover, many lead users are not only knowledgeable of creating original ideas of innovation, but also capable to actualize the ideas into new products (von Hippel and Katz 2002). Many enthusiastic members contribute their creativity and problem solving skills and directly involve in the design and engineering process of innovations. They actively response to the call for contribution from companies, elaborate detailed product concept, question and challenge it, select the options, personalize the features, vote to evaluate the results, and test the new products. For example, Duotone, a well-known snowboard manufacturer, initiated a design competition in the online community of the Austrian youth radio station FM4 and received many extraordinary blue prints (Franz and Wolkinger 2003).

In recent years the maturity of web 2.0 technology, characterized by great interactivity, global access, richness in contents, good convenience and low cost, further improves the efficiency of distant interaction among community members and between members and companies. It accelerates the 'free revealing' (von Hippel 1998) process transferring lead users' innovative ideas to companies, and improves the efficiency of community based innovation. Some online community platforms have already developed online design toolkits to facilitate members' product design (von Hippel and Katz 2002). By drag and drop the basic components, people can design and feature options, engineering constraints and process impacts are displayed in real-time to create the ideal product.

OSS and the development model

OSS development is an important and special case of community based innovation (Henkel 2006). The initiation of OSS projects often starts from a user's personal need of having a new software package. The information is published in the OSS community and is echoed by other users and programmers who have the same needs. The users and programmers collectively develop the software by self-selecting the technical roles they will take and the component they will contribute (Gacek and Arief 2004). The whole development process is openly visible in the community, the source codes are publicly downloadable and there is no management regime, financial budget or time planning during the development process. This user initiation and community involvement characterize OSS

development with the features of the 'user-centric innovation' (von Hippel and von Krogh 2003). Raymond (1999) describes the OSS development as a "bazaar" model characterized by developers' decentralized and even random activity in a flat and loosely connected network. OSS developers are independent actors who autonomously make decision about what, when and how to contribute to the project, contrasting to the classic in-house "cathedral" model where the development of software is designed, monitored and controlled by a central "master architect". Raymond believes the "bazaar" is supreme because the open source code enables users and OSS funs, who are often in a huge number in communities, to test, scrutinize, debug and refine the software or give quick feedback to the developers. New functionality and needs are rapidly reported and added to the software. This makes OSS have low defect density than proprietary software.

Despite its popularity, the bazaar model however is increasingly challenged by recent empirical studies. It was found that most OSS projects were not developed by dynamic and interactive developer groups, instead, were developed by very small-sized insolated teams. For example, among the nearly 100,000 projects in the flag OSS community SourceForce.org, most of them have only two or less developers (Christley and Madey 2007). Most of the projects are inactive and never release any version for distribution. Healy and Schussman (2003) found the median number of developers per project is just a little more than one. Krishnamurthy (2002) found that the median of the number of developers in OSS projects is four, and therefore believes that most OSS projects are developed in a close "cave", instead of in an open community.

It was founded that there exists an informal core-peripheral hierarchy in many OSS projects (Moon and Sproull 2000; Koch and Schnider 2002). The initiator of the project and some core members play significant role in the development process by leading the project, developing the majority of the components, verifying and selecting the patches uploaded by contributors, and mediating the confliction between developers. They have more power in directing and regulating the development process. For example, in Apache and Mozilla projects (Mockus et al. 2002), the core teams consist of 10 and 15 developers respectively who contribute most of the code and monitor the evolution of the project. The people reviewing the code and fixing bugs are an order of magnitude more than the core team, and the people reporting bugs and providing user feedbacks are an order magnitude more than those who develop and revise the code.

It also has been noticed that in some OSS projects the organizational model tends to be more formalized. Regular offline meetings are organized in order to better coordinate the development, for example, Apache conferences in US and Europe, the Zope/plone and Pypy development sprints, and the GNOME annual project conferences (German 2004), in which the core developers of the projects build up technical plan and timetable for future development. Another phenomenon worthy of noting is that in recent years many organizations and companies take part in the OSS development force. Companies release their proprietary software to be OSS or initiate new OSS projects in online communities. Instead of leaving the project completely 'free' of direction, the companies often present some guidance in terms of specification, functionality etc. to 'steer' the development. They also financially 'sponsor' the OSS projects and provide payment to the developers, who in many cases are employees of the companies (Lakhani and Wolf 2001).

Despite the above mentioned trends of change, OSS development model obviously is not possible to be adopted directly for commercial software development. The underlying altruistic and free-sharing philosophy of OSS community blocks the possibility to initiate a project for commercial purpose. The loose schedule management, weak quality control, high rate of hibernation/failure as well as the core-peripheral participating structure make OSS model inappropriate for commercial project. We have to find a way to overcome the above mentioned management and technical barriers in order to involve the community developers in commercial software development. This will be further discussed later.

Crowdsourcing in online communities

Crowdsourcing recently emerged as a mechanism to attract and motivate the community members to take part in the open innovation process. Companies publish in the online communities the information of their business challenges and call for solution by promising a monetary reward to the successful solution providers. Community members work for the projects and answer the public bidding by providing their ideas and solutions. The final winner will be determined by the company or by public vote of the community members, and get the cash rewards.

Case studies prove crowdsourcing in online communities is an efficient way of exploiting the low-cost intelligent resources to create contents, solve problems, collect innovative ideas and solution, and

even conduct R&D projects. Treadless, a t-shirt design company, holds weekly competition for new T-shirt design in an online community with over 600,000 registered members from all over the world. Every week up to ten new designs were selected for production from more than 800 submissions through an open polled by the community members (Fuchs and Schreier 2011, Lakhani and Panetta 2007). Innocentive, an Internet marketplace for call for open innovation, publishes R&D challenges of commercial companies and offers rewards up to $100,000 for solution. More than 170,000 scientists, engineers, and researchers from 175 countries registered as member in the 'solver' community. Up to 2009, 800 technical challenges were posted and 12,528 solutions were submitted. More than 300 chemistry, biology and engineering problems have been figured out by the community members and $3.9 million was awarded (Lakhani and Jeppesen 2007). Dell computer launched the Idea Storm initiative, where users from around the globe have been invited to suggest product improvements and new product ideas online. This initiative has resulted in more than 10,000 idea submissions (Poetz and Schreier 2012). In another case, Peugeot, the French car manufacturer, operates an online car design contest every two years with a prize 10,000 Euros since 2000. In the competition 2008, several hundred thousands of automobile funs from over 100 countries registered in the community and submitted 2,500 designs (Puah et al. 2011). Similar strategies were adopted by many companies across industries, including Adidas, BBC, BMW, Boeing, Ducati, and Muji (Berthon et al. 2007, Ogawa and Piller 2006; Piller and Walcher 2006; Sawhney et al. 2005).

Company cases reveal that in the new business environment crowdsourcing in online communities present many benefit to commercial companies. By moving the innovation activities from internal R&D labs to external online communities, companies potentially have the access to a big number of professionals, specialists, experts, as well as amateurs distributed around the world. This huge intelligent pool may greatly enforce companies' innovation capacity if it could be well exploited.

The case studies presented above however also illustrate a number of critical limitations of crowdsourcing. The crowdsourcing projects in most of the cases are small and based on individual work. The designs or solutions submitted are generally from separated individuals; not much, if not none, collaboration between community members happens in the problem solving and innovation process. This could be explained by the characteristics of both the demand and supply sides of crowdsourcing market. On the demand side, the crowdsourcing projects are often small in size in terms of workload required. They generally demand more for intelligence and creativity, and less for labor and time input, for example, the design of T-shirt, outline of concept car, etc, and therefore the need of collaboration is not exigent. The projects also generally involve only one special set of skills or knowledge, and not many cross-disciplinary problems. This also diminishes the need for collaboration.

On the supply side, online community members are geographically distributed and generally do not know each other. To build up the sense of trust and to recognize each others' capability, people need to interact and work together for a long time. However, the discrete individual crowdsourcing projects are not able to provide this ground. Moreover, crowdsourcing itself in most cases is an open contest for cash return. In online community setting where no authoritative or hierarchic structure exist, the tasks of organizing a team, managing the progress, coordinating members' activities, evaluating their contribution, sharing of rewards, etc. are much more difficult than in real world organizations. This significantly increases the transaction cost of the crowdsourcing biding of team work and makes the collaboration and teamwork uneconomic.

In the small sized and simple problem solving crowdsourcing projects, individual community members are able to work out the challenge on the base of personal intelligence and creativity. However, the incapability of collaboration and team work may present problems when companies searching online for solution or subcontract for large, cross-disciplinary, time- and labor-demanding projects. In commercial environment, the complex and large size projects generally require a number of people work together with division of labor and skills, and are often under time and budget restriction. This presents a challenge to adopt the community innovation model in commercial software development which is our interest.

Research approach

Action research

Action research is a joint effort of academic researchers and business managers in order to figure out some organizational problems whilst conducting an academic research to expand scientific knowledge.

Contrast to the classic social research such as quantitative survey oriented research or qualitative case studies where researchers focus on studying the phenomena but do not change anything in the context, action research aims at both making organizational change and studying the changing process which finally leads to knowledge creation. With the integrated double objectives and the ensued research design, action research was regarded as a good way to enhance the relevance of academic research without abandoning rigour (Iversen et al. 1994). Action research is conducted in the real organizational and business settings. It is generally initiated by a real business problem, then researchers and management practitioner collaboratively analyze and diagnose the problem with support of academic theories, and create some therapeutic solution. The solution will be practically implemented in the organization in order to change in the problematic situation. The process and effect of the change will be evaluated and studied to ensure learning. By so doing, the action research contributes both to the practical organizational concerns and to the body of knowledge of social science (Rapoport 1990).

The objective of the presented project is to develop a new business model which utilizes the technical force in online community to develop commercial software projects. The research was initiated as a doctoral research project in 2007. The researcher owned a middle-sized software company Juling Infotech Inc. in Shagnhai, China (referred as Huazisoft, the product brand of the company, in the following text). The other researcher is associate professor in innovation management and engages in studying software engineering community. The two researchers initiated the project as an action research with both business and academic objectives, and the double identity of the first researcher, as a management doctoral researcher and as a business manager, ensures an easy access to the target company.

In line with the double objectives, the research is conducted in the way of information system prototyping (Naumann and Jenkins 1982; Connell and Shafer 1989; Baskerville and Wood-Harper 1998). Prototyping as a paradigm of information system development originates from engineering system development where the designers provide the user a tentative system for experimental purposes, and then users' experiences and feedback are used to modify and improve the prototype, which is provided to user again for experimental use. These iterative processes of user experiment and design modification make designers better understand users' specific demand and the application environment and lead to quality design. IS research is highly empirical, practical and professional based and the combination of professional methods with academic rigor adds value to the quality of research outcome. In particular prototyping was regarded as an important approach of IS action research due to its inherent nature of iterative social interaction and learning for the purpose of organization development. As Baskerville and Wood-Harper (1998) summarized that prototyping moves the system design process into the user's multivariate social setting, permits highly interpretive assumptions about observation generating ample qualitative data, and involves an direct intervention by the designer in the user organizational context. During the development process, the researchers are conducting participatory observation in order to study the suitability of the designed system and its impact of change to the organization. These are typical features of action research.

The research setting

Huazisoft specializes in developing e-business solutions, ERP/CRM systems, and functional information systems for local market. In 2007 the company had around 50 employees. Huazisoft involves in OSS development as an enterprise member of Open Source China and contributed to many OSS projects. As part of its business portfolio, Huazisoft runs an online software engineering community www.Misuland.com with hundred-thousands of registered members. The community was solely a technical forum facilitating software engineers' advice seeking and technical problem solving without explicit direct business benefit.

Huazisoft found itself in a dilemma situation in managing its technical force. The years around millennium saw a boom in the market of customized software packages. Many companies invested to update their information system or to develop Internet based business applications. In responding to the demand, Huazisoft rapidly expanded its technical force. It was found however the market was quite in fluctuation. In some seasons the company needed to have development force double to its capacity in order to fulfill the development planning, while in some other seasons only 60-70% of the technical capacity was utilized. When demand went down, Huazisoft had to lay-off engineers while when new projects came, Huazisoft rushed to recruit engineers in market. The fluctuation made it difficult to build up a stable technical force and made the balance sheet tight. It also happened that some projects involved specific industrial knowledge beyond its accumulation, and Huazisoft had to

urgently search for specialists in the market.

As a middle-sized software company, maintaining a big group of engineers regardless market fluctuation is difficult. A potential solution is outsourcing. Huazisoft however found it difficult as the projects the company contracted were generally in small and middle size and with low profit margin. Outsourcing to third party makes the projects non-profitable. Huazisoft also tested the strategy to recruit engineers by short term project based contracts, while the practice was proved to be unsuccessful due to problems caused by high turnover, short integration period and insufficient training.

The researchers and Huazisoft management team analyzed the company's business problems, resources available, and possible strategies to figure out the problem. The open innovation model and successful practice of OSS development provided a possible direction for a solution. An idea was proposed that it might be feasible that Huazisoft utilizes the technical resources in its online community Misuland.com to develop commercial software. After rounds of discussion, the researchers and Huazisoft management team decided to develop a new business model as well as the associated supporting IS platform to actualize the idea. After an extensive investigation in academic literature and industrial professional expresses, it was confirmed that this possible model was novel without precedents in practice.

Although the OSS community development model and successful stories of crowdsourcing provided useful references, none of them could be directly transferred to the research setting. Huazisoft realized that this project was not to just develop an IS platform to facilitate its software project management, instead, it aimed at creating a new business model to replace the company' old model and software development practice. The company attached great importance to the collaboration with research teams and to the guidance from management theories. The collaboration was going very well in which researchers and Huazisoft management team both contribute to the iterative process of diagnosis, action planning, action taking, evaluation and learning.

The research process

This research adopted the classic evolutionary prototyping approach where the final solution is developed by an iterative process of user testing, evaluation, and further revision until the full functionality is well achieved. The research team firstly investigated and confirmed online community members' motivation to participate the commercial software projects and then specified the system requirement, the relationship between different actors, the model to actualize crowdsourcing and to make payment, and the software development platform. Huazisoft technical development team then actualized the system and provided it to the community. With 4 real-life software projects Huazisoft contracted, the system was put into practice on a trail base in Misuland community. In total 33 community members joined the 4 projects, used the platform, and developed the real-life commercial projects. During and after every project, community developers provided valuable feedback and suggestions continuously, more and more details of the system were put onto table, and improved the research team and Huazisoft management team's understanding of the new business model and software development method. The system was continuously revised and improved, and new version was provided for the next project prototyping cycle for further use, evaluation, and improvement (Figure 1)

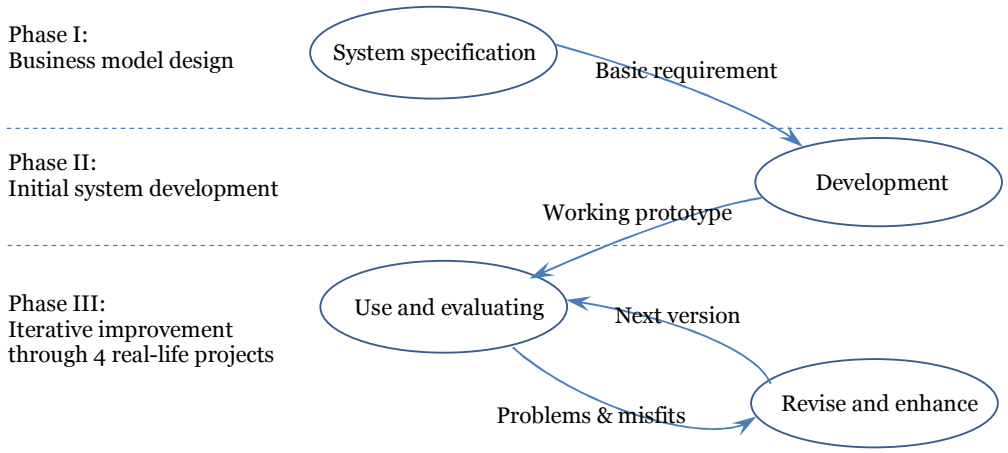

Figure 1: The research process [revised from the prototyping model of Naumann and Jenkins (1982)]

Phase I: Business model design

A business model describes how a company produces, delivers and sells its products or services in order to generate sustainable revenues. Design of business model requires a clear specification of the issues: the dominant value creation drivers, the activities to be performed, the linking and sequencing of the activities, and who will perform the activities (Zott and Amit 2009). In the presented research, to design the business model of community based software development, besides the issues related to the practice of customized software development, a number of new questions need to be explored:

1. Are the members of online software engineering communities willing to join commercial projects?
2. How to motivate and recruit the community developers?
3. How to organize and coordinate the distributed community developers to work together?
4. How to monitor and control project quality, cost and schedule in community environment?

25 active members of Misuland community were interviewed though telephone or instant massager in early 2008. Community members presented ideas that they joined the community in order to learn, to practice and improve their skill, to share, to make friends, to have fun, etc. When being asked whether they were interested in joining some software projects in the community and getting extra payment, almost all interviewees shown strong interest and gave very positive feedback. The typical answer was like "no problem, I am happy to join the project in my off-duty time", "it would be good if there are some opportunities in Misuland to make some extra money", etc.

An online free call questionnaire survey was conducted in Misuland after the interview and 997 community members answered the questionnaire. 88.9% respondents reported that they were willing to join the paid software development project. 13.8% respondents reported that they could spend over 4 hours per day, 25.2% could spend 3-4 hours, and 47.9% could spend 2-3 hours, the other 13.1% could spend less than 2 hours. 56.3% reported that it is no problem to work together with community members who never meet in reality to development software. 43.7% presented concerns that it may be difficult to coordinate due to people's different technical thinking, personal disposition, etc. Despite the possible coverage bias associated with the sampling method[1], the interviews and survey gave researchers positive support that recruiting community members to develop commercial software is feasible.

Referring the successful practice of crowdsourcing communities (Fuchs and Schreier 2011; Lakhani and Jeppesen 2007; Lakhani and Panetta 2007; Poetz and Schreier 2012; Puah et al. 2011; Berthon et al. 2007; Ogawa and Piller 2006; Piller and Walcher 2006; Sawhney et al. 2005), the research team decided to adopt crowdsourcing to recruit and motivate online software engineers on a project base.

[1] The interviewees were active community members and may have different characteristics from ordinary members. The online survey was made by open call in the community. It is likely the people who filled in the questionnaire were those more interested in online software development than those who were not interested. This might create coverage bias.

The related business model and realization mechanism were designed including the generic model, crowdsourcing rules, virtual currency, project lifecycle, team construction approach, quality control approaches, etc.

Phase II: Initial system development

Following the design of the new business model, an IS system was designed and developed as the platform to actualize the new way of software project development. It consists of three subsystems: community, project development management, and instant messager. The community subsystem is used to perform community membership management, crowdsourcing management, payment management, component library, technique forum, etc. Project management subsystem includes functions of project team management, process control, quality control, code management, project file management, deliverable testing and project closure, etc. The subsystem instant messager acts as the key communication tools among project team members during the whole process of crowdsourcing and project development, and between Huazisoft internal team and community developers. It is highly integrated with the community and project management subsystem.

Phase III: Prototyping with real-life projects

After the system was developed, it was put into practice in Misuland community on a trial base with real-life projects. The primary purpose is to verify the soundness of the design ideas and to ensure that system can support community development in different conditions. 4 projects were conducted sequentially with the system including an online customer service system, an operation monitoring system, an E-commerce system and an MIS system (Table 1). The systems are typical projects of Huazisoft in terms of technical domain and project size.

Learning and improvement

In every project, Huazisoft management team had a close look in every step of crowdsourcing and development process. The developers were asked to report any inconvenience or problem they found in the CBCD model and supporting platform during the project. As part of project closure process, the developers were requested to fill in a detailed form to evaluate the system. After reviewing the feedback, Huazisoft management team had telephone interviews with every developer to further gather information. Then the research team, the management team and the internal technical staff involved in the project had meetings to collectively evaluate the prototyping process, review the feedback, discuss the problem in current version, and finally make decision about what modification should be done to improve the system. After the system was modified, it was put into practice in the next prototyping cycle.

The original CBCD supporting platform was development by Huazisoft technical team with two-year efforts. When it was put into practice, however, it was found incontinent by community developers from time to time. For example, in the initial CBCD model, community developers were expected to join the project from early stage of system design. This decision was made on the base of the results of the above mentioned interviews, in which many community developers expressed strong interest and confidence in joining in the system design work. However, in the first prototyping project, an online customer service system, it was found very difficult for the community members to find common available time slots to have online meetings. Even when meetings were set up, the online discussion was not efficient in terms of interactive decision making. This caused delay in milestone delivery. After having phone call with the developers one by one, the research team decided that the company internal technical team needed to carry out the system design and modulation tasks so as to reduce the reliance on community developers' collective decision making. The related elements in the CBCD model and the supporting platform were modified accordingly.

In the same project, as another example of learning and improvement, one community developer requested to use one of his own pre-developed components, which, as he claimed, were developed by himself before and he had the full copyright, to fulfill a functionality of the software. This was against the original thoughts about component library, which, at the beginning, only consisted of components developed internally by Huazisoft and it was expected new components would be added when community developers accomplish the crowdsourcing projects by CBCD model. Permission to use community developers' old components definitely would improve efficiency, but what is the risk and what is its implication to the copyright of the software? After intensive discussion between the research team and company managers, as well as with clients, it was finally decided to allow developers to use their own components. In order to balance the risk and the impact to software

copyright, rules and regulations were modified or re-designed, including the component library regulations, mechanism of component pricing, declaration of originality and copyright, agreement of component leasing, contract template with clients etc. This project was developed much faster than Huazisoft technical team originally expected as a result, partially, of the inclusion of a number of key pre-developed components from the community developers.

Every project in the prototyping process provided valuable feedbacks to the research team, as shown in table 1, and improved the CBCD model step by step from both business and technical perspectives. Many lessons were experience based. For example, the best cycle of milestone delivery in CBCD model was found to be 3 to 5 days, which is much shorter than that of in-house software development, and is adjustable according to project size, technical complexity, number of community developers involved, etc. The improvement of the model was made in a spiral way in which many elements were revised again and again. For example, community developers were excluded from early stage system design due to the lessons learned from the first project, while later, in the fourth project, an MIS system, the door involving community developers in system design was re-open but in a different way.

In this project, at the very beginning when Huazisoft technical team discussed the system specification and technical direction with the client, the team members a doubt about the architecture of the solution. An engineer posted the question in Misuland forum seeking for advice. More than 20 community members replied the call and posted more than 60 replies. One community member, net nickname Flying_cutter, a well-known technical expert in the community, shown strong interest and finally figured the problem. He also gave many suggestions to Huazisoft technical team to optimize the preliminary design. After knowing the community member was located locally in Shanghai and had plenty of free time, the research team decided to make an experiment whether the local community developers could be involved in customer demand specification and system design. The research team communicated this idea with Flying_cutter and got positive response. Flying_cutter recommended a friend of him, another active member of Misuland community, net nickname Tomato_sweetie, to join the project. These two community developers participated the project from system specification stage, visited the client with project manager of Huazisoft, and accomplished system design and planning. Flying_cutter was nominated as project manager then and 3 other community developers were selected through crowdsourcing to join the project for system implementation, coding and testing. The project was very successful with both shorter project lifecycle, i.e. 358 man-days against 500 man-days originally estimated by Huazisoft technical team, and excellent product quality. During and after this project, CBCD model was modified and new elements were added to the platform to support the activities when community developers are involved as project manager and in early stage of system design.

As presented briefly in table 1, every project in the prototyping cycles provided valuable feedbacks to the research team from different perspectives, and contributed the improvement of the CBCD model and the supporting platform. As a matter of fact, the model was continuously put into practice with commercial projects after the prototyping cycles. Up to December 2012 over 30 projects were developed with the CBCD model. The 4 projects presented in table 1 gave the most important lessons leading to significant improvement in the model. In the later projects, problems and misfits were also identified, while in less frequency and smaller scope, and leaded to some minor modifications of the system which was operated stably.

Table 1: The prototyping process with real-life projects

Prototyping cycle	Project	Project size and community developers involved	Project arrangement	Lessons learned	Problems identified	Modification made to the model/system
1	An online customer service system	- 3 community developers accomplished the project with 125 man-days. (Huazisoft initially estimated the project to be 400 man-days)	- Project manager was internal developer. - System specification and modularization design were initially expected to be done by community developers, but later were done by internal team. - Functional modules were developed by community developers.	- Project can greatly benefit from community developers' devised technical capacity and resources. - Pre-developed components should be accepted if copyright issue could be settled.	- Team building and development coordination among developers are more difficult than originally understood. - The system over-expected community developers to carry out system design.	- System design task was adjusted to be accomplished by internal team. - Coordination subsystem was redesigned. - Process monitor function was redesigned. - Component library and associated agreements and rules were redesigned.
2	An operation monitoring system	- 2 community developers accomplished the project with 103 man-days.	- Project manager was internal developer. - System specification and modularization design were done by internal team. - Functional modules were developed by community developers.	- The core technique of the system was beyond the capacity of Huazisoft. Community development expands the project domain of the company. - Instead of contracting every developer, it is better to only contract the leading developer in some case.	- Communication between developers and internal team was not well supported. - Submission of milestone deliverables was not well regulated. - Difficult to define the role and responsibility of every developer who participated in developing a module/project.	- Coordination and communication subsystem were enforced. - Process monitor function were enforced by clear definition of stage deliverables in the planning process. - Encourage one leading developer to organize a team to contract a module/project. The leader takes responsibility to manage the team. No contract is signed with the team members.
3	An E-commerce system	- Community development failed at the middle of the project lifecycle due to the quit of team leader. - Huazisoft team spent around 200 man-days to accomplish the project.	- Project manager was internal developer. - System specification and modularization design were done by internal team. - Functional modules were developed by community developers.	- Risk of breach of crowdsourcing contract should not be under-estimated. - Unfinished deliverables of community developers are hardly useful. Better re-developed it completely.	- Backup mechanism for failure and breach of contract were not well prepared.	- Backup mechanism was re-designed. - Process control by internal team was enforced. - Buffer of time schedule was prolonged. - Internal backup resources should be available anytime.
4	An MIS system	- 5 community developers accomplished the project with 358 man-days. (Huazisoft initially estimated the project to be 500 man-days)	- System specification was done by internal team and a community developer together. - Project manager role was given to a community developer. - Modularization design and modules development were done by community developers.	- Community developers can be involved in the whole project lifecycle, not just implementation and testing stage as previously specified.	- The development management system lacks of mechanism facilitate the involvement of community developer as project manager and to control the associated the risk.	- Rule was set that only well acquainted community developers could be entrusted as project manager. - Function to support community developers' involvement in system specification and modularization was added. - Function to support community developers' role as project manager was added.

Crowdsourcing based community development model

The model and project lifecycle

The distinction between OSS and classic in-house development models raises three challenges: 1. how to systematically motivate and select software professionals from online community? 2. how to organize and facilitate the distributed software professionals to work together online? 3. how to ensure the quality of the software developed? A hybrid model (refers as crowdsourcing based community development model, CBCD) integrating the advantages of crowdsourcing, in-house and OSS development models was developed as following (Figure 2).

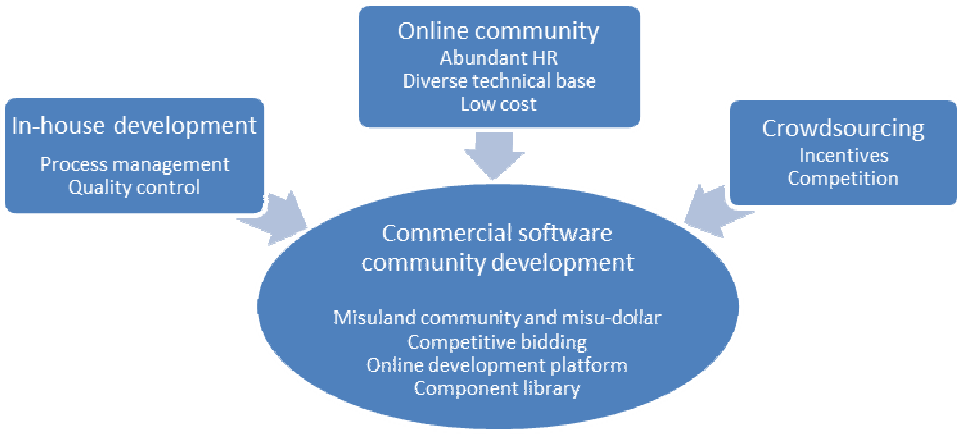

Figure 2: CBCD as an integration of crowdsourcing, OSS and in-house software development models

In the proposed CBCD model, the online software engineering community stands at the center of this model. Huazisoft had a software engineering technical forum, Misuland.com, in which over 200,000 software professionals were registered in 2007 when the research was initiated. Many members are very knowledgeable and skillful, and are real experts in their technical area.

The community development process starts from the Huazisoft's new initiation of a software project (Figure 3). The company project management team specifies the software demand with clients and discomposes the project into modules. The crowdsourcing process was initiated when the modules (sometimes the whole project when it is small) are put into a competitive bidding system in Misuland community. Responding to the call for bidding, community developers self-organize teams, submit tender schemes which specify the proposed technical solution, budget, timetable etc. The project management team valuates the tenders, selected the winner teams and issue contracts. Then the members of winner teams collaboratively develop the modules in the online platform. When the modules passed the test, they will be integrated into final software system with full documentation by the Huazisoft project team. After the software is checked and accepted by the client, payment will be made to the software professionals following the crowdsourcing contracts. The code packages created by community developers are reviewed by a team of Huazisoft, and then put into a software component library. The developers could choose to sell it or to lease it to the component library.

In order to support the community development model, Huazisoft developed a full set of platform, standards and protocols including project modularization and planning standard, crowdsourcing regulations, online software development platform, instant messenger, component interface standard, communication protocol, component trading, leasing and renting rules, etc. When design and develop the above elements, Huazisoft searched in the market and OSS communities, but did not find affordable, mature and reliable system or subsystems (with a few exceptions) which can satisfy the specific purpose of the CBCD model. Therefore Huazisoft designed and developed the big majority of the above elements. A

few elements, for example, online version management system, were developed based on OSS. Huazisoft also maintain an in-house development workforce in order to develop the project in case that community developers fail to satisfy client demands or unexpected contract breach happens.

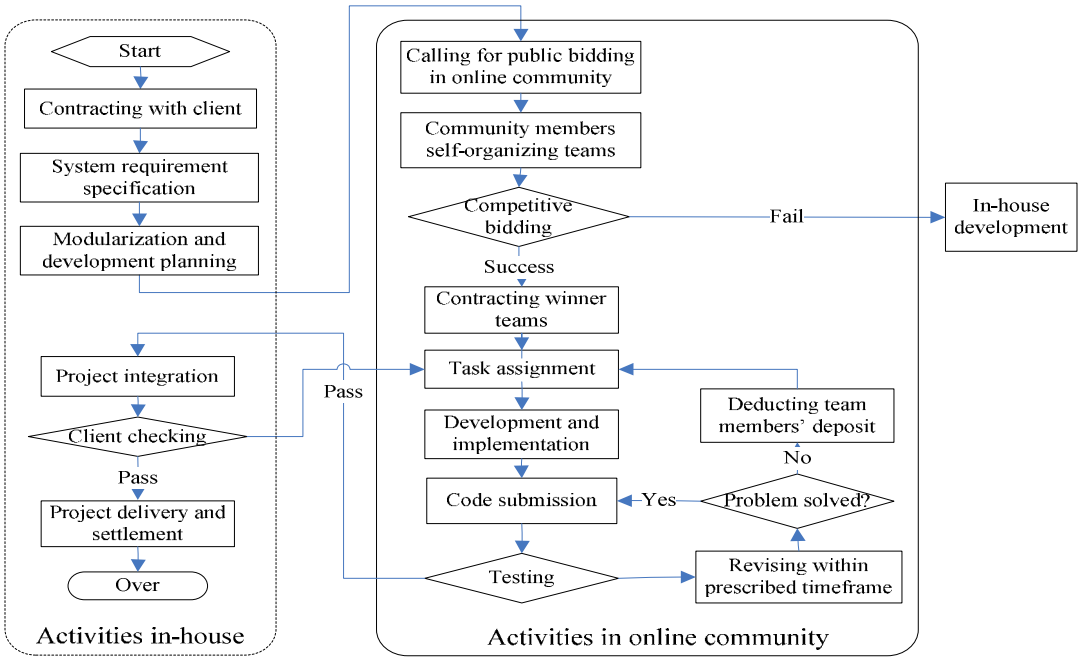

Figure 3: The project life cycle of crowdsourcing based community development model

In contrast to the conventional in-house software development which typically comprises of four broad phases including specification, analysis, design, and implementation, Misuland model keeps the core phases of specification, analysis and design in-house, and execute the implementation and test by virtual teams in the online community.

Supporting the distributed collaboration of self-organizing teams

The community developers are distributed geographically, have no experience of working together, and hardly see each other during the whole process of project development. It is a critical challenge how to systematically organize, coordinate and support the distributed collaboration of the community members. With efforts of the action research and a long process of trial and error over two years, Huazisoft developed a set of practice and technical standards to support the crowdsourcing and distributed development in online community.

Team construction

The registration information of community members in the technical forum is simple and minimized where people can use their email as account ID. When the community member wants to join the software development project, they must provide real identity, address, bank account and other personal information to go through a second round registration. The real identity will be used in the crowdsourcing contracts when they win the bidding.

Everybody in Misuland community has the right to answer the call for bidding by organizing a team or alone. In most of the cases, team is appreciated when the project involves multiple technical capacities and requires time delivery. In order to facilitate team construction, a community member technique evaluation system was implemented with information from two sources. The first element comes from Misuland forum, a software engineering community, where there is a point reward system, similar to the

expert account system in China Software Developer Net (CSDN) community (Assimakopoulos and Yan 2008). Community members' breakdown of the points presents the members' detailed technical strength in different technical areas. The second, more importantly, is the post-project technical evaluation system, similar to the reputation system of Ebay. After a project is accomplished, Huazisoft project management team evaluates the developers' technical performance in a set of quantitative scales. This information is useful when the community members want to organize team to answer a call for bidding.

Actually, with long 'living' experience in Misuland community, many members are familiar with each other online. Members with high points accumulated in the problem solving forum are well-respected and considered as highly qualified expects. When a member decides to set up a team to answer a call for bidding, s/he generally tries to recruit the experts with good technical reputation and the members s/he has good 'virtual' personal relationship with. In many cases, after successful accomplishment of a project, the members of the virtual team decide to continue to work together and tender for the next project.

Online project development platform

In order to support the collaborative development of the virtual team, an online platform was developed, which consists of three integrated systems (Figure 4). 1. Instant messager system: it is a package similar with Microsoft MSN while with more special functions to support distributed software development. It is used by the community members to communicate with each other during the whole project life cycle from building up team, creating tendering proposal, task assignment, implementation and test, to final submission. The distributed developers use it not only for communication and technical discussion, but also share documents and synchronize work progress. 2. Project management system: it is the network version of Huazisoft's in-house software project management system in line with CMM2 level of quality control. The Huazisoft project management team use system to organize, coordinate, monitor and integrate the development work of community developers. The final payment and component pickup is also conducted in the system. 3. Version management system: it allows community developers to collectively manage and synchronize codes and documents developed.

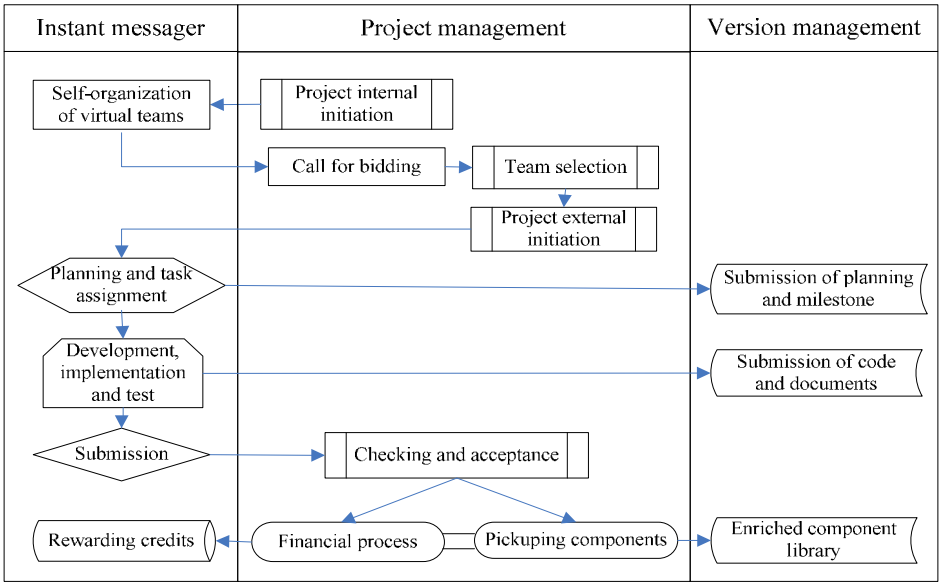

Figure 4: The online project development platform

Interaction between in-house workforce and community developers

During the course of a project, most of the administrative communication is made between the Huazisoft software management team and the virtual team leader, as illuminated by figure 5. The technical communication is intensively made among the team members with the instant messager. All the technical communication, different version of the code and documentation are monitored by the project

management team in the online project development platform.

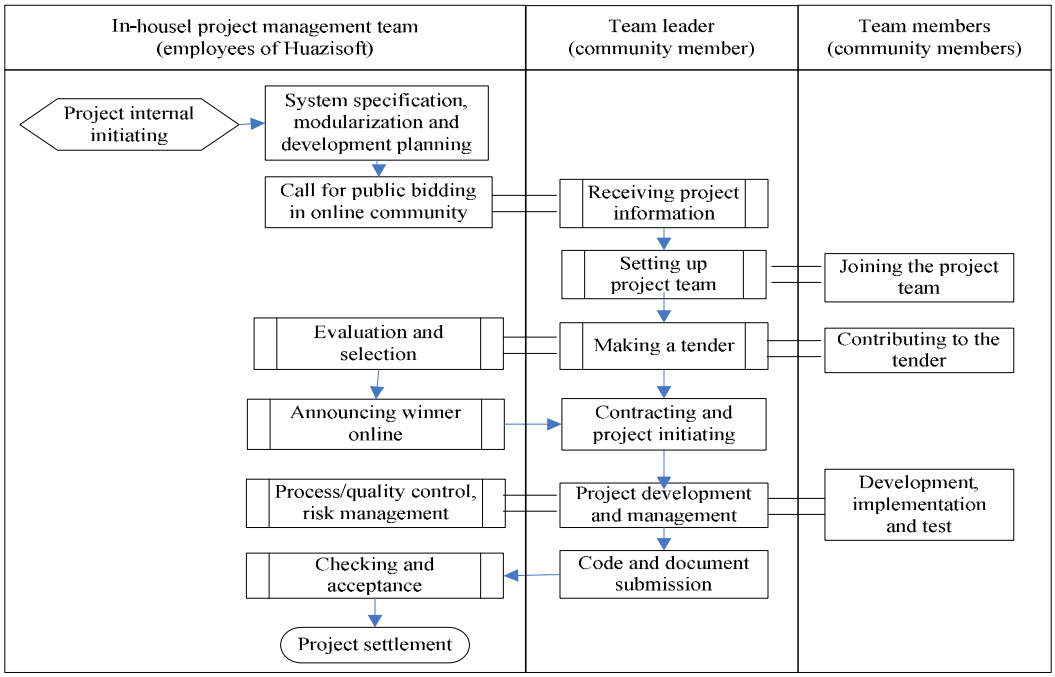

Figure 5: Different roles of in-house workforce and community developers

Quality control

Community developers' collaboration in the crowdsourcing projects is often provisional, and ongoing through distance communication and asynchronous coordination. This brings risk in product quality. On the top of CMM2 software process improvement practice integrated in the development management process and online platform, the research team developed a set of rules to regulate the community development.

- Although winners of the crowdsourcing bidding are distributed virtual teams, the contract with team leader is formal and has legal effect. The team leader is liable for breach of contract in which the requirements on function, quality, timetable, etc. are clearly specified.
- The system specification, detailed demand description, development planning and detailed requirements are released in the community simultaneously with the call for bidding. Only the submissions complying with the requirements will be accepted for test and check. The submissions must be well modularized, standardized and reusable.
- An amount of Misu-dollar (currency in Misuland community) is frozen as deposit from the accounts of winner team members after the contract is signed.
- The submission of codes and documents must be in pure text format for safety concerns. Submission containing unsafe or vicious codes will be penalized.

In addition, Huazisoft also has a 'fire-fighting' mechanism in case of failure of community development. For example, an amount of time is always reserved in the development planning and crowdsourcing contract; submission of codes and documents is made in a stepwise way so that the development progress is well-monitored and controlled; an in-house 'fire-fighting' team is always available in case of emergency of unsatisfied submission or breach of contract.

Component library

Every time after a project is accomplished, some well-developed modules or functionalities are collected

into Misuland component library. The developer or owner of the component set up the price to sell the copyright to Huazisoft, or in most cases, to store it in the library for leasing it to other software projects. Huazisoft developed the standard of the component library, protocol of interface, criteria of collection, declaration of copyright, pricing guide, component category, etc. to regulate the building up and use of the library. There are over 1000 components in the library up to the end of 2012. Among them, about 100 are owned by Huazisoft, 200 were collected from the 30 projects developed by CBCD model, and over 700 were uploaded by Misuland community members (including company members).

Misuland community developers use the component library intensively. Among the 30 software projects accomplished by CBCD model, 21 used components, and accumulatively used 62 times. One project used up to 8 components. The most frequently adopted components include printing management module, E-commerce customer management module, virtual printing module, community membership management module, etc. Adopting the well pre-developed components significantly improves the product quality and development efficiency. Two projects in the prototyping cycles (see Table 1) saved up to 69% and 28% of the man-days originally estimated by Huazisoft technical team, as a result, partially, of the intensive adoption of pre-developed components contributed by community developers.

Copyright issue

Huazisoft has very flexible proposition with copyright issue and makes up agreement with clients based on their special demand during the project negation and contracting stage. In some projects in which clients insist to have full copyright with source code, component and open source elements are excluded during the CBCD development. This however happened rarely in Huazisoft's business operation, i.e. only 5 out of the 30 projects. The key reason is cost and speed of the development. The contracting price of a project permitting the involvement of open source elements and pre-developed components may be only 30-70% of the same project with full copyright. Most clients of Huazisoft are small-middle companies which care more about cost and time of delivery than exclusive ownership of copyright. They generally expect cheap contracting price, fast completion and guaranteed maintenance, while give up owning copyright. This makes it possible the extensive use of open source and component in most projects developed by the CBCD model.

On the other hand, Huazisoft is also cautious in copyright issue when managing the CBCD development. Community developers selected by crowdsourcing are required to sign a declaration to claim that all the codes they submit are original, or abidance by the PGL/BSD/Apache agreement if OSS components are included. By so doing, Huazisoft makes itself safe by shifting the risk of infringement of copyright to community developers.

Discussion, Conclusion and Limitation

In this research, a crowdsourcing based community software development model was designed, established, tested and put into practice in an Internet community by means of an action research. The CBCD model developed can be regarded as a hybrid of OSS and in-house development models where crowdsourcing mechanism serves as a bridge connecting the two parts. The model was proved to be successful as a new business model and a novel way of organizing software development.

From business perspective, CBCD model integrates the online software engineering communities into the value chain of software companies. Comparing to the traditional in-house software development practice which mainly utilizes the resources within company, CBCD model makes it possible for companies to exploit the abundant resources in online software engineering communities where it exists a huge number of software talents with diverse technical specialties and know-how. The competition-based crowdsourcing practice enables companies to pick-up the most creative solutions and select the elite engineers to carry out the software projects. It also provides companies flexibility in human resources management where long term employment contracts are replaced by cost-saving project-based contracts. Serving as a huge accessible intelligent pool, online communities significantly enhance software companies' technical capability, expand their business range, and enables them to seize new business opportunities and projects beyond the companies' current technique reserves. All these provide competitive advantages to software companies.

From technical perspective, what makes CBCD model interesting is its successful combination of the advantages of OSS and in-house development models. CBCD model tries to change the voluntary nature of the OSS development in which a big majority of OSS projects were observed as inactive/hibernating (Madey et al. 2004), progressing very slowly (Capiluppi et al. 2003), and never release any version for distribution even after years (Healy and Schussman 2003). It imports the technical and human resources from online communities whilst uses the management mechanism of in-house software development model to control budget, quality, schedule and other project risks. Monetary reward is used as an incentive to recruit community developers and makes them not only commit to contribute to the software projects, but also accept the tough discipline and management control, which generally does not exist in the community environment. The supporting IS platform, as an indispensable part of the model, supports the community developers' communication, sharing of resources, and interactive remote collaboration. The involvement of pre-developed components and the flexible proposition in terms of copyright, which is mainly based on clients' choice, improves product quality and the efficiency of the development. Table 2 presents a detailed comparison between the proposed CBCD model and OSS and in-house software development models.

Table 2: Comparison between OSS, in-house and CBCD models

	OSS	In-house	CBCD
Product	Open source software	Proprietary software	Proprietary software
Motivation of developers	Altruism, social and indirect return	Value for money	Value for money
Initialization of a project	Community members' "an itch worth scratching"	Commercial initiative from clients	Commercial initiative from clients
System analysis	Common agreed understanding of the application in the community	User-participated requirement specification, done by employees	User participated requirement specification, done by employees /community members
Module design	High modularization, done by core community members	High/Middle modularization, done by employees	High modularization, done by employees /community members
System implementation and test	Done by community members	Done by employees	Done by community members
Team building	Self-organizing in community	Employment contract and hierarchical management	Self-organizing in community
Quality control	By community leader and peer evaluation	By employees	By employees
Budget and schedule control	Self-organizing, loose control, low efficiency	Full management control, high efficiency	Full management control, high efficiency
Cost	Very low, if not none	High	Low
Rewards to developers	Intangible, indirect non-monetary rewards	Salary	Monetary rewards, generally second income for the developers

CBCD model is also a successful attempt to carry out complex commercial projects by crowdsourcing in online communities. Due to the constraint of internet-based remote communication, crowdsourcing projects in online communities are generally small in size and based on individual work. Individual creativity is more emphasized than collective collaboration. CBCD model proves that crowdsourcing complex projects with team work is possible. Several factors are found important: First, the big size complex project needs to be appropriately decomposed into relatively independent packages, i.e. modules, which can be crowdsourced separately. The company carrying out the crowdsourcing activity needs to take the responsibility of decomposition and modularization with their specific domain knowledge. Second, a sound supporting tools, i.e. the supporting IS platform in CBCD model, needs to be designed, developed and provided to facilitate the remote collaboration among community members. In complex technical projects it generally requires intensive and durable interaction among the participants, which is naturally difficult in online communities. The crowdsourcing service provider needs to design and develop an appropriate mechanism, from both business and technical perspectives, to facilitate the remote collaboration and to manage the risks associated with the project, for example, budget, quality, schedule,

and copyright, etc.

The CBCD model can possibly be extended to other knowledge intensive industries, where intelligent input and creativity and are more required in the production or service process than onsite presence of the knowledge workers. The crowdsourcing mechanism and supporting platform make it possible to recruit experts distributed geographically in online professional communities and enable them to work together. Architecture design companies, for example, could have a similar system to recruit the best architects around the world to work on a single project. In health care service, as another example, clinics could recruit experienced doctors in online communities to collectively diagnose the incurable diseases. The modern communication tools facilitated by instant messager, video conference, instant sharing of equipment output, even virtual reality system, etc. can help the distributed experts to work together.

As a new method of software development, CBCD model has some potential weakness and risks which need special attentions. The first issue is related to copyright. Currently CBCD model requires community developers to sign a set of agreements and declarations to claim the originality of the modules, documents and components they submit, and to claim the work is done in their off-duty time if they are employees of other organizations. In the business environment in China where the legal system is relatively loose, this treatment works. It is not sure however this treatment is enough to avoid the legal risk in other business environment. Second, CBCD model was developed and tested in Misuland software engineering communities, where community members show strong interest in participating commercial projects and getting a second income. This may not be true in other professional communities and in other cultural environment. It may happen that in some professional communities dominated by free sharing and altruistic ideology, for example, pure OSS communities, the crowdsourcing mechanism may not work. Third, CBCD model is tested with small-middle sized customized software projects in this action research, which are generally simple in design and technique, with workload from several hundreds to two thousands man-days. For projects with big size, complicated design, or involving advanced technology, CBCD model may not be able to handle. All these require further research in the future.

References:

Assimakopoulos, D., and Yan, J. 2006. "Sources of Knowledge Acquisition for Chinese Software Engineers", *R&D Management* (36:1), pp. 97-106.

Assimakopoulos, D., and Yan, J. 2008. "An innovative model of a computer-mediated professional community: China software developer net", *International Journal of Technology Managem*ent (43: 1), pp 238-251.

Baskerville, R.L., and Wood-Harper, A.T. 1998. "Diversity in information systems action research methods", *European Journal of Information Systems* (7), pp. 90-107.

Berthon, P.R., Pitt, L.F., McCarthy, I., and Kates, S.M. 2007. "When customers get clever: Managerial approaches to dealing with creative consumers", *Business Horizons* (50:1), pp. 39–47.

Brown, J.S., and Duguid, P. 2000. "Knowledge and organization: a social-practice perspective", *Organization Science* (12:2), pp. 198-213.

Chesbrough, H.W. 2003. *Open Innovation: The New Imperative for Creating and Profiting from Technology,* Harvard Business School Press, Boston.

Connell, J., and Shafer, L. 1989. *Structured Rapid Prototyping: An Evolutionary Approach to Software Development,* Yourdon, Englewood Cliffs.

Franke, N., and Shah, S. 2003. "How communities support innovative activities: an exploration of assistance and sharing among end-users", *Research Policy* (32:1), pp. 157-179.

Franz, R., and Wolkinger, T. 2003. "Customer integration with virtual communities", in the proceeding of 37th Hawaii International Conference on System Science, Hawaii.

Fuchs, C., and M. Schreier. 2011. "Customer empowerment in new product development", *Journal of Product Innovation Management* (28:1), pp. 17–32.

Fuller, J., Bartl, M., Ernst, H., and Muhlbacher, H. 2006. "Community Based Innovation: How to Integrate Members of Online communities into New Product Development", *Electron Commerce Research* (6), pp. 57-73.

Gacek, C., and Arief, B. 2004 "The many meanings of open source", *IEEE Software* (21:1), pp. 34-40.
German, D.M. 2004. "The GNOME Project: A Case Study of Open Source, Global Software Development," *Software Process: Improvement and Practice* (8:4), pp. 201-215.
Hara, N. 2000. Social construction of knowledge in professional communities of practice: tales in courtrooms, Unpublished doctoral dissertation, Bloomington: Indiana University.
Healy, K., and Schussman, A. 2003. "The ecology of open source software", Conference Papers: American Sociological Association, Atlanta Georgia, pp. 1-24.
Henkel, J. 2006. "Selective revealing in open innovation processes: the case of embedded Linux", *Research Policy* (35), pp. 953–969.
Iversen, J.H., Mathiassen, L., and Nielsen, P.A. 2004. "Managing Risk In Software Process Improvement: An Action Research Approach," *MIS Quarterly* (28:3), pp. 395-433.
Koch, S., and Schneider, G. 2002. "Effort, co-operation and co-ordination in an F/OSS software project: GNOME", *Information Systems Journal* (12:1), pp. 27–42.
Koku, E., and Wellman, B. 2002. "Scholarly Networks as Learning Communities: The Case of Technet", in Barab, B., Kling, R. and Gray, J., *Building Online Communities in the Service of Learning* (eds), Cambridge University Press.
Krishnamurthy, S. 2002. "Cave or Community? An Empirical Examination of 100 Mature F/OSS Projects", First Monday (76), [online], http://firstmonday.org/issues/issue7_6/krishnamurthy/index.html.
Lakhani, K.R., and Jeppesen, L.B. 2007. "Getting Unusual Suspects to Solve R&D Puzzles", *Harvard Business Review* (85:5), pp. 30-32.
Lakhani, K.R., and Panetta, J. 2007. "The Principles of Distributed Innovation", *Innovations: Technology, Governance, Globalization* (2:3), pp. 97-112.
Lakhani, K.R., and von Hippel, E. 2003. "How Open Source Software Works: "Free" User-to-User Assistance", *Research Policy* (32:6), pp. 923–943.
Lee, G.K., and Cole, R. 2003. "From a Firm-based to a Community-based Model of Knowledge Creation: The case of the Linux Kernel Development", *Organization Science* (14:6), pp. 633-649.
Moon, J.Y., and Sproull, L. 2000. "Essence of distributed work: the case of Linux kernel", *First Monday* (5), [online], http://firstmonday.org/issues/issue5_11/moon.
Naumann, J.D., and Jenkins, A.M. 1982. "Prototyping: The New Paradigm for Systems Development," *MIS Quarterly* (6:3), pp. 29-44.
Ogawa, S., and F. T. Piller. 2006. "Collective customer commitment: Reducing the risks of new product development", *MIT Sloan Management Review* (47,2), pp. 65–72.
O'Mahony, S., and Ferraro, F. 2007. "Managing the boundary of an open project". *Academy of Management Journal* (50), pp. 1079–1106.
Piller, F. T., and D. Walcher. 2006. Toolkits for idea competitions: A novel method to integrate users in new product development, *R&D Management* (36:3), pp. 307–318.
Poetz, M.K., and Schreier, M. 2012. "The Value of Crowdsourcing: Can Users Really Compete with Professionals in Generating New Product Ideas?", *Journal of Product Innovation management* (29:2), pp. 245–256
Puah, C., Bakar, A.Z.A., and Ching, C.W. 2011. "Strategies for community based crowdsourcing", in the Proceeding of International Conference on Research and Innovation in Information Systems (ICRIIS 2011), pp.1-4.
Rappaport, J. 1990. "Research methods and the empowerment social agenda", in *Researching community psychology: Issues of theory and methods*, Tolan P., Key C., Chertok F., and Jason L. (eds), Washington DC: American Psychological Association, pp. 51–63.
Raymond, E.S. 1999. *The Cathedral and The Bazaar*, O'Reilly, Sebastopol, CA.
Sawhney, M., and Prandelli, E. 2000. "Communities of creation: Managing distributed innovation in turbulent markets", *California Management Review* (42:4), pp. 24-54.
Sawhney, M., Verona, G., and Prandelli, E. 2005. Collaborating to create: The Internet as a platform for customer engagement in product innovation, *Journal of Interactive Marketing* (19), pp. 4–17.
Schubert, P., and Ginsburg, M. 2000. "Virtual communities of transaction: the role of personalization in electronic commerce", *Electronic Markets* (10, 1), pp.45-55.
Von Hippel, E. 1998. "Economics of Product Development by Users: The Impact of "Sticky" Local Information", *Management Science* (44:5), pp. 629-644.
Von Hippel, E., and Katz, R. 2002. "Shifting Innovation to Users via Toolkits", *Management Science* (48:7), pp. 1-13.

Von Hippel, E., and von Krogh, G. 2003. "Open source software and the private-collective innovation model: Issues for organization science", *Organization Science* (14:2), pp. 209-223.
Wasko, M., and Faraj, S. 2000. "It is what one does: why people participate and help others in electronic communities of practice", *Journal of strategic information systems* (9:2-3), pp. 155-173.