J. Software Engineering & Applications, 2010, 3, 1131-1140
doi:10.4236/jsea.2010.312132 Published Online December 2010 (http://www.scirp.org/journal/jsea)

Empirical Research on Critical Success Factors of Agile Software Process Improvement*

Jiangping Wan[1,2], Ruoting Wang[1]

[1]School of Business Administration, South China University of Technology, Guangzhou, China; [2]Institute of Emerging Industrialization Development, South China University of Technology, Guangzhou, China.
Email: scutwjp@126.com, mawangrt@gmail.com

Received October 29th, 2010; revised November 20th, 2010; accepted November 29th, 2010.

ABSTRACT

In this paper, we discuss agile software process improvement in P company with their description of process management in current level and analysis of problems, design the P Company success factors model in organizational culture, systems, products, customers, markets, leadership, technology and other key dimensions, which is verified through questionnaire in P company. In the end, we apply knowledge creation theory to analyze the open source software community with successful application of the typical agile software method, propose ten principles of know- ledge creation in open source software community: *Self-organizing, Code sharing, Adaptation, Usability, Sustention, Talent, Interaction, Collaboration, Happiness, and Democracy.*

Keywords: *Agile Methodology, Software Process Improvement, Critical Success Factor, Knowledge Creation, Open Source Software Community*

1. Introduction

Agile software Development encourages the formation of collaborative and self-organization teams that will have a huge competitive advantage over those who hold the view that a software-development organization is nothing more than a pile of twisty little people all alike. A gelled software team is the most powerful software development force there is. The Agile Manifesto is in the following: 1) Individuals and interactions over processes and tools; 2) Working software over comprehensive documentation; 3) Customer collaboration over contract negotiation; 4) Responding to change over following a plan. Twelve principles underlie the Agile Manifesto, including 1) Our highest priority is to satisfy the customer through early and continuous delivery of valuable software. 2) Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage. 3) Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. 4) Business people and developers must work together daily throughout the project. 5) Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. 6) The most efficient and effective method of conveying information to and within a development team is face-to- face conversation. 7) Working software is the primary measure of progress. 8) Agile processes promote sustainable development. 9) The sponsors, developers, and users should be able to maintain a constant pace indefinitely. 10) Continuous attention to technical excellence and good design enhances agility. 11) Simplicity. 12) The best architectures, requirements, and designs emerge from self-organizing teams [1].

P company is a multi-business company, and its main business is telecommunications services of Hong Kong. The P company referred in this paper is software development department of IT business division in P company. The staff in this department are more than 600 people, distributed in Hong Kong, Guangzhou, Beijing, Shanghai and other places. Department leadership attaches great importance to software process improvement, and the company has received ISO9001 and CMMI-5 certification. The market situation has changed greatly after financial crisis, P company has to adjust their software development process to meet the challenges of the

*This research was supported by Key Project of Guangdong Province Education Office (06JDXM63002), NSF of China (70471091), and QualiPSo (IST- FP6-IP-034763).

coming business environment. As the agile development method is efficient and simple, P company tries to use it in small area of some projects.

This paper is organized as follows: the first part is software process analysis before implementation of agile method, which contains process management and organizational culture; then the second part is to design and verify the model of critical success factors in P company, proposing ten principles with the analysis of agile methods typical of the knowledge creation, based on the results of the increasingly popular open-source software development; the last part is the conclusion.

2. Analysis on the Process of P Corporation

2.1. Analysis of Process Management

In order to meet the rating requirements of CMMI, P company design a set of quality management processes, including quality assurance program, process quality assurance, product quality assurance and evaluation, product testing, defect management and other aspects.

P company's process management tool is mainly used in the Gantt Chart. First, Project manager and core team members (each team leader) make the task decomposition, usually at the third level of decomposition to the WBS. Team leaders should complete a progress report on the situation to the project manager every week. Forced to survive pressure many employees often work overtime for free. When many low-level project staff have accumulated some project experience, often choose to leave the company and find another job.

P company usually send one or two senior analyst or business analyst to IT department of customer, and the analysts are in the charge of building Workshop with senior analyst from IT department of customer and communication between project delivery team and users. The workshop plays an important role: they collect and collate customer first-hand information, submit to the project delivery team to do demand analysis; they solve the problems from the demand analysis in delivery team; they review the requirements specifications made by delivery team; As P company usually uses JRP [2] method in the process of demand analysis, Work Shop is also responsible for organizing and coordinating the JRP meeting, including the development agenda, chairing the meeting, summary report, etc. If certain information has been misinterpreted or mishandled by some individuals of Workshop, the entire delivery team will be greatly affected.

Software development is knowledge-intensive work, tacit knowledge is more important than explicit knowledge in the development process [3]. The main communication channel of tacit knowledge in P company is that the staff familiar with each other to communicate privately. Of course, this mode of transmission is very inefficient.

2.2. Analysis of Organizational Culture

Agile software process improvement is not only a simple process, but also is related to some factors such as organizational culture [4]. In this paper we analysis some factors that are related with agile process improvement. 1). The prevalence of overtime culture. Overtime work is not suggested in the agile methodology, and is contrary to the principle "offer the employee sustainable development". 2) the culture of low-level trust. The top leaders in the company doubt the staff's work capacity and professionalism, always suspect that the staff can not get the job done because they don't have the right attitude. 3) Lack of the spirit of mutual cooperation. P company pays little attention to the cultivation of team spirit, and seldom organize activities about team building. However, once there is something wrong with the project, it's not a good thing for each member.

In all, the prominent phenomenon in P company's software and organizational culture are: tedious process, too much documents, formal assessment, serious overtime work, non-smooth knowledge communication channels, lack of trust between top leader and employee and spirit of mutual cooperation.

3. Design of Critical Success Factor Model of P Corporation

3.1. Analysis of Critical Success Factors

3.1.1. Support from Top Leaders

Paulk, Mark C states that if there is no support from top leaders, software process improvement support can not continue for long time [5]. Jarvenppa and Ives state that the model of information system to support top leaders should be divided into two dimensions: participation and recognition [6]. This paper argues that these views also adapts to the agile process improvement in P company.

Hypothesis (H1): Support from top leaders plays a positive role in promoting successful implementation of agile process improvement.

3.1.2. Support from Organization

Sharifi believes that organization that has agile capability must get the organization support [7]. In this paper we find that organizational factors that are related to P company are as follows: creating a clear vision, establishing agile organizational culture, the establishment of a learning organization and changing management, etc.

Hypothesis (H2): Support from organization plays a positive role in promoting successful implementation of agile process improvement.

3.1.3. Tool and Technology

Sharifi states that if organization wants to get the agile capacity it must use suitable technology and tools [7]. In 12 principles of agile declaration it states that we should always pay attention to matching of excellent technology and design, so that we can improve software the rapid response capability [8]. P company mostly develop enterprise applications, so some aspects among them are similar, such as security management, document management and clients management, etc. Use of software reuse is appropriate to obtain agility.

Hypothesis (H3): Use of the tools and technology plays a positive role in promoting successful implementtation of agile process improvement.

3.1.4. Suitable Import Method

The number of the software development projects is more than 10 every year. In small ones it needs ten employees, while in larger ones it needs 50 or 100 employees. Agile methods not only require certain skills, but also need employee's self-discipline [9].

Hypothesis (H4): Suitable import method plays a positive role in promoting successful implementation of agile process improvement.

3.1.5. Education and Training

Martin Fowler said "Whatever you select some technology, it's not easy to make it clear how the specific implementation. Agile methods are particular, because it will need you to change the mind! Many people just focus on specific practices rather than the philosophy behind them. Is it possible that you ignore the philosophy of the system and look forward to good results?" [10,11].

Hypothesis (H5): Education and training play a positive role in promoting successful implementation of agile process improvement.

3.2. Analysis of Control Variables

Three factors are selected as control variables in successful implementation of agile process improvement in P company: 1) Support from clients. Without their understanding, support and cooperation, agile software process is difficult to successfully implement in P company. 2) Types of project contract. It's more difficult to implement if fixed price contract is used in some project. 3) Types of clients. Two types of software product are offered by P company. One is E-Government software for departments of government; the other is enterprise business software. In this paper we define the types of clients as control variable because they do not have enough strength to select customers.

3.3. Successful Agile Implementation

Kieran states that the organizations' agility is mainly reflected in four aspects: 1) Innovation. Agile methods provide innovation, and better and new method should be continuously explored to solve the problem. 2) Rapid response to change. When organizations are in rapidly changing environment, we should get rapid response to change and adjust organization to survive. 3) Initiative. Rapid response is passive, and always after the change. Organizations should not wait for the appearance of change, take action before the change, or even create the change. 4) Learning. Organizations should not only get response to change and create it, and learning knowledge from it. Organizations should have a good learning ability to be creative, in order to adapt to change and create it. In this paper whether the agile process improvement is success is determined according to the four aspects above. Critical Success Factor Model in agile process improvement in P Corporation is shown in **Figure 1**.

3.4. Questionnaire Design and Development

3.4.1. The Composition of the Questionnaire

The questionnaire includes four parts: 1) The introduction of questionnaire. Content and basic situation are included. 2) Basic information of samples: such as duty, work experience, and their understanding of agile, etc. 3) Main part. 46 questions are used to measure critical success factors. 4) Measurements of success implementtation. Four questions are included in it (**Table 1**). Control variables are not included in question items, because they are used to make exploratory research.

3.4.2. Questionnaire Delivery and Data Collection

In this paper we distribute the questionnaires in e-mail in P company with help of the sector vice president. A total of 80 questionnaires are distributed, 51 valid questionnaires are recovered. Staff in investigation, including sector vice president and senior project managers, project managers, technical specialists (architects and analysts), programmers, testers and system administrators, covers almost all the roles related with agile process improvement in company (**Table 2**).

Table 1. Statistics of positions in investigation.

Positions	Number of Samples	Percent
Management (department vice president, senior project manager, project manager)	8	15.69%
Technical experts (system architects, system analysts)	10	19.61%
Programmer	23	45.10%
Testing and system administrators	10	19.61%
Total	**51**	**100.00%**

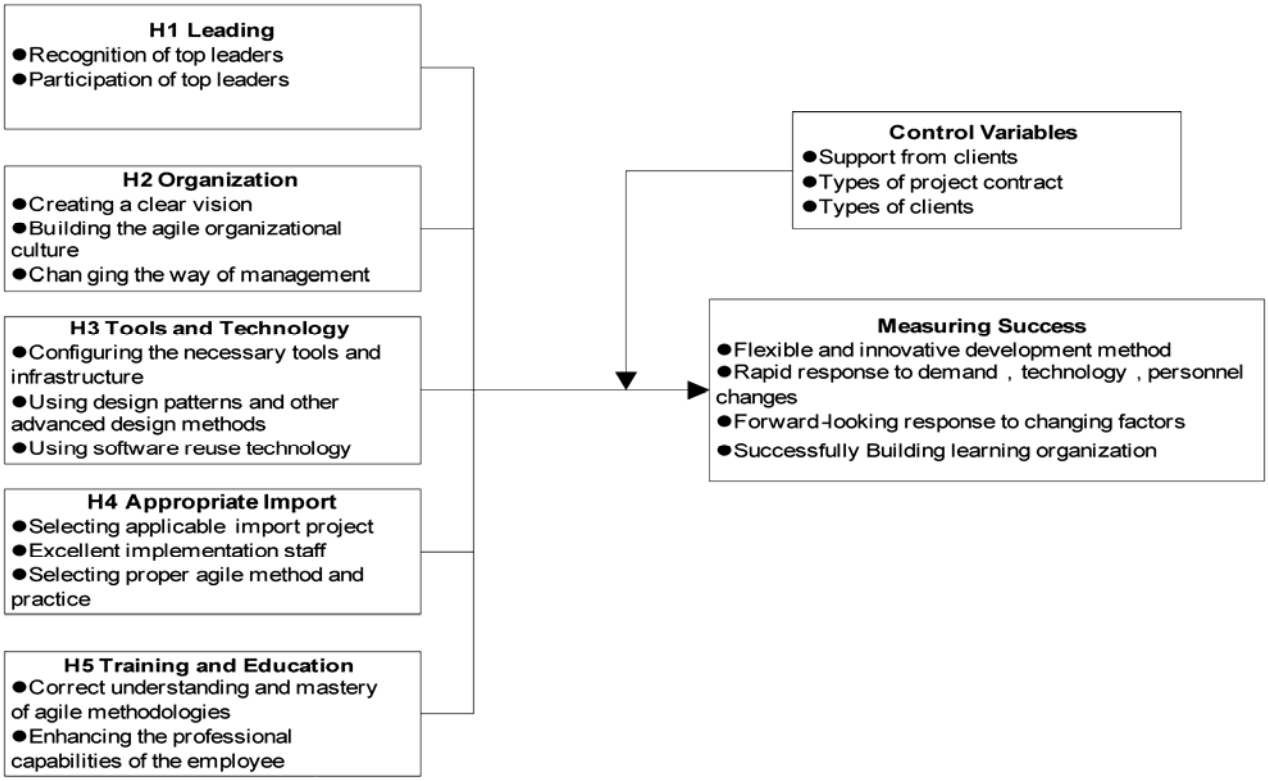

Figure 1. Critical success factor model in agile process improvement in P Corporation.

Table 2. Scales table composition.

Research Factors	Dimensions	References	Question Items
Leading	1) Recognition of top leaders 2) Participation of top leaders	Jarvenppa etc.	Q5-Q9
Organization	1) Creating a clear vision 2) Building the agile organizational culture 3) Changing the way of management	Sharifi Jim Highsmith Kruchten etc.	Q10-Q16
Tools and Technology	1) Configuring the necessary tools and infrastructure 2) Using design patterns and other advanced design methods 3) Using software reuse technology	Sharifi Erich Gamma etc.	Q17-Q22
Appropriate import	1) Selecting applicable import project 2) Excellent implementation staff 3) Selecting proper agile method and practice	Scott Ambler Internal interview	Q23-Q25
Training and Education	1) Correct understanding and mastery of agile methodologies 2) Enhancing the professional capabilities of the employee	Martine Fowler Boehm Internal interview	Q26-Q27
Measuring Success	1) Flexible and innovative development method 2) Rapid response to demand, technology, personnel changes 3) Forward-looking response to changing factors 4) Successfully Building learning organization	Kieran	Q28-Q31

4. Validation of Critical Success Factor Model of P Corporation

4.1. Validity

This paper is validated from face validity, content validity, construct validity [12].

4.1.1. Face Validity

We make a pre-survey of small sample before the official release, according to survey results we adjust the questionnaire to ensure that the contents of the questionnaire is acceptable for the most. Therefore, this questionnaire meets the requirements of face validity.

4.1.2. Content Validity

Most of the critical success factors and performance evaluation indexes are selected in the research results of experts and scholars, while small part of them are sorted and classified by the interviews. Therefore, content validity meets the requirements of research.

4.1.3. Construct Validity

In **Table 3**, KMO > 0.6, the degree of common variance among the five variables is "mediocre" bordering on "middling". While the significance probability of Bartlett's Test is 0.000 (< 0.01), statistics is suitable to factor analysis. **Table 4** shows that the extracted principal components factors consistent with the hypothesis. But they are different as follows: 1) Q8 of the leading is divided into a main component alone, can not be merged into any other components. 2) Training and education factor and appropriate import factor are combined into a principal component factor; it suggests that we should strengthen training and education at the early stage of agile methods. 3) Q14 (Building the agile organizational culture) of the organization is included into the factor of

Table 3. Results of KMO and Bartlett's test.

KMO		0.656
Bartlett's Test	Approx. Chi-Square	1039.763
	df	253
	Sig.	0.000

Table 4. Component matrix.

Research Factors	Question Items	Loading Factors				
		Factor 1	Factor 2	Factor 3	Factor 4	Factor 5
Tools and Technology	Q18	0.921				
	Q17	0.867				
	Q21	0.777				
	Q19	0.752				
	Q20	0.741				
	Q22	0.699				
Organization	Q12		0.882			
	Q11		0.823			
	Q15		0.771			
	Q13		0.755			
	Q16		0.715			
	Q10		0.624			
Training and Education	Q27			0.804		
	Q26			0.759		
Appropriate Import	Q25			0.836		
	Q23			0.789		
	Q24			0.607		
Organization	Q14			0.600		
Leading	Q9				0.856	
	Q5				0.841	
	Q7				0.707	
	Q6				0.631	
	Q8					0.664

training and education, because they are linked closely with the concepts.

4.2. Reliability

Table 5 shows that the reliability analysis results of critical success factors and performance evaluation indexes. Most factors' Cronbach's Alpha are around 0.8, the minimum one is 0.690. So questionnaire has high reliability.

5. Discussion

Through empirical research in this paper, we find that: 1) Training and education plays a important role in promoting agile process improvement. We must have professional skills, good professional basic knowledge to achieve the objective. We suggest that learning organizations needs to be built in organization. 2) Agile methods must be established within agile culture, mainly refers to mutual trust and cooperation of the corporate culture. This study suggests that we should establish the corporate culture of mutual trust to improve development efficiency. In addition, collaboration within the development teams could reduce duplication of effort and greatly improve development efficiency. 3) Attention to the design and application of advanced technology do not receive widespread support. Tools and technical factors related to the question Q18~Q22 scores are generally lower than other factors, most of them are around 3 to 4. But at least it shows that too much emphasis on the importance of design and technology is not correct. Of course, it may be not suitable in other situations because it's just a case. For example, the discipline of the developer is very critical in the agile methods, but it is not selected as a critical success factor because of the strict process management of P company. This paper argues that the increasingly popular open-source software community is typical model of the successful agile software development, reflecting the direction of knowledge creation at the Internet age, and it is worthy of deep study and promoting.

6. Knowledge Creation in Open Source Software Community

In our understanding, software development is knowledge innovation process in the nature. Open source development is both particular bazaar development and typical agile development, so we can derive knowledge creation principles of Open source development from factors of knowledge creation both existing in two type developments.

6.1. Knowledge Creation Theory

Nonaka states that innovation, which is key form of organizational knowledge, cannot be explained sufficiently in terms of information processing. Innovation is a process in which the organization creates and defines problems and then actively develops new knowledge to solve them [13]. He advanced SECI model and bar theory, and pointed out that there are two major latitudes as theory framework of organizational knowledge creation: knowledge's ontology and epistemology. The former includes three levels: individual, group and organization, and the latter includes the tacit knowledge and the explicit knowledge. The key of knowledge creation is to convert tacit knowledge into explicit knowledge. In the process of organizational knowledge creation, organization is to provide proper local to benefit individual's knowledge creation and accumulations and related group activities. He also suggested five enabling conditions on organizational level, which are intention, autonomy, chaos/fluctuation, redundancy and requisite variety [14].

6.2. Knowledge Creation in Values of Agile Development

We will analyze factors of knowledge creation in values of agile development through five enabling conditions of organization knowledge creation, and then find out which values can promote knowledge creation. The relationship between the values of agile development and five enabling conditions of knowledge creation is illustrated in **Table 6** we find Individuals and interactions, Working software, Customer collaboration and responding to change all involve enabling conditions. Therefore, we consider they promote knowledge creation.

6.3. Rules of Bazaar Development

Eric Raymongd described open source development style lively and in detail—release early and often, delegate everything you can, be open to the point [15]. He considered that development style seemed to resemble a great babbling bazaar of different agendas and approaches, which was quite different from traditional software deve-

Table 5. Cronbach's Alpha.

Research Factors	Question Items	Cronbach's α
Leading	Q5~Q9	0.690
Organization	Q10~Q16	0.886
Tools and Technology	Q17~Q22	0.899
Appropriate import	Q23~Q25	0.815
Training and Education	Q26~Q27	0.863
Measuring Success	Q28~Q31	0.778
Total		0.855

Table 6. Analysis of knowledge creation in values of agile development.

Enabling / value	Intention	Autonomy	Fluctuation and Creative Chaos	Redundancy	Requisite Variety
Individuals and interactions	√	√	√		
Working software	√				
Customer collaboration	√	√		√	√
Responding to change	√		√		
Individuals and interactions	√	√	√		
Working software	√				
Customer collaboration	√	√		√	√
Responding to change	√		√		

Note: √ indicates that corresponding value involves corresponding enabling condition, that is to say, corresponding value can promote knowledge creation.

lopment. He called it bazaar development and concluded 19 rules as follows: 1) Every good work of software starts by scratching a developer's personal itch; 2) Good programmers know what to write. Great ones know what to rewrite (and reuse); 3) "Plan to throw one away; you will, anyhow."[16]; 4) If you have the right attitude, interesting problems will find you; 5) When you lose interest in a program, your last duty to it is to hand it off to a competent successor; 6) Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging; 7) Release early. Release often. And listen to your customers; 8) Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone; 9) Smart data structures and dumb code works a lot better than the other way around; 10) If you treat your beta-testers as if they're your most valuable resource, they will respond by becoming your most valuable resource; 11) The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better; 12) Often, the most striking and innovative solutions come from realizing that your concept of the problem was wrong; 13) "Perfection (in design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away"; 14) Any tool should be useful in the expected way, but a truly great tool lends itself to uses you never expected ; 15) When writing gateway software of any kind, take pains to disturb the data stream as little as possible—and never throw away information unless the recipient forces you to; 16) When your language is nowhere near Turing-complete, syntactic sugar can your friend; 17) A security system is only as secure as its secret. Beware of pseudo-secrets; 18) To solve an interesting problem, start by finding a problem that is interesting to you; 19) Provided the development coordinator has a communications medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one.

6.4. Knowledge Creation in Rules of Bazzar Development

The relationship between rules in bazzar development and agile development values those promote the knowledge creation is illustrated in **Table 7**.

6.5. Ten Principles for Knowledge Creation in OSS Development

The principles, including both enabling knowledge factors and agile development features, are in the following: **Principle 1. Self-organizing**. Open source software (OSS) project teams are all self-organizing teams, since team members work together on designing and developping because of common interest, so it is free to join and exit. Team members are all in high work enthusiasm, willing to think and contribute, and can bring best architectures, requirement and design. **Principle 2. Code-sharing**. Demand change is the related bottle-neck among all the models of individual code, however, code sharing eliminates the bottle-neck. It accelerates the speed of development. Open Source community seems this as one of the most basic principles, as this is the best form to realize the knowledge sharing. **Principle 3. Adaptation**. OSS development encourages adaptive plan, not predictable practice. That is to say, detailed schedule can not be based on short-term view establishment. What's more, at regular intervals, the team reflects on how to become more effective, then turns and adjusts its behavior accordingly. **Principle 4. Usability**. The soft ware should be usability. These, including right concept for software architecture, rational design, information enough and assured security, are important for usability. **Principle 5. Sustention**. It is free to join or exit an OSS project. However, though persons with ability in an OSS team flow fluently, it still keeps a good successor. The project leader turns his work over next leader when he

Table 7. Analysis for knowledge creation in bazaar development.

Principles / Value	R 1	R 2	R 3	R 4	R 5	R 6	R 7	R 8	R 9	R 10	R 11	R 12	R 13	R 14	R 15	R 16	R 17	R 18	R 19
Individuals and interactions	√				√		√			√	√					√		√	√
Working software									√			√	√		√		√		
Customer collaboration		√		√				√											
Responding to change			√			√	√	√						√					√

Note: √ indicates that corresponding value involves corresponding enabling condition, that is to say, corresponding value can promote knowledge creation.

exits an OSS project. **Principle 6. Talent**. The number of the membership is not restricted. OSS is community-based model different from firm-based model. This model won't be constrained in the organizational boundary. By technology-mediated interaction, it also won't be constrained in the geographical boundary. **Principle 7. Interaction**. Release early. Release often. Respond quickly. OSS development take stepwise deliver mode. To release early and often, and get quick feedback, is one of the most important parts for Linux development model. It promotes fast flow of knowledge. **Principle 8. Collaboration**. OSS development emphasizes teamwork and collaboration, and its bazaar development model is collaborative development model. An OSS project assembles a number of developers and larger areas of co-developers. If you don't promote teamwork and collaboration by communication, you can't make the project go on smoothly. **Principle 9: Happiness**. An OSS project is attractive and challenging. Developers work together on the OSS project because of common interest. They generally take pleasure in a task when it falls in a sort of optimal challenge zone; not so easy as to be boring, not too hard to achieve. A happy programmer is one who is neither underutilized nor weighed down with ill-formulated goals and stressful process friction. **Principle 10: Democracy**. OSS development style cannot be based on power relationships—and even if they could be, leadership by coercion would not produce the results we see. The achievement of OSS goal is only simple aggregations of many wishes, and it is exact what OSS needed. Power relationships won't work in OSS project volunteers set up.

7. Features of Knowledge Creation in Open Source Community

Knowledge creation in the bazzer and the cathedral is illustrated in **Table 8**. It is obvious the advantages in the bazzar by the ten principles of knowledge creation.

In our understanding, there are three advantages for OSS development in the following.

First, Knowledge Sharing Intensively. In traditional software development, most knowledge is stored in the brains of developers, which belongs to the tacit knowledge. Although companies adopt documentation and repository, hoping to convert the tacit knowledge into explicit knowledge of companies, employees don't have a high enthusiasm for knowledge sharing. Therefore, there is less knowledge sharing. In OSS development, OSS license provides a legal environment of code sharing, and OSS itself is the essence of OSS development. Developers in OSS development are volunteers who are willing to share the code and communicate with each other and have spirit of collaborative development and teamwork.

Second, Knowledge Conversion Rapidly. Traditional software development takes a long period of time to release a new version. Surely it also takes a long period time to get feedback. This kind of development environment fluctuates little, and knowledge flows slowly. Taking a long time to get feedback means it has a long cycle to get new information, so it prolongs speed of conversion. In OSS development, the frequent experimental release assures the quick conversion from tacit knowledge into explicit knowledge. And feedback through quick peer review assures developers gain new useful information fast. Then adaptive feature of development team makes the explicit knowledge converse into tacit knowledge quickly.

Third, Top Talents' Quality and Quantity. Traditional software development is almost firm-based model, and the gaining of developers is constrained by organizational and geographical boundaries. However, the main part of knowledge creation is person, so this kind of model necessary restricts knowledge creation. OSS development is community-based model; members are not restricted in organizational and geographical boundaries. What's more, open source community does not welcome poor developers, so members are top talents.

Table 8. Knowledge creation in the bazzer and the cathedral.

	The Bazzar	The Cathedral	The Principle
Speed for Knowledge	Quick	Slow	3, 5, 7
Knowledge Sharing	High	Low	2, 4, 8, 10
Knowledge Talent	Wide & Much	Narrow & Less	1, 6, 9

Table 9. Knowledge creation principles of open source developpment emerge in apache server.

Principle	Description
Self-organizing	Anyone with an interest in working on Apache development can join the developer mailing list, which was archived monthly.
Code sharing	It was OSS software.
Adaptation	Mailing list ensures that the changes are appropriate.
Usability	The most widely deployed web server, Rational software architecture, Good in reliability and expandability
Sustention	New developers tend to focus on areas where the former maintainer is no longer interested in working, or in the development of new architectures and features that have no preexisting claims (frontier building).
Talent	The participation in Apache development overall was quite wide, with almost 400 individuals contributing code that was incorporated into a comparatively small product.
Interaction	Since anyone can subscribe to the mailing list, the changes are reviewed by many people outside the core development community, which often results in useful feedback before the software is formally released as a package.
Collaboration	182 people contributed to 695 PR changes, while 249 people contributed to 6092 non-PR changes.
Happiness	Developers tend to work on problems that are identified with areas of the code they are most familiar. Some work on the product's core services, while others work on particular features that they developed.
Democracy	The Apache software development process is a result of both the nature of the project and the backgrounds of the project leaders. It was clear from the very beginning that a geographically distributed set of volunteers, without any traditional organizational ties, would require a unique development process in order to make decisions.

8. Case Study

The Apache server [17] is, according to the Netcraft survey, the most widely deployed web server at the time of this writing. In fact, the Apache server has grown in "market share" each year since it first appeared in the survey in 1996. By any standard, Apache is very successful [18]. The knowledge creation principles of Open Source development emerge in Apache server (**Table 9**).

9. Conclusions

The conclusions are as follows: 1) Education and training play a positive role in promoting successful implement-tation of agile process improvement. 2) Agile methods must be established within agile culture, mainly refers to mutual trust and cooperation of the corporate culture. 3) Attention to the design and application of advanced technology do not receive widespread support. We suggest that too much emphasis on the importance of design and technology is not correct in P company. In the end, we use applications knowledge creation theory to analyze the open source software community with successful application of the typical agile software method, propose ten principles of knowledge creation in open source software community, features of knowledge creation in open source community, case study for Apache server development, and we hope that more researchers could join in the study and practice.

10. Acknowledgements

Thanks for helpful discussion with Mr. Yuan Weihua, Mr. Zhou Zhijun, Mrs. Zou Wei and Mr. Wang Shuwen.

REFERENCES

[1] K. Beck, M. Beedle, *et al*., "Manifesto for Agile Software Development," 2009. http://agilemanifesto.org

[2] S. McConnell, "Rapid Development Taming Wild Software Schedules," Microsoft Press, Washington DC, 1996.

[3] C. L. Liu, "The Tutorial of Information Systems Project Management," Tsinghua University Press, Beijing, 2005.

[4] M. C. Paulk, "Extreme Programming from a CMM Perspective," *IEEE Software*, Vol. 18, No. 6, 2001. pp. 19- 26. doi:10.1109/52.965798

[5] M. C. Paulk, "Software Process Proverbs," *Crosstalk*: *The Journal of Defense Software Engineering*, Vol. 10, No. 1, 1997, pp. 4-7.

[6] Q. F. Min, "The Empirical Study of Critical Success Factors of ERP Implementation in China," Doctoral Dissertation, Dalian University of Technology, Dalian, 2005.

[7] H. Sharifi and Z. Zhang, "A Methodology for Achieving Agility in Manufacturing Organizations: An Introduction," *International Journal of Production Economics*, Vol. 62, No. 1-2, May 1999, pp. 7-22. doi:10.1016/S0925-5273(98)00217-5

[8] J. Z. Zhang, L. Q. Qian and S. Y. Zhu, "An Overview of Agile Methodology," *Computer Applications and Software*, Vol. 19, 2002, pp. 1-9.

[9] J. Highsmith, "Agile Project Management: Creating Innovative Products," Addison Wesley, Boston, 2004.

[10] J. Li, "Martin Fowler Talks Scrum Certificate Authority, Agile Today and Tomorrow," 2008. http://www.infoq.com/cn/news/2008/06/martin-agile-scrum

[11] K. Conboy and B. Fitzgerald, "Agile Drivers, Capabilities, and Value: An Over Arching Assessment Framework for Systems Development," In: K. C. Desouza, Ed., *Agile Information Systems*; *Conceptualization*, *Construction*, *and Managemen*t, Butterworth-Heinemann, Burlington, 2007, pp. 207-221.

[12] Z. Y. Lin, "SPSS Operation and Application," Peking University Press, Beijing, 2007.

[13] I. Nonaka, "The Knowledge Creating Company," *Harvard Business Review*, Vol. 69, No. 6, 1991, pp. 96-104.

[14] G. von Krogh, *et al.*, "Enabling Knowledge Creation: How to Unlock the Mystery of Tacit Knowledge and Release the Power of Innovation," Oxford University Press, New York, 2000.

[15] E. Raymond, "The Cathedral and the Bazaar," *Knowledge*, *Technology*, *and Policy*, Vol. 12, No. 3, 1999, pp. 23-49. doi:10.1007/s12130-999-1026-0

[16] F. P. Brooks, Jr., "No Silver Bullet: Essence and Accidents of Software Engineering," *Computer*, Vol. 20, No. 4, 1987, pp. 10-19. doi:10.1109/MC.1987.1663532

[17] A. Mockus, R. T. Fielding and J. Herbsleb, "A Case Study of Open Source Software Development: The Apache Server," *Proceedings of the* 22*nd International Conference on Software Engineering*, Limerick, 4-11 June 2000, pp. 263-272.

[18] Netcraft Survey. http://www.netcraft.com/survey

