Managing Knowledge in Development of Agile Software

Mohammed Abdul Bari
Department of Computer Science, College of Science & Arts
University of Al-Kharj
Wadi Al-Dawasir-11991, Kingdom of Saudi Arabia

Dr. Shahanawaj Ahamad
Department of Computer Science, College of Science & Arts
University of Al-Kharj
Wadi Al-Dawasir-11991, Kingdom of Saudi Arabia

Abstract— **Software development is a knowledge-intensive work and the main attention is how to manage it. The systematic reviews of empirical studies presents, how knowledge management is used in software engineering and development work. This paper presents how knowledge is used in agile software development and how knowledge is transferred to agile software using agile manifesto. It then argues for the need to scale agile development strategies in knowledge management to address the full delivery. The paper explores the eight agile software scaling factors with knowledge management and their implication for successfully scaling of agile software delivery to meet the real world needs of software development organization.**

Keywords- Knowledge management; Agile software; Scaling factor; Agility; Knowledge capturing

I. Introduction

Knowledge management is "A method that simplifies the process of sharing, distributing, creating, capturing and understanding the company knowledge [1]. Argyris [2] define "Knowledge is a fluid mix of framed experience, values, contextual information and expert insight that provide a frame work for evaluation and incorporating new experience and new information. According to Nonoka and Takeuchie [3] Knowledge passes through different modes of conversion , which makes the knowledge more refined and spreads it across different layers in an organization.

II. Knowledge Management in Software Development

Software development is a knowledge intensive activity. The main assets of software development are not manufacturing plants, building and machines but the knowledge held by the employees and development culture of organization. Software development has long recognized the need for managing knowledge so that the community could learn from the knowledge management. As the field of software engineering matures, there is an increase demand for empirically validated results and not just the testing of technology [4].

Companies developing information system have fail to learn effective means for problem solving to an extent that they have learned to fail [5]. The main differences between methods are they are plan based or traditional, which rely primary on managing explicit knowledge or agile method [6]. There has been much discussion in software development, how to manage knowledge reusing life cycle experience which is gain by processing and producing software development projects which is often referred as experience factory [7] which is stored in experience base, by storing generalizing, tailoring and formalizing experience so that it is easy to reuse. In May 2002 issue of IEEE software [8] was devoted to knowledge management in software engineering, giving several example of knowledge management, applications in software companies. In 2003, the book "Managing Software Engineering Knowledge "[9] was published touching various range of topics from identifying why knowledge management is important in software engineering and development [10].

III. Knowledge Management in Agile Software

A. *Agile Software development*

It consists of set of practice for software development, which has been created by experience practitioners [11]. In Williams and Cockburn [12] stated that agile development is "about feedback and change ". Agile software development techniques have taken the industry by storm, nearly 76% of software organization reported in 2009 that they had one or more agile software underway [13]. According to Agile manifesto 2001[14], it underlines 12 basic principles which are given below:

1. The organization highest priority is to satisfy the customer by continuous delivery of software.
2. Welcoming the changes in requirement even at the later part of development.
3. Delivering the software frequently.
4. Business people and developers must work together throughout the project.
5. Build project around individuals, give them environment and support they needed.
6. Face to face conversation in team with developers.
7. Working software is only means to progress.
8. Agile process prototype sustainable developed.
9. Continuous attention result is excellent product.
10. The act of maximizing the amount of work done.
11. Self-organized the team with requirement, architectures and others.
12. Give regular interval to the project team.

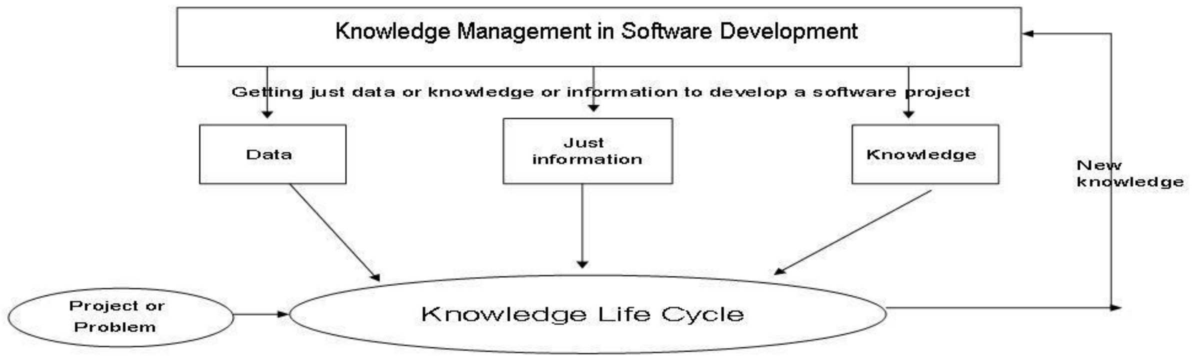
Knowledge Management in Software Development
Getting just data or knowledge or information to develop a software project
Data
Just information
Knowledge
New knowledge
Project or Problem
Knowledge Life Cycle

Figure 1: Knowledge Life Cycle

B. *Agile Methods*

- Agile Modeling (AM): it is a practice based methodology for modeling and documentation of software based system. It is planned to be a collection of values, principal and practice for modeling software that can be applied on software development project in flexible manner. [15]
- Agile Unified Process (AUP): it is simplified version of RUP (Rational Unified Process). It describe a simple, easy to understand approach to developed business application software using agile technology and concept [16]
- Dynamic System Development Method (DSDM): It is based upon Rapid application development methodology [17]. In 2004 DSDM become generic approach to project management and solution delivery. It emphasized continuous user /customer involvement [18].
- Essential Unified Process (EssUP): It was invented by Ivar Jacobson [19] which is an improvement of Rational Unified Process. It uses use case, iterative development, architecture driven development, team practice and process practice which is borrowed from RUP (Rational Unified Process) [19]. The main idea here is that you can pick those practices that are applicable to your situation and combine them in to yours own process.
- Extreme Programming (XP): It is a software development methodology which is planned to improve software quality by changing customer requirement. It frequently releases a short development cycle which is intended to improve productivity and introduce check point where the new customer requirement can be adopted.[20]
- Open Unified Process (OpenUP): It is an open source process developed within conceals foundation. It preserves the essential characteristics of RUP/unified process [21] which include incremental development, use case and scenarios deriving development. [22].
- Scrum: It is an iterative increment methodology for project management often seen in agile software development. Although it is mainly used for management of software development. It can also be used to run software maintenance.[23]
- Velocity: It measure the productivity in agile software development .Velocity tracking is an act of measuring said velocity. The velocity is calculated by counting the number of unit of work completed in certain interval, determined at the start of the project. [24]
- Feature Driven Development (FDD): It is an iterative and incremental software development process. FDD blends a number of industry recognized best practices like domain object modeling, developing by feature, individual class, feature teams, inspections, configuration management, regular builds and visibility of progress and result in to cohesive whole. These practices are all driven from client-valued feature perspective. Its main purpose is to deliver tangible, working software repeatedly in timely manner. [25]

IV. Knowledge Management in Agile Software

The knowledge is capture from agile software, it is also capture from market research, surrounding area and scientific method, kept in knowledge management box (which consist of people, rules, method (old and new) files etc.). Whenever an organization gets a project the developers will search their knowledge box, they learn from it before starting of project or during the making and also after making the project. Sometime an innovative method is developed for a particular project , once the project is developed the method is send to knowledge management box so that it can used at later part for different project .

V. Agile Software in Scale

In early days agile software development techniques were small and relatively straight forward. Today the picture has changed and organization want to apply agility software techniques to a broader set of project. They are dealing with problems which requires large teams, distribute work force many more. They are eight scaling factor that define agility software.

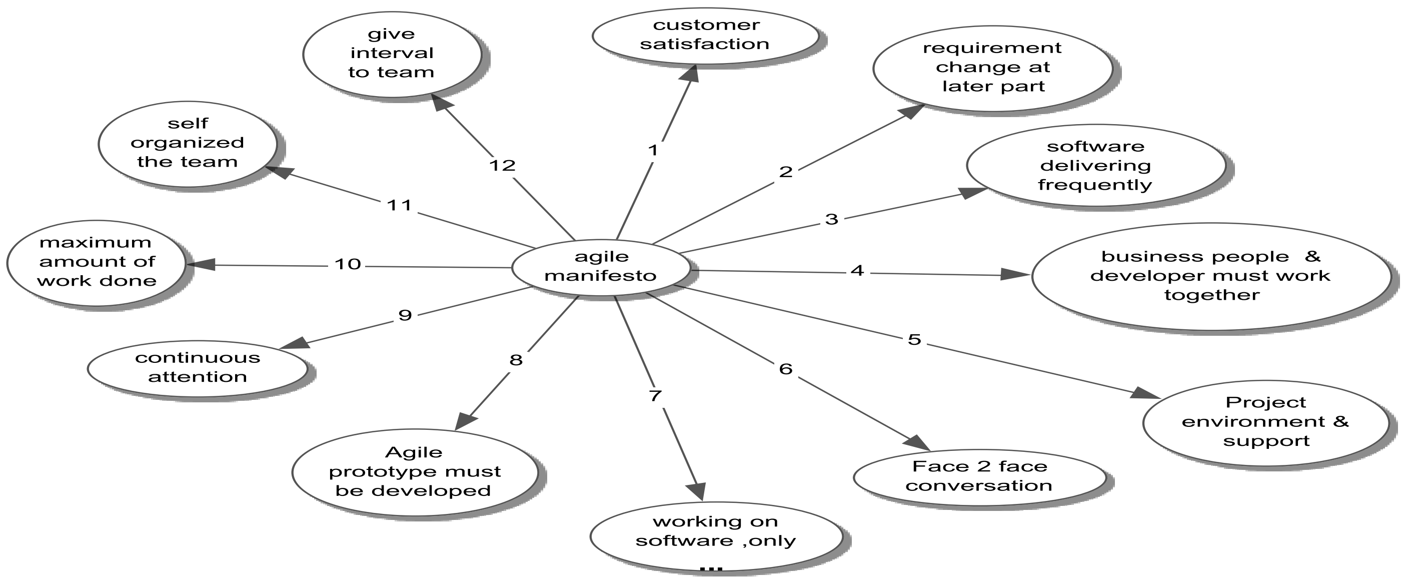
agile manifesto
customer satisfaction
requirement change at later part
software delivering frequently
business people & developer must work together
Project environment & support
Face 2 face conversation
working on software ,only ...
Agile prototype must be developed
continuous attention
maximum amount of work done
self organized the team
give interval to team
1
2
3
4
5
6
7
8
9
10
11
12

Figure 2: Agile Manifesto

- Team Size: Agility process work very good when the team size is small, as the team size increases communication risks increases and co-ordination become more difficult. In order to learn something from the past, they have to go through the knowledge management capture process which is created by previous projects.
- Geographical Distribution: when the team size is distributed in different countries, effective collaboration become more difficult, more challenging and more error likely to occur and it became more difficult to capture the knowledge from knowledge management capture process.
- Regulatory Compliance: Issues such as ISO 9000[26], these mandates bring requirement of their own, this means, the formality of the work has to increase.
- Domain Complexity: some project team find themselves addressing a straight forward problem, more complex domain require greater emphasis on exploring and experimenting [27].
- Organization Distribution: many project teams includes members from different division, different partner companies or from some external service firms. The more organizationally distributed teams, the more the relationship will be contractual.
- Technical Complexity: Some applications are more complex than others. It is easy to achieve high level quality if you're building a new system from scratch but it is not easy to develop a new application with the existing agility software.
- Organization complexity: your existing organization structure and culture may reflect waterfall [28] values, which increases the complexity of adopting and scaling agile strategies within your organization.
- Enterprise Discipline: Many organizations want to have common infrastructure platform to lower the cost, reduce time, and improve consistency, that is very difficult if project team focus only on their immediate needs.

VI. Future Scope

The scope of this study suggested that agile software development is effective and suitable for many situation and environment. However, at present only few empirically validated studies can be found to support the claims. More ever, the frequent releases of new agile software development methods also bring confusions rather than clarity. The urgent need now (more than new model) is to adopt a few particular methods which can be used by software professional, projects and organizations to choose a particular method to produce a right product at a right time.

VII. Conclusion

Knowledge management may provide important contribution in developing software. Today, there is no doubt that organizations have exploited their potential to create knowledge by focusing on their developing software members by not only focusing on externally developed information, knowledge or data. Studies have shown that traditional plan-driven software development methodologies are not used in practice. Many organizations have been successful at adopting agile software development approaches. Agile scaling model provides a road map for complexities which occur when adopting and tailoring agile method. The knowledge management used in agile software development method provide a novel way of approaching software development problem's , while also maintaining that the method are by no means capable of solving all problems.

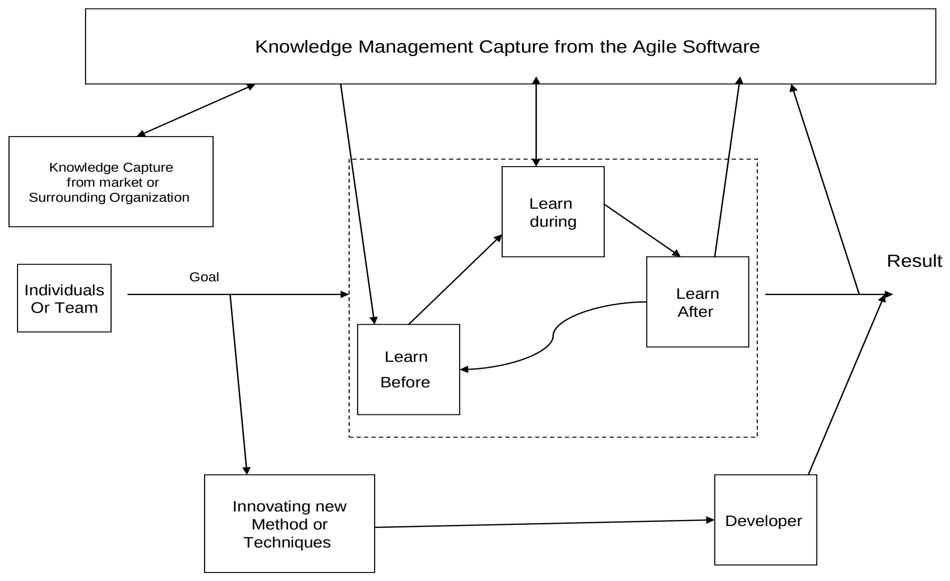
Knowledge Management Capture from the Agile Software
Knowledge Capture from market or Surrounding Organization
Individuals Or Team
Goal
Learn during
Learn After
Learn Before
Result
Innovating new Method or Techniques
Developer

Figure 3: Knowledge Management in Agile Software

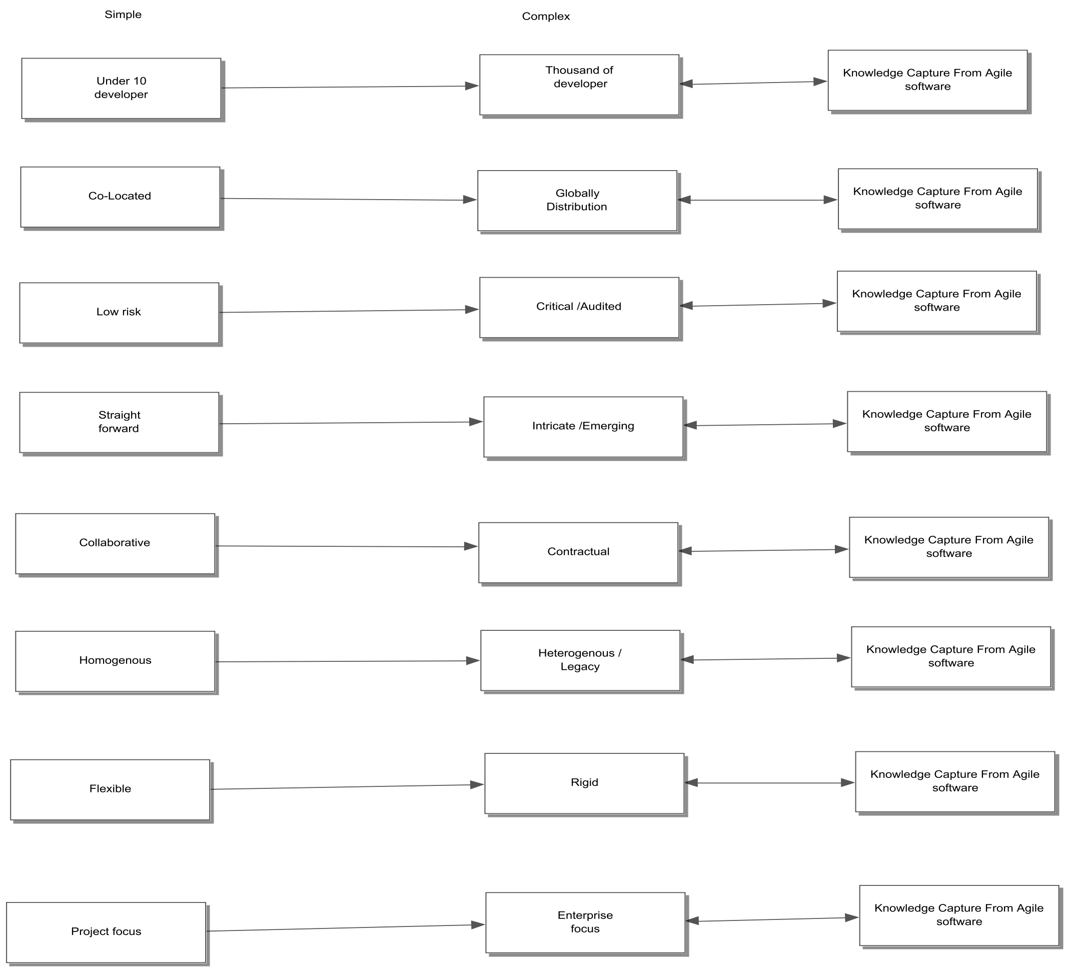
Simple
Complex
Under 10 developer
Thousand of developer
Knowledge Capture From Agile software
Co-Located
Globally Distribution
Knowledge Capture From Agile software
Low risk
Critical /Audited
Knowledge Capture From Agile software
Straight forward
Intricate /Emerging
Knowledge Capture From Agile software
Collaborative
Contractual
Knowledge Capture From Agile software
Homogenous
Heterogenous / Legacy
Knowledge Capture From Agile software
Flexible
Rigid
Knowledge Capture From Agile software
Project focus
Enterprise focus
Knowledge Capture From Agile software

Figure 4: Scaling Factor in Agility

REFERENCES

[1] T.H.Davenprot, L.prusak," *Working Knowledge: how organizations Manage what they Know*", Harvard Business School Press, Boston, USA, 1998.

[2] Argyris C.,"*Knowledge for Action* ". (1993), San Francisco, CA: Jossey-Bass.

[3] I.Nonaka, H.Takeuchi.," *The Knowledge-Creating Company*" , Oxford University Press 1995.

[4] Finn Olav Bjornson , Torgeir Dingsoyr , " *Knowledge management in software engineering : A systematic review of studied concepts , finding and search methods used* ", (2008), International Journal of Information and Software Technology .

[5] K.Lyytinen, D.Robey, "*Learning failure in information systems development* ", (1999), Information system journal.

[6] S.Nerur, V. Baligepally," *Theoretical reflections on agile development methodologies* ", (2007), Communications of the ACM 50 79-83.

[7] V.R.Basili, G.Caldiera, H.D. Rombach," *The experience factory*: ,"(1994) in JJ Marciniak (Ed),Encyclopedia of Sofware Engineering ,J.John WILEY , New York.

[8] M. Lindvall , I.Rus , " *Knowledge management in software engineering* ", (2002), IEEE software .

[9] H.D Doran," *Agile knowledge management in practice"* ,(2004) in :Proceeding of the Sixth International Workshop on Learning Software Organization , Springer Verlag, Banff, Canada.

[10] M. Lindvall, I.Rus," *Knowledge Management for Software Organization*", (2003) in A. Aybuke et al.(Eds.), Managing Software Engineering Knowledge , Springer Verlag, Berlin..

[11] D.Kolb ," *Experiential Learning: Experience as the Source of Learning and Development* ",(1984), Prentice Hall, Englewood Cliffs,USA.

[12]S. Koening, "*Integrated process and knowledge management for product definition , development and delivery* ,(2003) in : Proceeding of the IEEE International Conference on Software-Science , Technology & Engineering .

[13] Dobb's Journal's July 2009 State of the IT Union Survey - www.ambysoft.com/surveys/state-OfITUnion200907.html

[14] Principles Behind the Agile Manifesto - www.agilemanifesto.org/principles.html

[15] Scott w. Ambler, "*Effective practice for modeling and documentation* "(.2007)

http://www.agilemodeling.com/.

[16] Scott w.Ambler." *The Agile Unified Process* ", (2009) http://www.ambysoft.com/unifiedprocess/agileUP.html

[17] Casemaker Totem,"*what is Rapid Application Development*?",(2000)

http://www.casemaker.com/download/products/totem/rad_wp.pdf

[18] Benjamin J.J.Voigt, Dr.M.Glinz,"*Dynamic System Development Method*", (2004), Department of Information Technology, University of Zurich, Retrieve on.

[19] Ivar Jacobson ," *Essential Unified Process* " (2010), http://en.wikipedia.org/wiki/Essential_Unified_Process

[20] Kent Beck ,Cynthia Andres ," *Extreme Programming Explained*",(2004), Addison –Wesley Professional.

[21] Wikipedia ,"*IBM Rational Unified Process* ",(2003). http://en.wikipedia.org/wiki/RUP.

[22] Wikipedia ," OpenUP ", (2009) http://en.wikipedia.org/wiki/Open_Unified_Process

[23] Mike Cohn," *Succeeding with Agile: Software Development Using Scrum*" ,(2009),The Addison –Wesley Series.

[24] Jeremy Weiskotten," *Velocity : Measuring and Planning an Agile Project* ,(2009,

http://agilesoftwaredevelopment.com/blog/jeremy/velocity-measuring and-planning-agil

[25] Wikipedia,"*Feature Driven Development* ", (2009), http://en.wikipedia.org/wiki/Feature_Driven_Development

[26] Hongyi Sun, "*Total quality management, ISO 9000 certification and performance improvement*",(2000) International Journal of Quality & Reliability Management, Vol. 17 Iss: 2, pp.168 – 179.

[27] Kruchten, P. (2009). "*The Context of Software Development* "- http://pkruchten.wordpress.com/2009/07/22/the-context-of-software-development/

[28] Ambler, S.W.," *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process*", (2002) New York: Wiley Press.

[29] Dobb's Journal's July 2009 State of the IT Union Survey - www.ambysoft.com/surveys/state-OfITUnion200907.html

[30] Dobb's Journal's 2008,"*Project Success Survey*" - www.ambysoft.com/surveys/success2008.html

[31] Pekka Abrahamsson ,Outi Salo , Jussi Ronkainen & Juhani Warsta ," *Agile software development method*", (2002), VVT Publication

AUTHORS PROFILE

Mr. Mohammed Abdul Bari is an Information System Architect and expert in handling software process improvement. His research area includes Business Process Reengineering, Process Modeling, Information System Redesign and Reengineering. He did B.E. in Computer Science & Engineering from Bangalore University, INDIA and M.S. in Information Systems from London South Bank University, United Kingdom, currently pursuing Ph.D. in Computer Science from University of Newcastle, District Columbia, U.S.A.

Dr. Shahanawaj Ahamad is an active academician and researcher in the field of Software Reverse Engineering with experience of ten years, working with Al-Kharj University's College of Science & Arts in Wadi Al-Dawasir, K.S.A. He is the member of various national and international academic and research groups, member of journal editorial board and reviewer. He is currently working on Legacy Systems Migration, Evolution and Reverse Engineering, published more than twenty papers in his credit in national and international journals and conference proceedings. He holds M. Tech. followed by Ph.D. in Computer Science major Software Engineering, supervised many bachelor projects and master thesis, currently supervisor of Ph.D. theses.