DEVELOPMENT OF ALGORITHMS TO SOLVE COMBINATORIAL PROBLEMS

Broderick Crawford, Carlos Castro
Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile
Universidad Técnica Federico Santa María, Valparaíso, Chile

Eric Monfroy
LINA, Université de Nantes, Nantes, France

Keywords: Software Engineering, Agile Development, Knowledge Management, Creativity, Optimization Algorithms.

Abstract: In this paper we present important human and social factors in relation with the development of algorithms to solve combinatorial problems using different techniques. We fix some concepts from knowledge management and software engineering as applied developing optimization algorithms, solvers and metaheuristics.

1 INTRODUCTION

The development of algorithms to solve optimization problems requires more knowledge than any single person can possess. It requires the collaboration of numerous individuals with complementary skills. The necessary resources to solve problems are distributed among the stakeholders and creative solutions emerge out of collaborative work. Creative thinking is an area that has been ignored in the development of Optimization Algorithms. Nevertheless, its successful application in the real world depends on a high degree of creativity and innovation (Vidal, 2005). The development of Optimization Algorithms and Metaheuristics to solve Combinatorial Problems assumes the same connotations it assumes in the field of Software Engineering. Then, the software development life cycle of them might be quite diverse and different models from other fields can be appropriate. This paper captures our experience with valuable concepts from Knowledge Management and Software Engineering applied when we are developing algorithms.

In Software Engineering, the traditional approaches (often referred to as plan-driven, task-based or Tayloristic), like the waterfall model and its variances, facilitate knowledge sharing primarily through documentation. They also promote usage of role based teams and detailed plans of the entire software development life-cycle. It shifts the focus from individuals and their creative abilities to the processes themselves. In contrary, agile methods emphasise and value individuals and interactions over processes.

The agile principles and values have realized 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, the study of the potential of techniques to foster creativity in software engineering is a very interesting issue (Gu and Tong, 2004).

There are few studies reported on the importance of creativity in software development. In management and business, researchers have done much work about creativity and obtained evidence that the employees who had appropriate creativity characteristics, worked on complex, challenging jobs, and were supervised in a supportive, noncontrolling fashion, produced more creative work. Since human creativity is thought as the source to resolve complex problem or create innovative products, one possibility to improve the software development process is to design a process which can stimulate the creativity of developers.

We believe that in Optimization Algorithms development projects, a better understanding of some value and interdisciplinary concepts from Creative Solving Problem (Vidal, 2005) and Knowledge Management (Nonaka and Takeuchi, 1995) offers important insights about the use of Software Engineering methodologies.

Crawford B., Castro C. and Monfroy E. (2007).
DEVELOPMENT OF ALGORITHMS TO SOLVE COMBINATORIAL PROBLEMS.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - ISAS*, pages 636-639
DOI: 10.5220/0002407006360639

2 KNOWLEDGE MANAGEMENT IN SOFTWARE ENGINEERING

The main argument to Knowledge Management in software engineering is that it is a creative and knowledge intensive activity. Software development is a process where every person involved has to make a large number of decisions and individual knowledge has to be shared and leveraged at a project and organization level, and this is exactly what KM proposes. People in such groups must collaborate, communicate, and coordinate their work, which makes knowledge management a necessity. In software development one can identify two types of knowledge: Knowledge embedded in the products or artifacts, since they are the result of highly creative activities and Meta-knowledge, that is knowledge about the products and processes. Some of the sources of knowledge (artifacts, objects, components, patterns, templates and containers) are stored in electronic form.

However, the majority of knowledge is tacit, residing in the brains of the employees. A way to address this problem can be to develop a knowledge sharing culture, as well as technology support for knowledge management. There are several reasons to believe that knowledge management for software engineering would be easier to implement than in other organizations: technology is not be intimidating to software engineers and they believe the tools will help them do a better job; all artifacts are already in electronic form and can easily be distributed and shared; and the fact that knowledge sharing between software engineers already does occur to a large degree in many successful software collaborative projects (Rus and Lindvall, 2002).

3 AGILE MANIFESTO

A new group of software development methodologies has appeared over the last few years. For a while these were known as lightweight methodologies, but now the accepted term is Agile methodologies. The most common of them are: eXtreme Programming, the Crystal Family, Agile Modeling, Adaptive Software Development, Scrum, Feature Driven Development, Dynamic System Development Method (Fowler, 2001). There exist many variations, but all of them share the common principles and core values specified in the Agile Manifesto (Chau and Maurer, 2004). Through this work they have come to value individuals and interactions over processes and tools. Working software over comprehensive documentation. Customer collaboration over contract negotiation. Responding to change over following a plan. These new methods attempt a useful compromise between no process and too much process, providing just enough process to gain a reasonable payoff. The result of all of this is that agile methods have some significant differences with the former engineering methods (Fowler, 2001): Agile methods are adaptive rather than predictive and they are people oriented rather than process oriented.

Most agile methodologies assume that change is inevitable, these methodologies have the ability to address variance and adaptability within the processes. In (Highsmith and Cockburn, 2001) Highsmith and Cockburn have fixed the role of creativity in agile teams assuming a world view that organizations are complex adaptive systems. A complex adaptive system is one in which decentralized, independent individuals interact in selforganizing ways, guided by a set of simple, generative rules, to create innovative, emergent results. Agile methods offer generative rules, a minimum set of things you must do under all situations to generate appropriate practices for special situations. A team that follows generative rules depends on individuals and their creativity to find ways to solve problems as they arise. Creativity, not voluminous written rules, is the only way to manage complex software development problems and diverse situations.

4 SOFTWARE CREATIVITY

There are many definitions of creativity, we use some ideas from (Franken, 2002): Creativity is defined as the tendency to generate or recognize ideas, alternatives, or possibilities that may be useful in solving problems, communicating with others, and entertaining ourselves and others. There are three reasons why people are motivated to be creative:

- need for novel, varied, and complex stimulation
- need to communicate ideas and values
- need to solve problems

In order to be creative, you need to be able to view things in new ways or from a different perspective. Among other things, you need to be able to generate new possibilities or new alternatives. Tests of creativity measure not only the number of alternatives that people can generate but the uniqueness of those alternatives. the ability to generate alternatives or to see things uniquely does not occur by change; it is linked to other, more fundamental qualities of thinking, such

as flexibility, tolerance of ambiguity or unpredictability, and the enjoyment of things heretofore unknown.

Toward an understanding of creativity in organizations, the use of a creativity management framework may be useful. Amabile (Amabile, 1996) had proposed a theory for the development of creativity. In her framework, creativity is hypothesized as a confluence of three kinds of resources:

- creativity-relevant skills (across domains)
- domain-relevant knowledge and skills (domain-specific)
- task motivation

Domain-relevant resources include factual knowledge, technical skills and special talents in the domain. Creativity-relevant resources include appropriate cognitive style, personality trait, conducive work style and knowledge of strategies for generating novel ideas. In specific, the major features of the appropriate cognitive style are the preference of breaking perceptual set and cognitive sets, keeping response options open, suspending judgment, etc. Furthermore, Amabile had proposed that intrinsic motivation was conducive to creativity; whereas extrinsic motivation was detrimental. Concerning the nurturing of intrinsic motivation, she and others highlighted the importance of promoting a playful attitude in the environment. Persons who are able to maintain playfulness, may continue to focus on the interest and enjoyment they derived from the task. They are more likely to keep their intrinsic motivation, even under external constraints.

The importance of creativity has been investigated in all the phases of software development process (Glass, 1995; Gu and Tong, 2004) and focused in the requirements engineering too (Robertson, 2005; Maiden and Robertson, 2005; Mich et al., 2005). Nevertheless, the use of techniques to foster creativity in requirements engineering is still shortly investigated. It is not surprising that the role of communication and interaction is central in many of the creativity techniques. The most popular creativity technique used for requirements identification is the classical brainstorming and more recently, role-playing-based scenarios, storyboard-illustrated scenarios, simulating and visualizing have been applied in an attempt to bring more creativity to requirements elicitation. These techniques try to address the problem of identifying the viewpoints of all the stakeholders (Mich et al., 2005). However, in requirements engineering the answers do not arrive by themselves, it is necessary to ask, observe, discover, and increasingly create requirements. If the goal is to build competitive and imaginative products, we must make creativity part of the requirements process. Indeed, the importance of creative thinking is expected to increase over the next decade (Maiden and Gizikis, 2001).

The industrial revolution replaced agriculture as the major economic activity, and then information technology replaced industrial production. Now, the information technology will be replaced with a new dominant economic activity focusing on creativity: The Conceptual Age. According to (Pink, 2005) we are moving from High Tech to High Touch and High Concept. The skill of storytelling is now a mandatory business skill. The workers in highest demand will be those with great social skills and a strong drawing portfolio. With the prevalence of search engines, facts are abundant and free, what is in demand now is the ability to put those facts in order and in context. The shift of IT organizations toward the creative sector and companies striving to design innovative products that combine and use existing technologies in unanticipated ways is beginning to justify this prediction.

In (Robertson, 2005; Robertson, 2002) very interesting open questions are proposed: Is inventing part of the requirements activity? It is if we want to advance. So who does the inventing? We cant rely on the customer to know what to invent. The designer sees his task as deriving the optimal solution to the stated requirements. We cant rely on programmers because theyre too far removed from the clients work to understand what needs to be invented. Requirements analysts are ideally placed to innovate. They understand the business problem, have updated knowledge of the technology, will be blamed if the new product doesnt please the customer, and know if inventions are appropriate to the work being studied. In short, requirements analysts are the people whose skills and position allows, indeed encourages, creativity.

In (Boden, 1990) the author, a leading authority on cognitive creativity, identifies basic types of creative processes: exploratory creativity explores a possible solution space and discovers new ideas, combinatorial creativity combines two or more ideas that already exist to create new ideas, and transformational creativity changes the solution space to make impossible things possible. Then, most Requirements Engineering activities are exploratory, acquiring and discovering requirements and knowledge about the problem domain. And the Requirements Engineering practitioners have explicitly focused on combinatorial and transformational creativity.

5 CONCLUSIONS

This paper was focused on some aspects of Knowledge Management and Creativity in the context of Optimization Algorithms development. The development of Optimization Algorithms is a field well suited for creative studies, since it is a creative activity where the problems often can only be solved through an iterative process faciliting Knowledge Management and exploration of new ideas.

Agile methods emphasis on people, communities of practice, communication, and collaboration in facilitating the practice of sharing tacit knowledge at a team level. They also foster a team culture of knowledge sharing, mutual trust and care. Agile development is not defined by a small set of practices and techniques. Agile development defines a strategic capability, a capability to create and respond to change, a capability to balance flexibility and structure, a capability to draw creativity and innovation out of a development team, and a capability to lead organizations through turbulence and uncertainty. They rough out blueprints (models), but they concentrate on creating working software. They focus on individuals and their skills and on the intense interaction of development team members among themselves and with customers and management.

The agile principles and values have recognized the importance of collaboration and interaction in the software development team. Because creative work commonly involves collaboration the study of techniques to foster creativity in software engineering is very interesting. Agile process to be helpful to generate novel and useful product. On the contrary, the discipline based work are perceived to be useless to produce novel products. The difference between them is that creative work can motivate the generation of something new.

Software development is a creative and knowledge intensive process that involves the integration of a variety of business and technical knowledge, an understanding from a Knowledge Management perspective offers important insights for designing and implementing Optimization Algorithms and Metaheuristics.

REFERENCES

Amabile, T. M. (1996). *Creativity in Context: Update to the Social Psychology of Creativity*. Westview Press.

Boden, M. (1990). *The Creative Mind*. Abacus.

Chau, T. and Maurer, F. (2004). Knowledge sharing in agile software teams. In Lenski, W., editor, *Logic versus Approximation: Essays Dedicated to Michael M. Richter on the Occasion of his 65th Birthday*, volume 3075 of *Lecture Notes in Artificial Intelligence*, pages 173–183. Springer.

Fowler, M. (2001). The new methodology. Available at http://www.martinfowler.com/articles/newMethodology.html.

Franken, R. E. (2002). *Human Motivation*. Thomson Learning College.

Glass, R. L. (1995). *Software creativity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Gu, M. and Tong, X. (2004). Towards hypotheses on creativity in software development. In Bomarius, F. and Iida, H., editors, *PROFES*, volume 3009 of *Lecture Notes in Computer Science*, pages 47–61. Springer.

Highsmith, J. and Cockburn, A. (2001). Agile software development: the business of innovation. *Computer*, 34(9):120–127.

Maiden, N. and Gizikis, A. (2001). Where do requirements come from? *IEEE Softw.*, 18(5):10–12.

Maiden, N. and Robertson, S. (2005). Integrating creativity into requirements processes: Experiences with an air traffic management system. In *13th IEEE International Conference on Requirements Engineering (RE 2005), 29 August - 2 September 2005, Paris, France*, pages 105–116. IEEE Computer Society.

Mich, L., Anesi, C., and Berry, D. M. (2005). Applying a pragmatics-based creativity-fostering technique to requirements elicitation. *Requir. Eng.*, 10(4):262–275.

Nonaka, I. and Takeuchi, H. (1995). *The Knowledge Creating Company*. Oxford University Press.

Pink, D. (2005). *A Whole New Mind: Moving from the Information Age to the Conceptual Age*. Riverhead Hardcover.

Robertson, J. (2002). Eureka! why analysts should invent requirements. *IEEE Softw.*, 19(4):20–22.

Robertson, J. (2005). Requirements analysts must also be inventors. *Software, IEEE*, 22(1):48–50.

Rus, I. and Lindvall, M. (2002). Knowledge management in software engineering. *IEEE Software*, 19(3):26–38. Available at http://fc-md.umd.edu/mikli/RusLindvallKMSE.pdf.

Vidal, V. V. (2005). Creativity for operational researchers. *Investigacao Operational*, 25(1):1–24.