The Role of Knowledge Management in Agile Software Development

Broderick Crawford[1,2], Claudio León de la Barra[1], Ricardo Soto[1,3], Mario Dorochesi[4], and Eric Monfroy[5]

[1] Pontificia Universidad Católica de Valparaíso, Chile
FirstName.Name@ucv.cl
[2] Universidad Finis Terrae, Chile
[3] Universidad Autónoma de Chile, Chile
[4] Universidad Técnica Federico Santa María, Chile
FirstName.Name@usm.cl
[5] Université de Nantes, France
FirstName.Name@univ-nantes.fr

Abstract. A software engineering project depends significantly on team performance, software is created by people for people involving human cooperation. In the last years, the tradicional software development approaches are changing and agile methods have received considerable appreciation. Among other attributes, the agilists claim that knowledge sharing is one of the keys to response to common problems and challenges of software development today. The agile principles and values have emphasized the importance of collaboration and interaction in the software development and, by other hand, creative work commonly involves collaboration in some form and it can be understood as an interaction between an individual and a sociocultural context. Agile methods had attained worldwide fame for its ability to increase the productivity of software teams by several magnitudes through empowering individuals, fostering a team-oriented environment, and focusing on project transparency and results. Particularly relevant are the team structure (creative and agile roles) and its functioning (creative techniques used).

Keywords: Knowledge Management, Software Engineering, Agile Development, Creativity, Creative teams.

1 Introduction

Knowledge management and associated processes have been incorporated progressively to frameworks of reference and organizational practices related to software development. These developments–more broadly–reach up topics related to organizational learning, complex systems, the individual and his personality [5,6].

One of the fundamental aspects in knowledge management is creation of knowledge and transformation of tacit knowledge into explicit knowledge (epistemological aspect); this process emerges and is performed person to person and group to group (ontological aspect) [14,9,16,18].

Considering the time spent since the publication of the Agile Manifesto [2] and the development of knowledge management, is especially relevant the research about mature development teams in order to reinforce creativity and productivity in software development [5,11,15,16].

2 Fundamental Aspects of Creative Teams

For this purposes it is necessary to consider that a team, particularly creative and productive, must present three fundamental aspects:

- The ordination and regulation criteria related to the dynamic structure of the team, which shows of how people prepare and order for ideation (generation of a new software), and elements (resources) they use.
- The practical criteria related with the team operation, which indicates the dynamics of action that people establish and develop for the ideation, and finally.
- The teleological criteria, referring to the purposes, it means, the objectives of the group of people (both individuals as such). The group itself and its future viability, and effectiveness in achieving the purpose for which they were established [1].

3 Creative Roles in Software Development

Considering that the purpose of team is to develop software, it becomes relevant the criteria related to the structure of the team and its operation.

Regarding the first criterion, the evidence indicates that the definition of the roles established by the main Agile methodologies (Extreme Programming, SCRUM, ...) corresponds to various creative/innovative roles advisable in high performance teams. This correspondence is not seen in the traditional methodologies or taylorians of software development, as these more traditional approaches point to the high standardization, extensive documentation and roles that do not privilege enough the relationship with others (for example, with the client), this traditional view contrasts to creativity itself, which favors dialogue and the interaction between roles in a flexible method (understanding, agile) and sometimes also contradicts with the productivity, in the extent that the developed software does not respond to the requirements and needs of the client.

Particularly in Extreme Programming (XP) roles are the Customer, who writes the requirements and functional testing; the programmer who maintains the code as simple as possible and helps the client to write the functional tests; the Tester, who runs the functional tests; the Tracker, who tracks the estimates made by the team, the coach, who is responsible for the global process; the consultant who has specific expertise and, finally, the Manager, who is a link between the client and the programmers.

Lumsdaine and Lumsdaine [13] have proposed the detective that collects a big quantity of information related to a problem or situation in startup; Explorer, who tries to perceive the problem or situation in a broader sense, defining it more precisely and anticipating changes in a relevant context; the artist, who generates new ideas to solve the problem or to face the situation; the Engineer, who evaluates the generated ideas; the judge, who decides which idea will be implemented and, finally, the producer, who is responsible for implementing selected idea.

Trías de Bes and Kotler [17] in their model A-F to foster creativity, innovation and change in organizations have proposed the Activators roles, those who start the process of innovation; the browsers, who are the specialists in search for information, to generate new ideas and to their implementation; the Creators, who generate new ideas; the Developers, who turn ideas into products and/or specific service; the Executors, who implement and execute ideas in the institution and finally; the Facilitators, who approve the necessary resources for the innovation process and facilitate that the process develops properly.

The equivalence between these different typologies has been developed, concluding a high parallelism between the Agilists roles and the creative roles [3,12,7,8].

4 Operation of the Software Creative Team

Moreover, regarding the performance of team of software development, agile methods propose a series of practices that indicate how the team should develop. Particularly in XP practices are: game planning, 40 hours weekly, small deliveries, metaphors, customer on site, tests, simple design, coding standards, refactoring, pair programming, collective ownership, continuous integration, just the rules and open workspaces.

In our opinion, such practices show the operation of the team and what is expected of people in general, emphasizing what should ensure the software development process, but do not deepen on how are generated the specific proposals in the new software, through specific techniques. This aspect is crucial to consider in mature teams, as it seeks to further enhance their performance and ensure their future viability in other software development projects.

For this purpose, it is possible to use what it has been present in the theory of creativity [4], in which it is argued that it is imperative that new ideas (understanding, new software or new code) are stimulated properly, and to do this, it is required that the process of ideation start from an "specific" product (a software that "preexists" in the customer's mind and surely shows a software problem or an "ideal" software in the mind of the programmer, coach or consultant ...), where constant modifications are made agreeing with the client. These modifications will constitute, in short, the new software.

To this end, a technique widely used is the SCAMPER [10] which proposes seven basic modification operations: substitution, which involves taking one or more elements of the initial "specific" product and change it with new ones; the combination, in which it is added one or more elements or new characteristics to the initial "specific" product ; amplification, which is exaggerated in ascending way one or more characteristics (elements) of the initial “specific” product; minimization, which refers–to the reverse of the previous operation–to reduce one or more characteristics, passing to another use, which consists in altering substantially any characteristic of the initial “specific” product, to such an extent that the final product will present another utility (unrelated to the original); the elimination, in which it is removed one or more characteristics (elements) of initial “specific” product and, finally, the reordering (re inverse), which refers to change the order or the sequence of one or more characteristics (elements) of the initial “specific” product.

5 Conclusions

We considered that a deepening and formalization of this type, both in the comparative study of Agilists roles and creative roles, as in the operation more subtle of the development team at the moment when generating new software, is fundamental, because it allows- in practice- to ensure that tacit knowledge of all team members is effectively results into explicit knowledge and shared.

In mature agile software development teams it is necessary to enhance the support of its operation. In this sense, creative techniques like SCAMPER and the use of initial points such as “preexisting” or “ideal” software can be useful to agilists.

References

1. Aranda, E., Aranda Muñoz, E.: Manual de la creatividad: Aplicaciones prácticas. Vicens Vives (2000)
2. Beck, K.: Agile alliance (2001), http://agilemanifesto.org
3. Crawford, B., de la Barra, C.L.: Enhancing creativity in agile software teams. In: Concas, G., Damiani, E., Scotto, M., Succi, G. (eds.) XP 2007. LNCS, vol. 4536, pp. 161–162. Springer, Heidelberg (2007)
4. de Bono, E., Arnau, A.: El pensamiento lateral práctico. In: Biblioteca Edward De Bono. Ediciones Paidós (2008)
5. Dingsøyr, T., Nerur, S.P., Balijepally, V., Moe, N.B.: A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software 85(6), 1213–1221 (2012)
6. Drury, M., Conboy, K., Power, K.: Obstacles to decision making in agile software development teams. Journal of Systems and Software 85(6), 1239–1254 (2012)
7. Glass, R.: Software creativity. Prentice-Hall, USA (1995)
8. Gu, M., Tong, X.: Towards hypotheses on creativity in software development. PROFES 3009, 47–61 (2004)
9. Jabar, M.A., Sidi, F., Selamat, M.H.: Tacit knowledge codification. Journal of Computer Science 6(10), 1170 (2010)
10. Kotler, P., Trías de Bes, F.: Marketing Lateral. Editorial Pearson/Prentice Hall, Spain (2004)
11. Layman, L., Williams, L., Damian, D., Bures, H.: Essential communication practices for extreme programming in a global software development team. Information & Software Technology 48(9), 781–794 (2006)

12. de la Barra, C.L., Crawford, B.: Fostering creativity thinking in agile software development. In: Holzinger, A. (ed.) USAB 2007. LNCS, vol. 4799, pp. 415–426. Springer, Heidelberg (2007)
13. Lumsdaine, E., Lumsdaine, M.: Creative Problem Solving: Thinking Skills for a Changing World. McGraw-Hill, New York (1995)
14. Nonaka, I., Takeuchi, H.: The Knowledge Creating Company. Oxford University Press (1995)
15. Stankovic, D., Nikolic, V., Djordjevic, M., Cao, D.-B.: A survey study of critical success factors in agile software projects in former yugoslavia it companies. Journal of Systems and Software (2013)
16. Sung, S.Y., Choi, J.N.: Effects of team knowledge management on the creativity and financial performance of organizational teams. Organizational Behavior and Human Decision Processes 118(1), 4–13 (2012)
17. Trías de Bes, F., Kotler, P.: Winning at Innovation: The A-to-F Model. Palgrave Macmillan (2011)
18. Wang, L.: The multi-dimension spiral model of the knowledge-based corporation - theoretical thinking of beijing future advertising corporation. JSW 4(5), 469–477 (2009)