Towards Managing Understandability of Quality-Related Information in Software Development Processes

Vladimir A. Shekhovtsov and Heinrich C. Mayr

Institute for Applied Informatics, Alpen-Adria-Universität Klagenfurt, Austria
{Volodymyr.Shekhovtsov,Heinrich.Mayr}@aau.at

Abstract. Establishing common understanding between the parties in the software process is important for dealing with quality of the prospective software. This process is difficult to organize because the parties (especially, developers and business stakeholders) perceive quality based on different world views. To address this problem, we aim at a solution for managing understandability of quality-related information in the software process. This solution provides the set of understandability assessment activities (aimed at diagnosing problems with communicated terms not belonging to the view of the target party) and understandability improvement activities (aimed at resolving these problems by translating problematic terms between world views and providing necessary explanations). These activities are supported by a modular ontology incorporating available quality-related knowledge; particular configuration of the ontology modules describes the quality view of the involved party. The proposed solution is expected to reduce the time and effort for establishing a communication basis while discussing software quality, thus cutting costs and strengthening the mutual trust of the parties.

Keywords: understandability, software process, software quality, quality-related communicated information.

1 Introduction

To organize successful software development processes, it is necessary to involve the affected business stakeholders throughout the development lifecycle. Such involvement, however, cannot be organized without establishing common understanding between the parties in the process, such as software developers and business stakeholders. In particular, it is necessary to have such understanding while dealing with quality of the software under development at different lifecycle stages. If the parties fail to understand each other on this issue, they tend to postpone all quality-related communication activities until the later stages of the project (such as acceptance testing); this decision could significantly increase the related costs and effort.

Establishing such common understanding is problematic because the parties (in particular, software developers and business stakeholders) think in different conceptualizations of the real world and use different terminologies, especially when dealing with the quality of the software under development: agreeing on a common point of view is usually time-consuming and often fails.

B. Murgante et al. (Eds.): ICCSA 2014, Part V, LNCS 8583, pp. 572–585, 2014.

To address the above problem, we propose to elaborate a tool-supported framework aimed, in part, at supporting the process of establishing common understanding between the parties on the quality of the software under development; the important component of this framework is aimed at assessing and improving such quality characteristic as *understandability* of quality-related information to be communicated between parties. In this paper, we present the conceptual foundations and the implementation procedures for assessing and improving this characteristic; this research is being conducted as part of the ongoing QuASE project[1] established in cooperation with four local software development companies.

The paper is structured as follows. In Section 2, we establish a context for the presented research by defining the place of understandability management in the framework of the QuASE project. Section 3 provides the background information about understandability as quality characteristic and understandability conflicts; Section 4 introduces understandability management process, Section 5 introduces ontological support for understandability management, the procedures for understandability assessment and improvement are described in Section 6. Section 7 outlines possible usage scenarios for understandability management, Section 8 describes related work; it is followed by conclusions.

2 Background: QuASE Project

Research activities related to understandability management for quality-related information in the software process are performed in a course of the QuASE project. This project is devoted to the research and tool development aiming at improving the process of quality-related communication between parties in the software process.

In addition to understandability of information described in this paper, this process addresses *quality of decisions* based on communicated information, addressing this characteristic is supposed to be achieved, in particular: by issuing recommendations on the ways of conducting quality-related communications based on the analysis of the past experience of organizing such communications; it is also supposed to support these decisions by forecasting communication parameters and prospective outcomes. The goal of these recommending activities is to increase the parties' awareness of the communication context and the possible ways of action prior to and during the communication, lowering the effort necessary for coming to the right decisions w.r.t. implementing the communication.

Addressing understandability and quality of decisions is supposed to be implemented based on knowledge-oriented access interface to the communication-related data (representing communicated information) collected in industry software development projects which is supposed to be implemented as a part of QuASE software solution. This interface can be exemplified as follows:

---

[1] The QuASE Project is sponsored by the Austrian Research Promotion Agency (FFG) in the framework of the Bridge 1 program (`http://www.ffg.at/bridge1`); Project ID: 3215531.

1. Communication-related data in the industrial software development projects is kept in the project repositories, in particular, those controlled by issue management systems (IMS) such as JIRA [8]. To implement knowledge-oriented access interface it is proposed to provide an access to the data available in such databases through an ontology (*QuOntology*) capturing the concepts related to the domain of quality-related communications between parties in the software process; this ontology is supposed to contain not only the generic domain concepts, but also the concepts specific for the particular development contexts.
2. The provided interface includes the set of concepts and their attributes extending the structure of the data available in project repositories (referred to as *primary project information*) with *semantic annotations*, represented by additional concepts, concept attributes, and inter-concept relationships necessary for addressing the above quality characteristics. In particular, such *annotation information* can correspond to the factors influencing the attitude to software quality possessed by the parties in the software process, and, as a result, to the process of these parties' decision making in a course of communication.

3 Background: Information Understandability

3.1 Understandability: A Definition

In selecting understandability as the quality characteristic to be addressed in QuASE project we follow ISO/IEC 9126 quality model [6] (which defines understandability as a quality characteristic for software artifacts) by extending it to the case of quality-related communicated information in the software process: understandability is defined as "*the capability of the quality-related communicated information to enable the target party to understand its meaning and follow procedures defined therein exactly as intended by the originating party*".

To elaborate on this, we follow the work of Adolph et al [1] who investigated the process of negotiating different perspectives in software development; in a course of this investigation, they defined the concept of *perspective mismatch*. According to this paper, "the inability to get everyone on the same page is a significant impediment to getting the job done; we referred to the source of these impediments, created by differing points of view to getting the job done, as a perspective mismatch".

Based on this concept, it is possible to state a draft operational definition for understandability which could be the source for defining the respective metrics: "*understandability of the quality-related communicated information is defined as a reciprocal of the perspective mismatch between the originated and the target party revealed from this information.*" Such mismatch can be measured as the distance between the point of view expressed in the particular piece of quality-related information and the point of view of the target party. As we defined understandability as a reciprocal, the higher the information understandability, the shorter has to be this integrated distance.

3.2 Understandability Conflicts

In classifying the understandability conflicts, we also follow Adolph et al [1] who define the following categories of such conflicts:

1. *Translation-inducting conflicts* are related to the situations when only the terminology differs, but perspectives are aligned i.e. people are talking about the same things and share the common communication goals;
2. *Broadening-inducting conflicts* are related to the situations when it is necessary to broaden the understanding of the job by trying to understand the other's point of view; these conflicts involve aligning the perspectives (views on quality) as precautions. An example of such conflict could be the situation when the communicated information is explained from the business point of view but not only with different terminology but with different e.g. quality view; e.g. business consequences of the particular decisions are not explained;
3. *Scouting-inducting conflicts* refer to the cases when neither side can express the perspective to the other so it is necessary for both sides to acquire additional information to perform this task; the problem with such conflicts is that sometimes the sides are not able to do so which could lead to endless cycles in the software process;

These three categories of conflicts serve as a motivation for QuASE understandability research. We aim at the following activities:

1. addressing translation-inducting conflicts by directly supporting the terminology translation with a goal of reducing the effort necessary to perform this task;
2. reducing the impact of broadening-inducting conflicts by supplementing the information with the necessary explanations broadening the perspective (e.g. supplementing particular technical terms with the appropriate explanations related to their influence to business);
3. eliminating or reducing the number of scouting-inducting cases and translate such cases into the cases of broadening-inducting conflicts by supplying the sides with the necessary information helping to explain their perspective to the other side.

3.3 Research Goals

The problem of addressing the above conflicts for the specific case of understandability of quality-related communicated information in the software process leads to establishing the following research goals:

1. Adapt the existing systems of categories of understandability conflicts such as defined in [1] to the current case of the understandability of quality-related communicated information in the software process;
2. Establish and evaluate understandability quality subcharacteristics (assessment criteria) and relevant metrics for quality-related communicated information in the software process;

3. Define and implement the procedures for revealing the prospective understandability conflicts based on quality-related communicated information in the software process and the appropriate criteria to assess the applicability of these procedures;
4. Define and implement the procedures for resolving the understandability conflicts based on quality-related communicated information in the software process and the appropriate criteria to assess the applicability of these procedures;

4 Understandability Management Process: An Overview

The above goals are planned to be addressed by the specific *understandability management process* comprised of the following activities:

1. *understandability assessment activities* aimed at revealing the prospective understandability conflicts; these activities evaluate the understandability for the particular pieces of quality-related communicated information and provide the relevant recommendations;
2. *understandability improvement activities* aimed at resolving the understandability conflicts; these activities aim at maximizing the understandability criteria related to target parties for the particular pieces of quality-related communicated information.

Establishing these activities is based on the auxiliary activities addressing research goals 1 and 2 as the former rely on the classification system of the understandability conflicts and the appropriate set of quality subcharacteristics and quantitative metrics.

5 Ontological Support for Understandability Management

The support for information understandability management in QuASE solution is based on implementing the access to the communicated information (whose understandability is being managed) through the modular ontology (QuOntology) providing the capabilities of translating between world views. Initial research on QuOntology has been published in [15], whereas [13] aims at presenting the current version of the relevant conceptualizations.

The structure of QuOntology is depicted on Fig.1; it includes the following three layers:

1. QuOntology core;
2. Domain ontology layer;
3. Context ontology layer.

QuOntology core represents a stable subset of the knowledge available as a result of research and industrial practice; the knowledge represented in core does not depend on the particular problem domain and the particular context. We use Unified Foundational Ontology (UFO) [4, 5] as a foundation for QuOntology core.

Domain ontologies represent the specifics of the particular problem domain which is addressed by the particular software under development (finance, banking, oil and gas etc.), the concepts from domain ontologies extends base concepts represented in QuOntology core [5]. Predefined domain ontology is *the ontology of quality harmonization* which includes the set of concepts specific for the domain of quality-related negotiations in software engineering; supplying additional domain ontologies is a separate task which should be completed while adapting QuASE solution to support development for the particular domain (it is planned to supplement the particular configuration of the QuASE tool with the set of domain ontologies).

Context ontologies represent the knowledge depending on particular components of the knowledge context; here we define the knowledge context as a set of all concepts connected to the particular communication process (organization, organization type, project, project type, stakeholder, stakeholder type, etc.), in particular, it is possible to distinguish:

1. Context ontology defined for the particular organization type;
2. Context ontology for the particular organization;
3. Context ontology for the particular project type;
4. Context ontology for the particular stakeholder category.

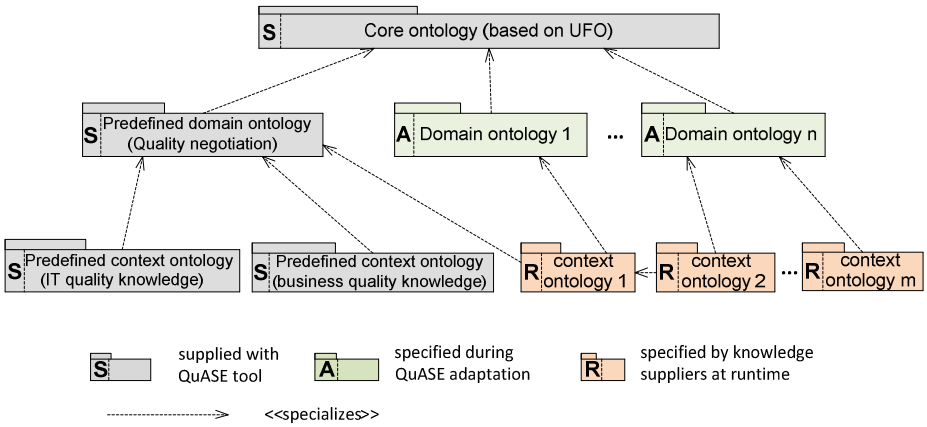

Fig. 1. QuASE ontology layers

All the concepts represented in context ontologies must extend the generic concepts represented in QuOntology core and the domain ontologies; in addition, it should be possible to extend the concepts represented other context ontologies. Two predefined context ontologies are planned to be implemented: *IT quality ontology*: the ontology representing common knowledge about quality possessed by IT people and *business quality ontology* representing common knowledge about quality possessed by business stakeholders; the additional context ontologies are supposed to be supplied by involving the experts (knowledge suppliers) at runtime.

6 Understandability Management Process: Activities

6.1 Text-Based Semantic Annotation

Both understandability assessment and understandability improvement activities rely on the common auxiliary activity aimed at *annotating primary information* (available in the project repositories) based on the additional knowledge provided by the QuOntology or explicitly specified by the user.

We refer to *semantic annotation* as to an activity of extending the set of attributes of quality-related communicated information or the set of its additional connected concepts with a set of attributes and concepts (*annotation information*) important for reaching particular goals of the QuASE solution.

In *text-based semantic annotation*, the values for the additional attributes of quality-related communicated information and the data for the additional connected concepts are extracted from natural language texts available in project repositories (in particular, such texts can represent issue descriptions).

Text-based semantic annotation is performed in two stages (Fig.2):

1. Analyzing natural language text with a purpose of recognizing and tagging the terms to be annotated;
2. Connecting the tagged terms to the hierarchies of concepts taken from the context ontologies (a particular configuration of such ontologies defines the knowledge context for the given piece of communicated information); as a result, the annotated text should be represented as a knowledge structure supplemented by the set of alternative viewport-related concept hierarchies.

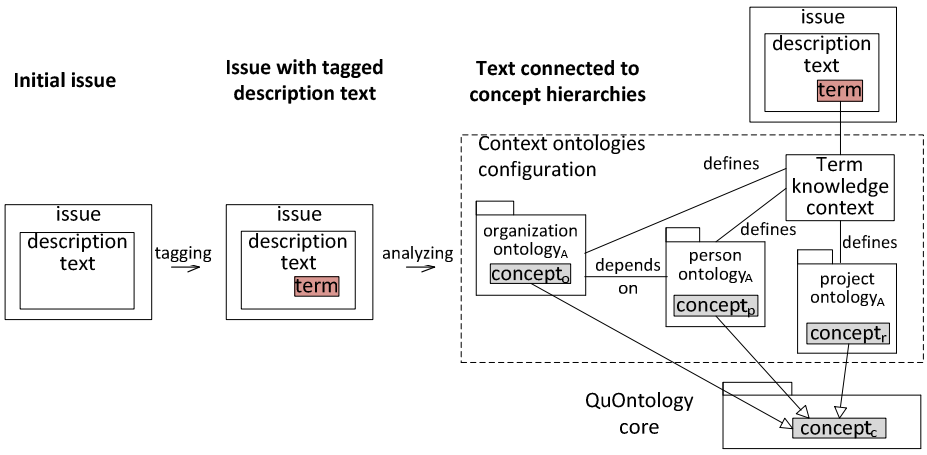

Fig. 2. Text-based semantic annotation (domain ontology layer is omitted, adapted from [14])

The results of text-based semantic annotation (annotation information) are used by information understandability management activities (both understandability assessment and understandability improvement); the specifics of this usage are outlined in the following sections.

6.2 Understandability Assessment

Understandability assessment activities involve assessing the particular piece of quality-related communicated information (e.g. represented by the annotated issue data available in the project database controlled by the issue management system) for the presence of understandability problems and the applicability of the particular understandability improvement techniques.

In particular, in a course of understandability assessment the following activities could be performed:

1. Calculating the distance between the current piece of communication information and the stored perspective information;
2. Reporting the distance and the categories of mismatch;
3. Forming the recommendations for performing further understandability improvement activities.

Particular recommendations could be related to the fact that in some situations (e.g. while dealing with a broadening-inducting problem) only the explanations are necessary to be provided and in other situation (i.e. while dealing with a translation-inducting problem) the terminology translation could be possible.

Additional research activities related to the understandability assessment could be, in particular, exemplified as follows:

1. Investigating the properties of the particular piece of communicated information which makes its translation feasible;
2. Investigating what makes a particular natural language text a good text from the point of view of its understandability for other parties and what makes a text a bad one;
3. For the case of IMS issues, investigating what makes a particular issue taken as a whole (as a set of attributes and natural language texts) a good issue from the point of view of its understandability for other parties;
4. Investigating the characteristics of the amount of effort to be applied to the particular text to reach the particular target related to its understandability.

6.3 Understandability Improvement

The goal of understandability improvement procedures is minimizing the distance between the current piece of quality-related communicated information and the perspective of the target party (the *understandability gap*). The following preliminary activities could be performed beforehand:

1. extracting the perspective information from the project repositories and storing the extracted perspectives in the knowledge base;
2. calculating the distance between the current piece of information and the stored perspective (by the means of understandability assessment)

After that, it is necessary to minimize the distance by applying the understandability improvement (perspective mismatch resolution) procedure.

The generic improvement approach could be specified as follows

1. Finding the source of misunderstanding and separating this source from the rest of the information; for terminology translation case this could be exemplified by singling out the set of problematic terms;
2. Bringing the source of misunderstanding closer to the perspective of the target party; for terminology translation case this can be exemplified by applying the translation procedures to the separated misunderstanding source; the appropriate termination criteria have to be defined for this process;
3. Merging the separated parts back to obtain the adapted textual description.

The following research questions have to be addressed prior to implementing understandability improvement procedures:

1. How to define the knowledge structure representing the point of view of the particular party in the software process?
2. How to define the distance between the particular piece of quality-related communicated information (e.g. a particular issue) and the point of view of the particular party? What are the characteristics of this distance?

Translation-Based Understandability Improvement Procedure. For the particular case of translation-inducting understandability conflict, it is proposed to apply the following translation-based understandability procedure aimed at transforming issue-related information between "world views" of the communicating parties. This procedure has to be applied to the natural language text, it is also possible to have a scenario of converting the structured information, but the main goal of the transformation subsystem is related to dealing with natural language specifications and stakeholder opinions.

To support understandability improvement through terminology translation it is necessary to implement switching between knowledge contexts related to recognized terms; this allows translating between world views of different communicating parties. The understandability improvement procedure for the case of translation-inducting conflict is performed in two stages (Fig.3):

1. Resolving (by applying the ontological reasoner) the corresponding concepts defined in context ontologies into the generic concepts defined in QuOntology core or domain ontologies;
2. Switching to the target context, looking up the target context concepts corresponding to the generic concepts; these target concepts form the translation results.

On the first stage, it should be possible to run the ontological reasoner to establish the connection between the recognized term and generic party-independent knowledge. In particular, if the description contains the term "Oracle RDBMS" (IT specific) the analysis process with a help of the reasoner should be able to relate it to more neutral term (e.g. "data storage"). By looking for all the generic concepts it would be possible

to figure out all party-independent knowledge related to the particular issue description or opinion fragments; so the purpose of this improvement technique is to enable matching the independent concepts in QuOntology core to the concepts found in the particular description.

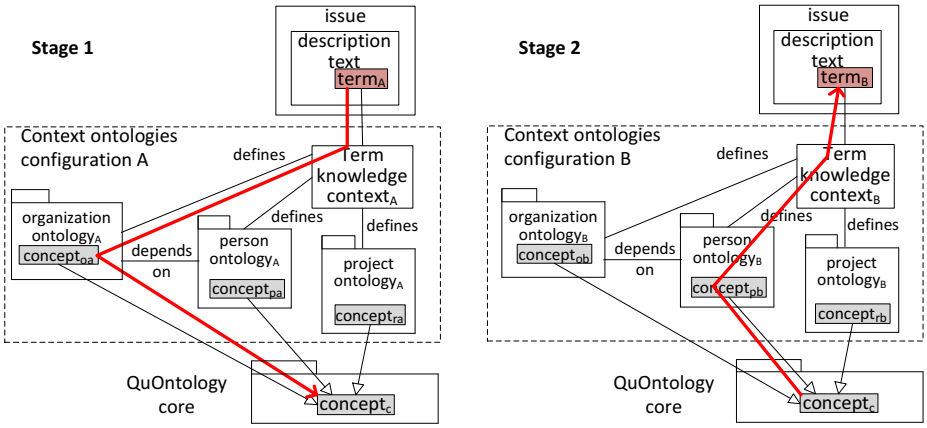

Fig. 3. Understandability improvement for the case of terminology conflicts

On the second stage, the different context ontology configurations should be used so the generic concepts have to be translated back to the context-specific concepts and then to corresponding target terms.

Explanation-Based Understandability Improvement Procedure. Besides the translation-inducting understandability improvement procedure, additional improvement techniques have to be specified to address the cases where the terminology translation is not sufficient. In particular, explanation-based improvement procedure has to be defined for the case of broadening-inducting conflicts; it has to supplement the problematic terms with extensive explanations aimed at support broadening of the particular point of view. To make able applying this procedure, the context ontology should contain explanations for concepts. A particular explanation could be defined:

1. in the same ontology as the concept to be explained: this reflects the situation when the target party directly knows about the particular concept, but needs additional information to understand it completely;
2. in a different related ontology at the same level (e.g. the ontology for other related context element): this reflects the situation when the target party can get the explanation for the particular term from the related context knowledge (e.g. the knowledge related to the relevant project or the involved organization);
3. in the ontology below the concept to be explained (Fig.4); this reflects the situation when the target party only knows about some generic definition of the concept, but needs additional target-specific information to understand it in necessary detail.

The procedure for explanation-based understandability improvement for the case when explanation is defined in the ontology below the concept to be explained is depicted on Fig.4. This procedure also involves switching between knowledge contexts and obtaining the necessary explanation from the target context.

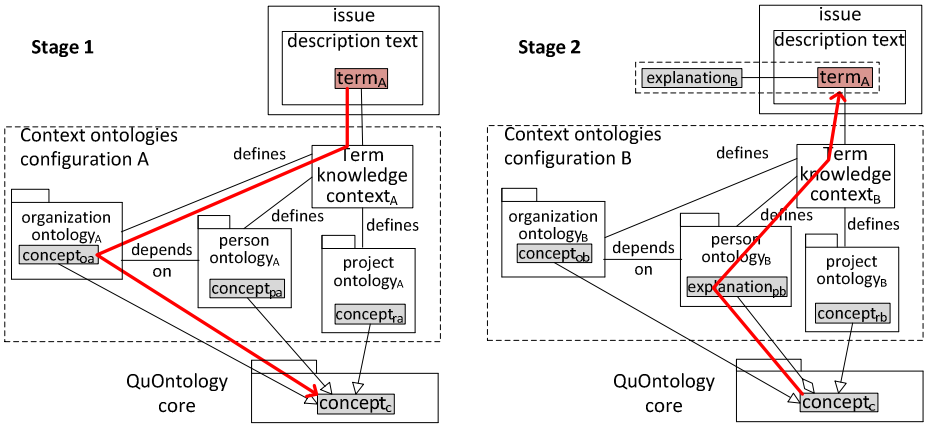

Fig. 4. Explanation-based understandability improvement

7 Usage Scenarios

In this section, we outline the usage scenarios for understandability assessment and understandability improvement for the case when communicated information is represented by issue descriptions, and the knowledge context is defined for particular business stakeholders.

Understandability assessment scenarios involve calculating understandability metrics for the source issues according for the given stakeholder's context, identifying the problems with the issue: showing the list of understandability problems for the given issue, and issuing recommendations for dealing with these problems.

For calculating understandability metrics the user has to:

1. select the source context (the original context for the issue e.g. its author) and the target context (the context to check the issue against e.g. the business stakeholder);
2. select the issue or the set of issues;
3. obtain the values characterizing the effort necessary to understand the issue in a target context (i.e. by the target stakeholder).

For diagnosing understandability problems for the particular issue, the user has to:

1. select source and target stakeholders and the set of issues as defined above;
2. obtain the set of problematic terms from the issue description text.

Besides understandability assessment scenarios, QuASE aims at supporting both translation-based and explanation-based understandability improvement scenarios. In translation-based scenario (Fig.5) the user has to:

1. select the issue or the set of issues or supplies the arbitrary text; Fig.5 reflects the situation when the arbitrary text has to be supplied;
2. select the source and target stakeholder;
3. obtain the version of the specified text with translated terms.

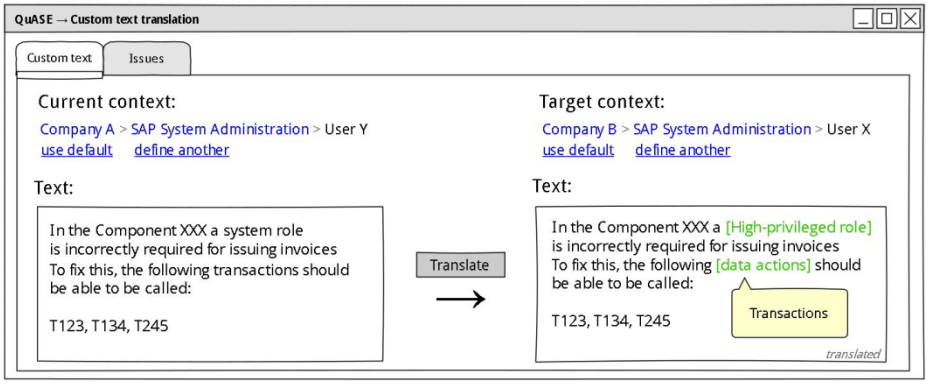

Fig. 5. Prototype UI for translation-based understandability improvement scenario

Explanation-based scenario entails performing the following activities (Fig.6):

1. selecting source and target stakeholders and the set of issues as defined above;
2. highlighting the problematic terms in the document and obtaining explanations for these terms.

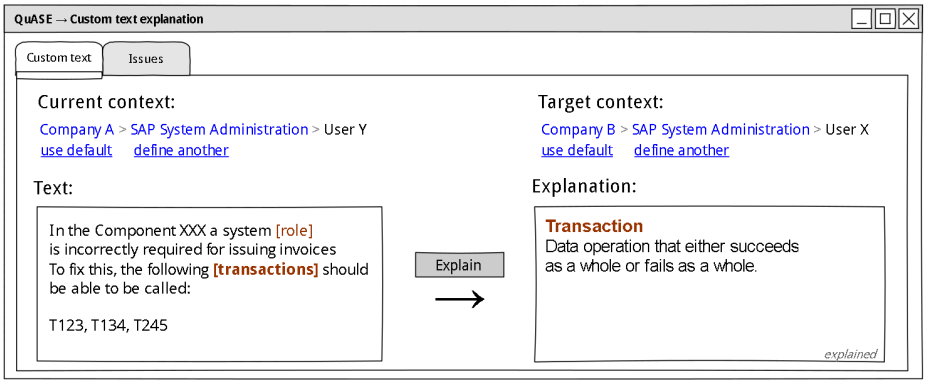

Fig. 6. Prototype UI for explanation-based understandability improvement scenario

8 Related Work

Current research addressing understandability of the information in the software process mainly deals with this quality characteristic defined for the following software process artifacts (we group these artifacts by the stage of the development process):

1. *Requirement engineering-related artifacts*: in particular, understandability of the requirement specifications is addressed in [9], whereas [2] deals understandability of the use case models;
2. *Design-related artifacts*: in particular, the set of metrics for measuring understandability of the conceptual models is defined in [11], understandability of entity-relationships diagrams is introduced in [3], and the set of factors influencing understandability of the business process models is outlined in [12];
3. *Implementation-related artifacts*: in particular, the set of metrics for source code understandability is defined in [7, 10];

The differences between our approach and the above techniques are as follows:

1. Most state-of-the-art techniques address understandability of the particular categories of development-related artifacts (such as requirements, source code, or conceptual models); our research, to the contrary, addresses understandability of the generic fragments of communicated information which could be contained in documents belonging to different categories; in this paper, these documents are exemplified by issues;
2. These techniques, as a rule, do not specifically address understandability of quality-related information;
3. They do not employ ontology-based approach for establishing common understanding between parties in the software process.

9 Conclusions and Future Work

In this paper, we presented a set of implementation procedures for ontology-based framework aimed at managing understandability of quality-related communicated information in the software process. This framework deals with information already available in project databases and aims at making the fragments of such information suitable to the view of quality possessed by the target party. It supports the processes of understandability assessment (aimed at diagnosing prospective problems) and understandability improvement (aimed at resolving such problems by bringing the information closer to the view of target party) and is based on the modular ontology aimed at representing the common knowledge to be communicated; the particular configuration of the ontology modules describes the knowledge on quality possessed by the particular understandability context such as the customer organization or the particular stakeholder; this ontology is applied to the raw issue data prior to communication. Establishing such framework allows us to address the problem of providing

the parties in the software process an easy way of adapting the information to the point of view of other parties.

We plan to continue our research by defining detailed implementation procedures and the tool support for understandability management activities; in addition, the ongoing research aims at establishing the structure of knowledge base (QuIRepository) mapping the ontological knowledge into the project database data.

References

1. Adolph, S., Kruchten, P., Hall, W.: Reconciling perspectives: A grounded theory of how people manage the process of software development. The Journal of Systems and Software 85, 1269–1286 (2012)
2. Anda, B., Jørgensen, M.: Quality and understandability of use case models. In: Lindskov Knudsen, J. (ed.) ECOOP 2001. LNCS, vol. 2072, pp. 402–428. Springer, Heidelberg (2001)
3. Genero, M., Poels, G., Piattini, M.: Defining and validating metrics for assessing the understandability of entity–relationship diagrams. Data & Knowledge Engineering 64, 534–557 (2008)
4. Guizzardi, G.: Ontological foundations for structural conceptual models. University of Twente (2005)
5. Guizzardi, G., Falbo, R., Guizzardi, R.S.: Grounding software domain ontologies in the unified foundational ontology (UFO): The case of the ODE software process ontology. In: Proceedings of the XI Iberoamerican Workshop on Requirements Engineering and Software Environments, pp. 244–251 (2008)
6. ISO/IEC 9126-1:2001: Software Engineering – Product Quality – Part 1: Quality Model. International Organization for Standardization, Geneva (2001)
7. Jin-Cherng, L., Kuo-Chiang, W.: A Model for Measuring Software Understandability. In: CIT 2006. IEEE (2006)
8. JIRA Issue Tracking System, `http://www.atlassian.com/software/jira` (accessed May 08, 2014)
9. Kamsties, E., von Knethen, A., Reussner, R.: A controlled experiment to evaluate how styles affect the understandability of requirements specifications. Information and Software Technology 45, 955–965 (2003)
10. Lin, J.-C., Wu, K.-C.: Evaluation of software understandability based on fuzzy matrix. Fuzzy Systems, 2008. In: FUZZ-IEEE 2008, pp. 887–892. IEEE (2008)
11. Mehmood, K., Cherfi, S.S.: Data quality through model quality: a quality model for measuring and improving the understandability of conceptual models. In: MDSEDQS 2009, pp. 29–32. ACM (2009)
12. Reijers, H.A., Mendling, J.: A study into the factors that influence the understandability of business process models. IEEE Transactions onSystems, Man and Cybernetics, Part A: Systems and Humans 41, 449–462 (2011)
13. Shekhovtsov, V.A., Mayr, H.C., Kop, C.: Harmonizing the Quality View of Stakeholders. In: Mistrik, I., Bahsoon, R., Eeles, R., Roshandel, R., Stal, M. (eds.) Relating System Quality and Software Architecture. Elsevier (in print, 2014)
14. Shekhovtsov, V.A., Mayr, H.C.: Managing Quality Related Information in Software Development Processes. In: CAiSE 2014 Forum. CEUR-WS.org (in print, 2014)
15. Shekhovtsov, V.A., Mayr, H.C., Kop, C.: Towards Conceptualizing Quality-Related Stakeholder Interactions in Software Development. In: Kop, C. (ed.) UNISON 2012. LNBIP, vol. 137, pp. 73–86. Springer, Heidelberg (2013)