Acquiring Empirical Knowledge to Support Intelligent Analysis of Quality-Related Issues in Software Development

Vladimir A. Shekhovtsov, Heinrich C. Mayr, Christian Kop
Application Engineering Group, Institute of Applied Informatics,
Alpen-Adria-Universität Klagenfurt, Klagenfurt, Austria
shekvl@yahoo.com, {heinrich, chris}@ifit.uni-klu.ac.at

***Abstract*— The paper describes the current state of the initial part of the ongoing project aimed at the intelligent support for dealing with quality-related information in the software process. We describe the empirical qualitative studies aimed at acquiring the knowledge on stakeholder perception of quality and quality-related stakeholder interactions. This knowledge is supposed to be incorporated into a common ontology serving as a conceptual foundation for the prospective intelligent issue monitoring and analysis system based on the semantic repository. This system can be used to support prediction of the quality-related behavior of the stakeholders and facilitate reuse of the relevant knowledge.**

***Keywords*- software quality, software process, quality-related interactions, ontology engineering, qualitative techniques**

I. INTRODUCTION

It is not possible to organize successful software processes without involving the respective business stakeholders throughout the software development lifecycle – starting from its early stages. The common prerequisite for such involvement is establishing communication channels between the parties in the software process (from both customer and the developer side). In particular, it is important to establish such channels to carry the information on the quality of the software under development (SUD).

Unfortunately, organizing such kind of involvement still remains an open problem mainly because of the difficulties in establishing the necessary channels, especially carrying quality-related information. This difficulty can be largely attributed to the fact that the parties in software process speak different languages while talking about quality and it usually takes a long time to agree on a common vocabulary. Often the business people refuse to talk about the quality before they experience the implemented system – but in this case it could be too late or too expensive to fix the problem. Also, the reusability of the quality-related solutions and the possibility to predict the participants' behavior in future quality-related interactions remain low.

We propose to address this problem by establishing the intelligent support for dealing with quality-related issues (in the same sense as used in the issue-tracking systems such as JIRA [1] or Mantis [2]) in the software process. It involves collecting rich descriptions of such issues into a semantic repository and using this information to predict the reaction of the parties to the future issues of similar kind or to facilitate coming to the common language by these parties – aimed at maintaining good quality-carrying communication channels between the parties in the software process. This is a goal for the ongoing *QuASE* project [22, 23] established in cooperation with two local software development companies.

This paper is devoted to describing the QuASE project as a work in progress. Its current stage aims at acquiring and formalizing knowledge about software quality as a result of carrying on empirical studies involving the representatives of both the partner companies and their customers with an ultimate purpose of incorporating this knowledge into a common ontology. This ontology is supposed to serve as a source of rich semantic information to be incorporated into the semantic repository to be used for facilitating intelligent analysis and prediction of quality-related issues.

The paper is structured as follows. Section 2 outlines the solution to be developed as a result of QuASE project and the goal of its current stage of knowledge acquisition. Section 3 describes the research activities conducted so far by defining the categories of knowledge to acquire and the empirical studies involved. Section 4 describes the related work; it is followed by conclusions and the description of the future work directions.

II. INTELLIGENT ISSUE ANALYSIS SUPPORT: AN OVERVIEW

The schematic overview of the intelligent issue analysis solution to be implemented as a result of the QuASE project is provided on Fig.1. This solution includes the issue registration functionality (supported by the *Registration Engine*) and analysis functionality (supported by the *Analysis Engine*). It relies on the *Semantic Issue Repository* for storing and retrieving the knowledge about past quality-related issues. All the rich semantics to supplement this knowledge is defined in the common ontology (*QuOntology*).

A. Registration and analysis engines

The Registration Engine is aimed at storing the information about quality-related issues accompanied with rich semantics provided by QuOntology. In particular, all the semantics of quality-related negotiations is supposed to be stored together with all relevant properties of the involved parties, projects, qualities etc. Also, this system should allow registering *interaction solutions* aimed at interacting with business stakeholders (prototypes, mockups, interactive simulations etc.): after the registration, stakeholders should be able to interact with these solutions through the *assessment support* interface of the proposed solution and make all their reactions and quality assessments registered into the repository together with the stimuli provided by the registered interaction solution.

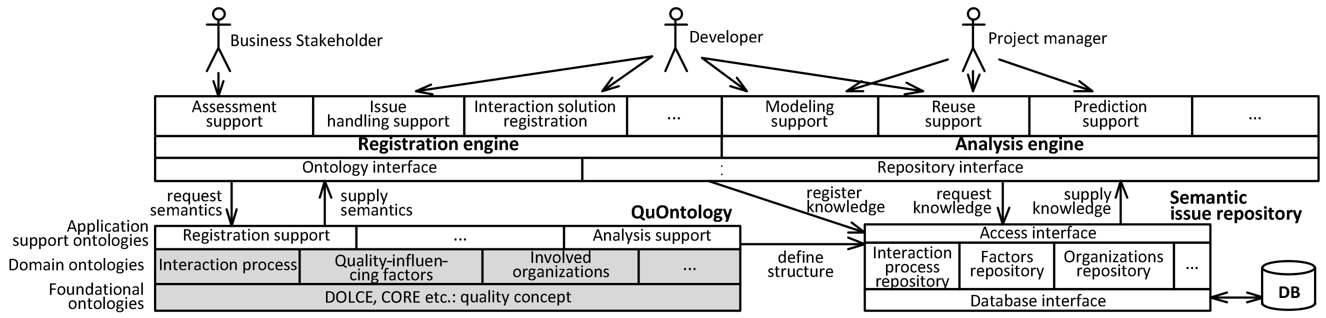

Figure 1. Intelligent support for quality-related issues

The Analysis Engine interacts with the repository to perform analysis of the past quality-related issues. In particular, it should allow: *aggregating the information* about issues to make useful generalizations and conclusions, *facilitating reuse* of specific issue-related information (e.g. if the encountered issue matches some historical data), *supporting prediction* of the behavior of the parties depending on the similarities between current issue and the historical data, *modeling* the behavior of the involved parties (e.g. the reaction of business stakeholders to the stimuli) or the prototypes with the specific properties.

B. QuOntology structure and its role in the project

QuOntology is defined on three levels (Fig.1). On the *foundational level*, we rely on existing formal ontologies with necessary modifications. For this purpose, we choose to reuse certain concepts from DOLCE ontology [15] with modifications related to the specific ways of representing quality-related information [18]. As DOLCE relies in its representation of quality on the notion of *conceptual space* per Gärdenfors [10, 19] we plan to investigate the additional ways of applying this notion and its modified forms [17] for our purpose. More details about this level are in [23].

The foundational level serves as a conceptual foundation of the *domain level*, which collects notions specific for the domain of quality-related stakeholder interaction. We propose to collect these notions in empirical studies; also, we plan to reuse specific ontologies such as those for people roles [7, 16], organizations [6, 8], and requirements [12].

On the *application level*, following Ontology-Based Software Engineering paradigm [11], QuOntology exposes itself through a set of application-level ontologies supporting the specific development tasks e.g. registration and analysis.

C. QuOntology engineering.

QuOntology engineering activities begin with empirical studies gathering essential knowledge about the topic of the study (the knowledge acquisition step). This knowledge is then incorporated into the ontology following the steps of ontology conceptualization, implementation, and validation with such activities as knowledge acquisition, ontology validation, and ontology documentation being performed iteratively on every step of the process. Current stage of the project deals with acquiring the knowledge for the foundational and domain levels of QuOntology (emphasized on Fig.1)

III. Knowledge acquisition

A. Acquisition methodology

To gain the necessary knowledge, we established the following qualitative studies in cooperation with industry partners: 1) *detailed interviews* questioning the partners' representatives of different backgrounds, in total, more than 20 hours of interviews with developers, project managers, and top management of the company were performed; 2) *post-mortal* and *documentation analysis* of the existing projects to gain the knowledge about their properties of interest (among the analyzed documents were meeting minutes as registered in issue tracking system at the company site, project documentation of different kind); we also plan to perform *observations* evidencing how the interactions between stakeholders and developers take place;

The obtained qualitative data after its transcription is supposed to be the subject of open and selective coding and concept analysis activities as defined for grounded theory [4, 9, 20]. We defined a hierarchy of codes reflecting the categories of knowledge to be acquired through the qualitative analysis (enumerated below). As a result, the set of concepts to be included into QuOntology is to be obtained.

B. The categories of knowledge to be acquired

We acquire the following categories of knowledge about quality-related phenomena of interest in the established empirical studies:

1. The phenomenon of software quality, different types of such quality
2. The categories of stakeholders participating in the interaction process, the ways of their selection
3. The differences in perception of quality for different categories of stakeholders
4. Quality-related stakeholder interaction process as it currently performed in industry
5. The properties of software projects and software under development which influence the practices of obtaining stakeholder opinions on SUD qualities
6. The factors influencing stakeholder opinions on quality during the negotiation process
7. The factors that influence software qualities that can be produced by the developers and serve as their arguments during the negotiation process

8. The ways of producing values of quality characteristics to be proposed to stakeholders
9. Possible real-world contexts for observing SUD quality, the ways of their selection
10. The ways of collecting and reusing the information about quality-related issues, including assessment data (stakeholder opinions on quality)
11. Real-world non-functional requirements and the ways of their elicitation
12. The ways of integrating quality-related information into software process activities

In this section, due to the space restrictions, we describe in detail the acquisition activities for only first five knowledge categories. For every category, we include the descriptions of both the empirical knowledge to be obtained and the process of conceptualizing this knowledge. Detailed treatment of the rest of the categories will be the target for the subsequent publications.

C. Category 1. The phenomenon of software quality

1) Obtained empirical knowledge

Out the particular categories for stakeholders, we interviewed the developers and project managers to figure out 1) their understanding of quality of the prospective systems; 2) the quality characteristics which they encounter most often, which are most important in projects and which are most likely to cause conflicts with stakeholders e.g. later in the development lifecycle; 3) the qualities the stakeholders usually encounter and their understanding of these qualities. In addition, the project managers were asked about 4) their opinions about neglecting the quality issues in projects.

2) Conceptualizing software quality

The obtained knowledge about the phenomenon of quality is planned to support the choice of formal ontologies to serve as a foundational level of QuOntology. The current definition of quality has been published in [21, 23] and is being developed further. In establishing these definitions, we rely on the efforts of establishing the notion of software quality and its usage in terms of the formal ontology [14] e.g. represented by DOLCE and, more specifically, by Core Ontology for Requirements Engineering (CORE) [12].

D. Category 2. SUD Stakeholders

1) Obtained empirical knowledge

We interviewed the developers to figure out 1) human stakeholders they work with; 2) the organizations employing these stakeholders; 3) which categories of stakeholders are interested in which qualities; 4) the characteristics of the stakeholders according to their ability to understand the quality in more IT-oriented way; 5) the characteristics of the domain and IT knowledge possessed by stakeholders and developers. The project managers and top management are, in addition, interviewed about 6) the possibility to select the people to work with at the customer side, the quality criteria for their selection, 7) the quality criteria for assigning the developers to the particular projects.

2) Conceptualizing SUD stakeholders

The obtained knowledge about existing stakeholder categories (including both parties in the development process: customers and developers) can be useful by itself to better understand people's needs with respect to the quality-related issues. In a process of incorporating this knowledge into QuOntology, we plan to merge it with the knowledge from the established ontologies of organizational roles [7, 16]. The knowledge about the organizations participating in the development process is planned to be combined with the knowledge originated from established organizational ontologies [6-8] before being incorporated into QuOntology in a process of conceptualization.

E. Category 3. The differences in perception of quality

1) Obtained empirical knowledge

We interviewed the developers to learn about their opinions on 1) the differences between their view on quality and the managers' view, 2) the differences in perception of quality between business stakeholders and IT persons and 3) the practices of resolving these differences. In addition, the project managers and top management people (e.g. CEO) are interviewed about 4) the differences between their view on quality and the view on quality possessed by the developers and business stakeholders.

2) Conceptualizing differences between quality views

The obtained knowledge about the differences in the perception of quality between different categories of stakeholders is conceptualized by introducing the notion of *quality realm* defined as the particular subdomain in the software engineering domain uniformly influencing the treatment of the software quality for all corresponding activities and concepts. We distinguish *user satisfaction realm* and *implementation realm*. Concepts belonging to the former represent user-centered view on the software quality: it employs a conceptualization of quality as it understood by business stakeholders. Concepts from the latter represent the view on quality possessed by the software developers.

F. Category 4. The stakeholder interaction process

1) Obtained empirical knowledge

We interviewed both developers and project managers about 1) the typical quality-related interaction with business stakeholders, 2) the roles of the people involved in such process, 3) the quality-related changes usually made in a course of the negotiations with a customer. Project managers, in addition, were asked about 4) the properties of the communication process and communication channels important for the success of the project (e.g. involved people, the way of establishing the channel etc.); 5) the current practice to ensure that both sides in a stakeholder interaction process are learning about the capabilities of each other and 6) the properties of the stakeholder interaction process that are important for the success of the project.

2) Conceptualizing stakeholder interaction

The preliminary results of this stage of research have been already published in [23]. We described the interaction process at two levels: coarse-grained level defining basic interaction activities (preparation, negotiation, agreement, implementation, checking) and fine-grained level defining these activities in detail. We also conceptualized the levels of SUD quality related to the particular activities of this process.

G. Category 5. The properties of software projects and SUD

1) Obtained empirical knowledge

We interviewed the developers to figure out 1) their view at characteristics of the projects they participated in, 2) the architectural solutions applied, the influence on quality characteristics of the software and the attitude to quality of the stakeholders and developers by the application of the particular solutions. Project managers' and top management were additionally asked about 3) the process of defining the scope of the project, the arguments used to keep the project in scope and to prevent the uncontrolled scope changes.

2) Conceptualization of projects and SUD properties

The obtained knowledge defines the important properties of software projects and the architecture of the SUD that could influence the process of obtaining stakeholder opinions on SUD qualities. An example of such properties could be the degree of how much the software process is stakeholder-driven or the way of breaking the SUD architecture into quality-bearing units (e.g. software services or components).

IV. Related Work

We consider the related works for the project activities of knowledge acquisition and conceptualization.

We refer to our paper [21] for a detailed literature review of the available quality conceptualization techniques. These techniques [12, 13]. are not completely suitable for our problem as they concerned with conceptualizing the quality itself without connections to the software process activities.

Process conceptualization techniques [3, 5], on the other hand, miss the notions of quality related to software process activities. In particular [5] proposes grounded theory-based conceptualization of the software development activities aimed at reaching common understanding between the parties in this process. We consider our work as an extension of such or similar generic conceptualizations specifically targeting quality-related communications.

V. Conclusions and Future Work

We described the project aimed, at its current stage, at acquiring empirical knowledge about dealing with quality-related issues in a software development with a purpose of incorporating this knowledge into the common ontology. This ontology establishes a common ground for organizing the knowledge about quality-related issues into the semantic repository – a knowledge base. This information could be reused directly while organizing future interactions; it also could be used to predict the behavior of the parties while encountering new quality-related issues. The proposed solution also facilitates coming to a common language for different parties in the software process by providing the set of quality-related concepts to be utilized by all these parties while participating in quality-related interaction activities.

The next stage of the empirical studies is devoted to interviewing the business stakeholders about their perception of quality. After completing the knowledge acquisition activities, the next stages of the ontology engineering process are supposed to be carried out, followed by establishing the issues repository and other parts of the solution.

References

[1] (28.05.2012). JIRA Issue Tracking System. Available: http://www.atlassian.com/software/jira

[2] (28.05.2012). Mantis Bug Tracker. Available: http://www.mantisbt.org

[3] S. T. Acuna and M. I. Sanchez-Segura, Eds., New Trends in Software Process Modeling. World Scientific, 2006.

[4] S. Adolph, W. Hall, and P. Kruchten, "Using grounded theory to study the experience of software development," Empir Software Eng, vol. 16, pp. 487–513, 2011.

[5] S. Adolph and P. Kruchten, "Reconciling Perspectives: How People Manage the Process of Software Development," in AGILE'11: IEEE, 2011, pp. 48-56.

[6] J. P. A. Almeida and E. C. S. Cardoso, "On the Elements of an Enterprise: Towards an Ontology-Based Account," in SAC'11: IEEE, 2011, pp. 323-330.

[7] G. Boella and L. van der Torre, "A Foundational Ontology of Organizations and Roles," in Declarative Agent Languages and Technologies IV: Springer, 2006, pp. 78-88.

[8] E. Bottazzi and R. Ferrario, "A Path to an Ontology of Organizations.," in VORTE'05, G. Guizzardi and G. Wagner, Eds., 2005, pp. 9-16.

[9] G. Coleman and R. O'Connor, "Investigating software process in practice: A grounded theory perspective," The Journal of Systems and Software, vol. 81, pp. 772–784, 2008.

[10] P. Gärdenfors, Conceptual Spaces: A Geometry of Thought. Cambridge, MA: MIT Press, 2000.

[11] W. Hesse, "Ontologies in the Software Engineering process," in Proc. EAI'05, Ceur-WS.org, vol. 141, 2005.

[12] I. Jureta, J. Mylopoulos, and S. Faulkner, "A core ontology for requirements," Applied Ontology vol. 4, pp. 169-244, 2009.

[13] M. Kassab, O. Ormandjieva, and M. Daneva, "An Ontology Based approach to Non-Functional Requirements Conceptualization," in Proc. 4th International Conference on Software Engineering Advances (SEA'09) New York: IEEE Press, 2009, pp. 299-308.

[14] C. Masolo and S. Borgo, "Qualities in formal ontology," in Proc. Workshop on Foundational Aspects of Ontologies (FOnt 2005), P. Hitzler, C. Lutz, and G. Stumme, Eds., 2005, pp. 2-16.

[15] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari, The WonderWeb Library of Foundational Ontologies. WonderWeb Deliverable D18. Ontology Library (final). Trento: ISTC-CNR, 2003.

[16] C. Masolo, L. Vieu, E. Bottazzi, C. Catenacci, R. Ferrario, A. Gangemi, and N. Guarino, "Social Roles and their Descriptions," in KR'2004, D. Dubois, C. A. Welty, and M.-A. Williams, Eds.: QuASEI Press, 2004, pp. 267-277.

[17] F. Probst, "Observations, Measurements and Semantic Reference Spaces," Applied Ontology, vol. 3, pp. 63-89, 2008.

[18] F. Probst, "Ontological Analysis of Observations and Measurements," in GIScience'06, 2006, pp. 304-320.

[19] M. Raubal, "Formalizing Conceptual Spaces," in Proc. FOIS 2004: IOS Press, 2004, pp. 153-164.

[20] K. Sharmaz, Constructing Grounded Theory: Sage Publications, 2006.

[21] V. A. Shekhovtsov, "On the evolution of quality conceptualization techniques," in The Evolution of Conceptual Modeling, LNCS, vol. 6520, R. Kaschek and L. Delcambre, Eds. Berlin-Heidelberg: Springer, 2011, pp. 117–136.

[22] V. A. Shekhovtsov, H. C. Mayr, and C. Kop, "Stakeholder Involvement into Quality Definition and Evaluation for Service-Oriented Systems," in Proc. USER'12 Workshop at ICSE'12: IEEE, 2012, pp. 49-52.

[23] V. A. Shekhovtsov, H. C. Mayr, and C. Kop, "Towards Conceptualizing Quality-Related Stakeholder Interactions in Software Development," in UNISCON 2012, LNBIP: Springer, 2012, in press.