Requirements Engineering Process Model for Informal Structural Domains

Karla Olmos, Jorge Rodas, and Luis Felipe Fernández

Abstract—**Tacit knowledge generates ambiguous, inappropriate and incomplete requirements, so it is extremely important do a good job on gathering meaningful requirements in order to obtain tailored high quality software. In this paper, a requirements engineering process model is presented, which has the aim of minimize the percentage of ambiguous, incomplete and not appropriate requirements and thus impact on software-solution quality, especially in domains where tacit knowledge has great relevance.**

Index Terms—**Requirements engineering, informal structured domains, tacit knowledge.**

I. Introduction

According to Easterbrook, "Requirements Engineering is a set of activities concerned with identifying and communicating the purpose of a software-intensive system, and the contexts in which it will be used. Hence, RE acts as the bridge between the real-word needs of users, customers, and other constituencies affected by a software system, and the capabilities and opportunities afforded by software-intensive technologies" [1].

This definition is interesting because reflect that a software system is an abstraction of a machine that interacts with a domain, as an organization, a physical or a technical world. The intersection between the phenomena of the machine and the world generates three concepts that should be distinguished: requirements (things in the world that stakeholders would like to achieve), specifications (descriptions of what the system being designed should do if it is to meet the requirements) and programs (descriptions of the properties and behaviours that, ultimately, satisfy customers) [2].

Under this perspective, Easterbrook classifies the types of software according to the interconnection of the software system to human activity. On one side of the spectrum are applications such as compilers and operating systems, with little interaction with human activities. For this type of system, requirements engineering has little or nothing to say due to the stable functionality of these devices. On the other side of the spectrum are software systems with a complex interaction with human activities, such as information systems or office automation systems. The development of this kind of system changes the system of human activity in which they are embedded and therefore, to understand human activities is essential. Easterbrook introduces the term of Software-Intensive System to describe this kind of systems that incorporate hardware, software and human activities, in which the requirements engineering is a key factor for the successful development of the project. However, there are other types of systems, such as Informal Structured Domains (ISD) (section 2), in which it is not enough consider the human activities but the knowledge of the people involved in the domain.

This paper introduces a Requirements Engineering Process Model for Informal Structured Domains. The process model considers the application domain and includes formally to the domain specialists, i.e. customers and users, to validate the partial results. The aim of the Process Model is to minimize the percentage of ambiguous, incomplete and not appropriate requirements and thus impact on software-solution quality. In Section 2, it will explain the Informal Structured Domains and its impact on Requirements Engineering. In Section 3, a Requirements Engineering Knowledge Flow Model will be introduced, which is part of the proposed process model. In Section 4, the original process model will be introduced. Finally, discussion and future work will be presented in Section 5.

II. Informal Structured Domains

A RE process always looks for a software solution attached to reality. That is why it is preferable to work with the highest amount of explicit knowledge. Several proposals to Requirements Engineering focus on systematic, refined and structured methods, such as Tropos [3], KAOS [4] and Techne [5]; in Domains where requirements are extremely clear and tacit knowledge has not enough relevance, these proposals have been effective. However, critical expectations, knowledge and needs of the stakeholders could frequently remain hidden, and the specification and its software-solution will not be adequate with users and/or customers' needs. So, it is generated additional cost and development time. Thus, it is required systematic means to make explicit the tacit knowledge (as much as possible) [6], especially if the Domain is an Informal Structured Domain (ISD) [7]. When a domain is ISD, it is important take into account the following features:

- Not all concepts and their relationships are formally defined, but rather these definitions tend to be based on consensus.
- Solutions of the problems in these Domains are diverse, consensus and unverifiable, and there are no algorithms to reach these solutions.
- To obtain the solution of a problem, specialists generally build a partial structure with the explicit knowledge. Thus, large amounts of tacit knowledge are always required to get an acceptable solution.

Manuscript received July 25, 2012; revised September 20, 2012.

The authors are with the Centro de Ingeniería de Software e Ingeniería del Conocimiento, Universidad Autónoma de Ciudad Juárez (e-mail: kolmos@uacj.mx, jorge.rodas@uacj.mx, lfernand@uacj.mx).

To develop software in Software-Intensive Systems it is necessary adopt a human-centered design, which makes human activities the focus of the design process. In ISD, it should be considered, not only the human activities, but the human knowledge, because of the great quantities of tacit knowledge that should be externalized and because of the development of this kind of system evolutes the knowledge of all the people embedded in the project. But working with human knowledge is not a trivial task, because it is personal, contextual and limited to the perspective and the role that the humans have about a domain. Namely, domain knowledge is dispersed. Moreover, according to Polanyi [8], this knowledge can be explicit or tacit. The first is related to theoretical knowledge, facts and other elements on which people are aware when thinking. On the other hand, tacit knowledge refers to personal and context-specific knowledge, which is hard to formalize and communicate.

Several authors have made proposals to minimize the effects of tacit knowledge. Nonetheless, they are incomplete or cost and time consuming. Nowadays, tacit knowledge in RE is an open research problem. Pital and Kaindl [9] propose a new perspective of RE based on knowledge management. Although the authors are aware that the problem of share knowledge in RE is not new, they suggest that this perspective offers specific insights and techniques for understanding and facilitating knowledge transfer and transformation. This paper agrees with Pital and Kaindl's perspective but goes further by proposing a new process model.

III. Requirements Engineering Knowledge Flow Model

Nonaka proposes a model of conversion of knowledge in organizations based on Polanyi's definition of tacit knowledge [10] and [11]. For Nonaka, knowledge creation in an organization is the result of social interaction of tacit and explicit knowledge. The model of Nonaka postulates four iterative conversion modes: *Socialization* (process of transferring tacit knowledge between individuals, by sharing mental models and technical skills), *Externalization* (process of converting tacit knowledge to explicit through the development of models, protocols and guidelines), *Combination* (process of recombining or reconfiguring existing bodies of explicit knowledge to create new explicit knowledge) and *Internalization* (process of learning by repetition of tasks and applying explicit knowledge).

In this paper, a Knowledge Flow Model for Requirements Engineering based on Nonaka's work is proposed. The model takes into account the following features of RE when it is applied to an ISD:

- There is symmetry of ignorance between requirements engineers and domain specialists, i.e. customers and users.
- It requires an arduous work of negotiation.
- The requirements engineer generates models to understand the properties of the domain and the system, and to obtain the specifications.
- According to Easterbrook clients and users are the only ones who can validate the system [1].

For the above, four states in a RE process were identified:

- Domain Knowledge Eduction (DKE). The objective of this stage is that requirementsengineer educes domain knowledge. In this stage, the *socialization* mode predominates.
- Model Generation (MG). In this stage, requirements engineers use the domain knowledge acquired in the stage of Domain Knowledge Eduction, her own knowledge of the machine and her experience to generate models. This is a complex activity in which *combination* and *externalization* modes are presented. In addition, as the requirement engineers develop models they *internalize* the domain knowledge.
- Model Discussion (MD). In this stage the models developed by requirements engineers will be discussed with the domain specialists. This phase takes place through *socialization*.
- Model Validation (MV). In this stage the domain specialists validate the generated models. To develop this activity they must *internalize* the knowledge behind models. To validate the model, a negotiation between the stakeholders is required. This process leads to the eduction of new domain knowledge, and then the cycle starts again.

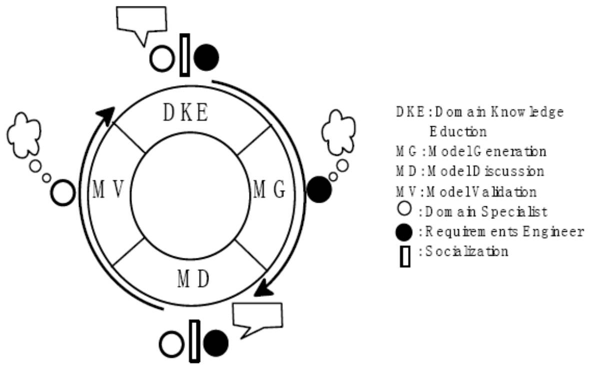

Fig. 1. Requirements engineering knowledge flow model.

On the bases of these states, a Knowledge Flow Model was generated. The Fig. 1 depicts this model which is iterative. The domain specialist is represented by an empty circle. The requirements engineer is represented by a full circle. Every socialization process is represented by these two circles separated by an empty rectangle.

IV. A Requirements Engineering Process Model for ISD

In this section, it is proposed a process model of Requirements Engineering with the aim to address the problems caused by the intrinsic characteristics of ISD's. The process model follows a system perspective that consists of three phases:

- Domain Modeling Phase (DM). In an ISD, the definition of concepts is ambiguous; thus, the first step of the process aims to understand the Domain properties. The externalization of this knowledge will enable to achieve a consensus among the stakeholder (even between the customers and users), to minimize the symmetry of ignorance and to minimize the loss of knowledge in the subsequent stages of software development.

- System Modeling Phase (SyM). In an ISD, understanding the software system and its effect on the application domain takes on special importance because of the characteristics of the activities, which depends on the context, i.e. is situated. It is important to formalize the system processes; therefore, the clients and users can assess or improve their own understanding of the domain.
- Software Modeling Phase (SwM). In this is the last phase the artefacts developed in previous phases will be used to derivate the document of specification.

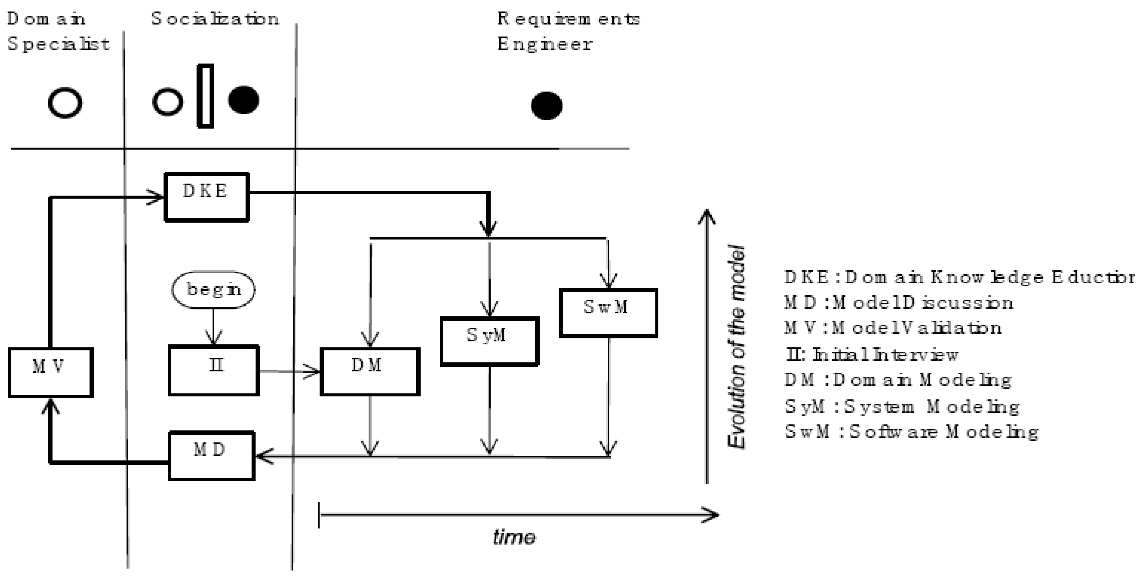

Fig. 2. Requirements engineering process model for ISD.

To face the challenges of the ISD these phases are combined according to the Knowledge Flow Model, as can be observed in Fig. 2. The swimlanes in the Fig represent the activities developed by the actors. In the first one are the activities developed by the domain specialist, i.e. customers and users. In the second one are the activities developed manly by a socialization process. In the last one are the activities developed by the requirements engineering, i.e. the model generation. It is considered that every project begins with an Initial Interview (II). In an ISD, the formulation of the problem and its solution evolves in parallel; therefore, the process model is evolutionary: as more knowledge is gained about the domain, the requirements engineer can return to earlier stages to refine the artefacts. That implies that the artefacts could be developed in parallel.

V. Discussion and Future Work

This article joins others in placing the tacit knowledge as the backbone for new proposals that facilitate the knowledge transition between people involve in a project. This paper also introduces an original Requirements Engineering Process Model. This process was developed on the bases of knowledge conversion model of Nonaka. The aim of the process model is to incorporate systematically the domain specialists in the process. In addition, our process model requires that the requirements engineer share with the domain specialists the models of the system to close the symmetry of ignorance.

As future work, it is necessary to apply this process to cases of study to verify their effectiveness and to improve it. In the same way, it is being developed an application to automate some tasks of the process.

References

[1] S. Easterbrook, Requirements Engineering Course. Universityof Toronto, 2005. [Online]. Available: http://www.cs.toronto.edu/~sme/CSC2106S/

[2] A. V. Lamsweerde, "From Worlds to Machines," in: B. Nuseibeh, P. Zave, P. (eds.) *Software Requirement and Design: the Work of Michael Jackson*, New Jersey: Good Friends Publishing. 2009, pp. 655-662.

[3] V. Bryl, P. Giorgini, and J. Mylopoulos, in: A. Ghose, G. Governatori, R. Sadananda (eds.) *Agent Computing and Multi-Agent Systems.* Lecture Notes in Artificial Intelligence, vol. 5044, Springer-Verlag, Berlin, Heidelberg. 2009, pp. 243-254.

[4] A. van Lamsweerde, "Requirements engineering: from craft to discipline," in *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering* (SIGSOFT '08/FSE-16). New York, NY, USA: ACM. 2008, pp. 238-249.

[5] I. Jureta, A. Borgida, N. Ernst, and J. Mylopoulos, "Techne: Towards a new generation of requirements modeling languages with goals, preferences, and inconsistency handling," in *the 18th IEEE International Requirements Engineering Conference (RE '10). Washington: IEEE Computer Society*. 2010, pp. 115-124.

[6] R. Gacitua, L. Ma, B. Nuseibeh, P. Piwek, A. N. de Roeck, M. Rouncefield, P. Sawyer, A. Willis, and H. Yang, "Making tacit requirement explicit," in *the Second International Workshop on Managing Requirements Knowledge (MARK '09). Atlanta: IEEE Computer Society*. 2009, pp. 40-44.

[7] K. Olmos, J. Rodas, and L. F. Fernández, "Pertinencia de la formalización de dominios semi-formalmente definidos en el Análisis inteligente de datos," *CULCyT: Cultura Cient fica y Tecnológica*, vol. 40-41, pp. 71—93, 2010.

[8] M. Polanyi, "The tacit dimension," *Routledge and K. Paul Press*, 1967.

[9] L. Pilat and H. Kaindl, "A knowledge management perspective of requirements engineering," in *the Fifth International Conference on Research Challenges in Information Science (RCIS)*, 2011, pp. 1-12.

[10] I. Nonaka and H. Takeuchi. *The Knowledge Creation Company.* Oxford University Press, 1995.

[11] I. Nonaka, "Tacit knowledge conversion: Controversy and advancement in organizational knowledge creation theory," *J. Organizational Science*, vol. 20, no. 3, pp. 635—652, 2009.