Distinguishing Soft-Goals and Quality Requirements in Software Requirements Modeling

Thi-Thuy-Hang HOANG
Louvain School of Management (LSM),
Université catholique de Louvain,
Louvain-la-Neuve, Belgium.
E-mail: thi.hoang@uclouvain.be

Alain PIROTTE
Louvain School of Management (LSM), and
Information Technologies, Electronics and Applied Mathematics (ICTEAM)
Université catholique de Louvain,
Louvain-la-Neuve, Belgium.
E-mail: alain.pirotte@uclouvain.be

***Abstract* - Requirements engineering plays an essential role in software development. Requirements are prescriptive statements that express situations to be enforced by a system in terms of its effects on its environment. There have been many discussions of functional versus non-functional requirements, of hard-goals versus soft-goals, of non-functional and quality requirements versus soft-goals. Quality requirements have often been treated as special cases of non-functional requirements or of soft-goals, without a clearly convincing distinction. In this paper, we formulate a somewhat unusual definition and metamodel for "quality requirements" and we explore some of its consequences. Quality requirements are not defined as a special kind of soft-goals, but as constraints on goals. We adapt the usual techniques of goal refinement to our definition and we argue that thus distinguishing soft-goals and quality requirements contributes to clarifying the system development process and the management of quality by the resulting software products.**

Keywords - requirements models, data modeling, soft-goals, quality requirements, non-functional requirements

I. REQUIREMENTS, GOALS, AND QUALITIES

Requirements engineering plays an essential role in software development. Requirements are prescriptive statements to be enforced by a new software system or by a revised version of an already existing system, possibly in cooperation with other system components, and formulated in terms of its environment [10]. Requirements contribute to a global model of the system in its early stages of development or revision. Then they help construct a specification for system design.

Requirements are identified and captured from exchanges of information between analysts and stakeholders of the future or revised system. Requirement analysts play the role of architects who progressively transform informal needs of stakeholders into a precise and possibly formal model. Collecting and understanding stakeholder informal descriptions and performing the transformations from informal to precise models are hard creative tasks. Thus, requirements engineering remains a delicate phase of software development as widely documented in the literature (see, e.g., [10,11,13]).

Requirements that concern functions and services of the future or revised system (i.e., questions about what the system should do) are called functional requirements. Requirements that concern how well (e.g., quality aspects of how speedily (performance), how cheaply (costs), how accurately, etc.) the system should provide its functions are typically called non-functional requirements. Non-functional requirements are often also presented as quality requirements in the literature without a clear distinction between them.

A more modern view than the distinction between functional and non-functional requirements presents the system as responsible to bring about desired states of the world specified as goals. Goals have been typically classified into hard-goals and soft-goals depending on the precision of their satisfaction criteria. Hard-goals have satisfaction criteria that are clearly defined. Like functional requirements, they concern functionality. Soft-goals are goals whose satisfaction criteria could not been defined in a clear-cut manner when they were formulated or whose satisfaction can be subjective. They may be judged as satisfied or unsatisfied to different degrees by different people and at different stages of system development. The intuition about soft-goals can be conveyed by the following examples that we further discuss later: "all banking transactions must be handled in a secure manner" or "online banking transactions must be offered with high availability". For some reasons, not enough information was available when they were formulated as requirements to precisely assess their satisfaction criteria.

There have been many analyses of requirements in terms of functional versus non-functional requirements, of hard-goals versus soft-goals, of non-functional and quality requirements versus soft-goals (see, e.g., [5,7], that clearly illustrate the difficulties involved). There are similarities between non-functional requirements and soft-goals in their typical characteristics of defining satisfaction criteria in an imprecise manner. To argue for a difference between soft-goals and non-functional requirements, it has been suggested that non-functional requirements define constraints but not system functions, while soft-goals characterize system states and thus can describe directly or indirectly system functions. But this is more intuition than an exploitable definition. In multi-agent systems, where goals are used extensively for

 ISBN: 978-1-61208-185-4

modeling requirements, non-functional requirements are often considered as a subset of soft-goals.

Much thinking has been devoted specifically to software quality. The coexistence of several approaches to defining software quality has been well summarized in a recent book [10]. It is argued that there are three principal ways to view software quality: (1) conformance to requirements; (2) reliability, portability and other -ilities; and (3) absence of defects. The third approach privileges quantitative methods for defect detection and removal using quality metrics. For [10], qualities like the -ilities are not practical because they are too vague (e.g., survivability) and most of them are irrelevant for users (like, e.g., portability). Problems with the first approach can arise if some requirements are badly selected, i.e., "toxic", missing or excess requirements, that unintentionally cause requirement-compliant products to go wrong. Our approach is of course more related to the first definition (quality as conformance to requirements).

This paper does not aim at comparing various definitions. In particular, many approaches to quality attributes regard them as useful only when they can be measured quantitatively. We do not discuss either the nuances between non-functional requirements and quality requirements. Instead, we present our own definition and metamodel of quality requirements by which they are different from soft-goals, and they constrain hard-goals and soft-goals. We explore some interesting consequences of that definition by adapting the usual techniques for goal analysis and refinement so that they can take advantage of this additional dimension. This paper substantially revises the core ideas of [8] after the completion of the first author's thesis [9].

Broadly, model-driven engineering allows software developers to focus on concepts more abstract than those directly supported by implementation platforms. The approach has contributed (1) languages (or metamodels) to describe various aspects of systems and (2) correspondences between metamodels, and the operationalization of such mappings as techniques and tools to help system development tasks. Model-driven system development is thus conducted by building and composing models, and by transforming them progressively into operational systems.

This paper essentially deals with a proposal for a new metamodel for quality requirements and soft-goals. It also discusses the correspondence of goal-refinement graphs expressed in our metamodel with other representations of goals. Section 2 of the paper motivates our approach to quality requirements. Section 3 gives precise definitions. Section 4 presents the integration of our approach into an adapted strategy for goal analysis and refinement. Section 5 expands on fulfilling quality requirements.

II. SEPARATING QUALITY REQUIREMENTS AND SOFT-GOALS

The intuition of our approach can be grasped through the following examples of soft-goal requirements mentioned in Section 1: "all banking transactions must be handled in a secure manner" and "online banking transactions must be offered with high availability".

Qualifying terms, like "in a secure manner" or "with high availability", represent qualities (of security or of availability) that are embedded inside the soft-goal statements. If those qualifying terms are "taken out" of the soft-goals, then we argue that the same situation can be described by a hard-goal (for example, "all banking transactions must be treated") constrained by what we will call a quality requirement (for example, "in a secure manner", i.e., with some quality of security).

General issues arising from this discussion include the following: (1) to what extent can quality requirements be extracted from soft-goals? (2) how can their relationships be adequately expressed in a precise metamodel? (3) when they can, how does the separation benefit software development, especially how does it improve and simplify the fulfillment of requirements in both functional and quality aspects? (4) which tools could help software developers? Although the examples above suggest that the answer to the first question is affirmative, these issues deserve more analysis. This paper addresses the first and second issues and, briefly, the third one. Some tools are proposed in [9].

III. OUR DEFINITION OF QUALITY REQUIREMENTS

We first formulate our basic definition of quality requirement (together with definitions of hard-goal and soft-goal), where the novelty essentially lies in the fact that quality requirements are defined as constraints on goals and not as non-functional requirements. Then, after some comments and examples, we complement the basic definition by definitions of the links between goals and quality requirements that express their dependencies.

A **goal** describes a state that the system-to-be should be able to bring about. A **hard-goal** is a goal for which satisfaction criteria can be precisely defined. A **soft-goal** is a goal for which satisfaction criteria are not defined in a clear-cut way. A **quality requirement** describes a constraint whose satisfaction or fulfillment ranges on a scale of possibilities and that can constrain a goal.

Degrees of satisfaction or of fulfillment of quality requirements are typically expressed as abstract levels that range from "not satisfied" to "fully satisfied" (e.g., "high", "low", "average", "cheap", "expensive", "affordable" etc.).

The following examples illustrate the proposed definitions. "Payment sent" is a hard-goal since it describes a well-defined state (of having or not having been sent) leading to defining payment functionalities. "Money transferred with high security" is a soft-goal since the level of security is not precisely stated. "Web pages served with high availability" is also a soft-goal for similar reasons. In those examples, "with high security" and "with high availability" are quality requirements.

Concerning the choice of terms, using "quality requirement" for a constraint of a different nature than a non-functional requirement may not be the best idea. Just saying "quality" would not be a better choice because the term is overloaded. We could maybe have chosen "quality constraint". We stayed with "quality requirement" and we tried to be as clear as possible throughout the text to avoid ambiguities.

 ISBN: 978-1-61208-185-4

A reason why it is difficult to define satisfaction criteria for some soft-goals is the presence of quality requirements tightly integrated "inside the soft-goal", like, e.g., "message confidentially sent" or "web pages served with high availability". Such soft-goals can be called "quality soft-goals". They typically appear in early stages of requirements formulation. Quality requirements can be elicited (i.e., extracted) from them, like "confidentiality" from "message confidentially sent". Such a complex soft-goal can be represented as a quality requirement "high availability" that constrains a hard-goal "web pages served". Similarly, the soft-goal "money transferred with high security" can be understood as the combination of the hard-goal "money transferred" and of the quality requirement "high security".

Other soft-goals are not so explicitly linked with a quality requirement, like, e.g., "Increase Sales" or "Make Customers Happier". The goal refinement process described in Section 4 transforms such soft-goals into other goals (eventually into hard-goals) that will make qualities more visible. For example, a strategy to "Make Customers Happier" could involve sub-soft-goals like "Provide Reliable and High-speed Connections", which could result in the hard-goal "Provide Connections" constrained by quality requirements "Reliability" and "High-speed".

Our full metamodel of quality requirement combines the basic definitions above with the following definition of **links between soft-goals and quality requirements**: some soft-goals can be viewed as the combination of a goal (hard-goal or soft-goal) and a quality requirement; such soft-goals can be alternatively re-expressed as a combination of the derived goal constrained by the derived quality requirement. We call "qualification link" the link from a quality requirement to a constrained goal and "elicit link" the link from a soft-goal and a quality requirement extracted from it.

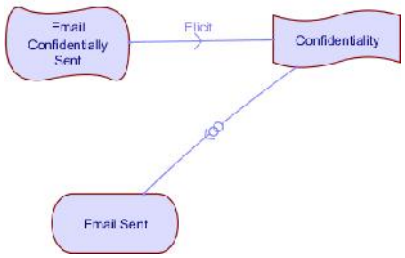

Figure 1: Quality requirement elicitation

Figure 1 shows an example of a soft-goal ("Email Confidentially Sent") re-expressed as a hard-goal ("Email Sent") constrained by a quality requirement ("Confidentiality") elicited from the soft-goal. We follow the familiar notations of [4] for goals: soft-goals are drawn as boxes with round-shaped vertical sides and wavy horizontal sides while hard-goals are drawn as boxes with straight horizontal sides and round-shaped vertical sides. We draw a quality requirement as a box with straight vertical sides and wavy horizontal sides. We draw a qualification link as an arrow joined with two small circles from a quality requirement to the goal that it constrains. We signal an elicit link by an arrow from the soft-goal to the elicited quality.

IV. GOAL REFINEMENT WITH QUALITY REQUIREMENTS

Section 4.1 recalls the basic ideas of goal refinement as a strategy for software development. Section 4.2 describes a modified strategy taking into account the new node type of quality requirements and the new types of link: elicitation, qualification, and contribution (the latter still to be defined). Section 4.3 summarizes the objectives of goal refinement.

4.1 Goal refinement

The main objective of goal-based requirements engineering, is to iteratively refine higher-level requirements until concrete system requirements are obtained. AND/OR decomposition graphs in KAOS [6,13] have become standard tools for such tasks. AND/OR decomposition is appropriate for hard-goals since they can be defined as logical conditions and their satisfaction can be checked by AND/OR relations between corresponding logical conditions of their sub-goals.

For soft-goals, additional weaker types of goal transformation are needed, as was suggested in the style of i* [1,4,12,14]. Derived soft-goals can contribute, negatively or positively, to fulfilling parent soft-goals. Such a contribution is made explicit by so-called contribution links that can take, in [4], one of five contribution levels: '++' (make), '+' (help), '?' (unknown), '-' (hurt) and '--' (break) with an obvious intuition. The idea is that such contributions links will allow application specialists to determine, for each candidate choice of system functions to implement lower-level goals, the satisfaction level of each top-level soft-goal, given the AND/OR contributions, and the satisfaction level of the derived goals and quality requirements.

4.2 Revised goal analysis

This section suggests an extended goal analysis process to accommodate quality requirements. Our definitions suggest to add quality requirement nodes and three new additional links to the usual AND/OR goal analysis. As introduced in Section 3, "elicitation" links describe how a quality requirement is extracted or inferred from a soft-goal in which it was "packaged" during the analysis of early requirements. Qualification" links relate quality requirements to goals on which they bear. "Contribution" links are used to describe how hard-goals can contribute (negatively or positively) to the satisfaction of quality requirements that constrain them. Quality requirements can themselves be refined into subqualities and they can propagate down in the goal decomposition tree.

4.2.1 OR decomposition

OR decompositions make explicit some alternatives to fulfill a goal in the goal-decomposition tree. In Figure 2, the goal "Invitation Sent" can be satisfied either by the goal "Invitation Sent By Email", or by the goal "Invitation Sent by Post", or by the goal "Invitation Communicated By Telephone". Since the "Promptness" quality is required of the parent goal, it is also required of all the alternatives.

 ISBN: 978-1-61208-185-4

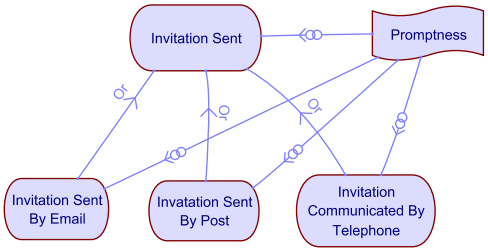

Figure 2: OR decomposition with quality requirement

This example can be analyzed differently with "Invitation Promptly Sent" packaged as a soft-goal. Figure 3 shows a version of the example where the "Promptness" quality requirement has been extracted not from the initial soft-goal "Invitation Promptly Sent", as Figure 2 suggests, but from its OR-subgoals "Invitation Promptly Communicated By Telephone". Of course, these are just illustrative examples and further analysis of the solution in Figure 3 may well evolve into that of Figure 2.

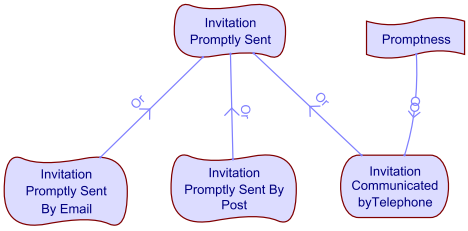

Figure 3: OR decomposition of Soft-goal

4.2.2 *AND decomposition*

With AND decomposition, a goal is satisfied if and only if its subnodes are satisfied. In Figure 4, the goal "Music Played" with quality "Legality" is satisfied if a relevant file is found and downloaded legally. Then the downloaded music file is opened to send sound to speakers. Hard-goal "Music File Opened" is not concerned with the "Legality" quality requirement since "Legality" can be completely satisfied by the other two hard-goals.

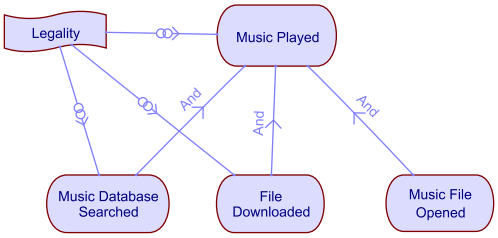

Figure 4: AND decomposition with quality requirement

Qualities can be decomposed into sub-qualities as suggested in [4]. Sub-qualities must be appropriately applied to sub-goals of an AND-decomposition, as illustrated in Figure 5. Quality "Economy" is decomposed into sub-qualities "Efficiency" and "Reusability". To have "Software Developed" with "Economy" (for the sake of the example), it must (i) be designed with "Reusability" in mind, and (ii) be provided with an efficient bug management system, supposing that the coding activities do not affect the overall development cost. This example illustrates the fact that the decomposition of quality requirements can be generic at higher levels of the analysis graph and application-dependent at more concrete lower levels.

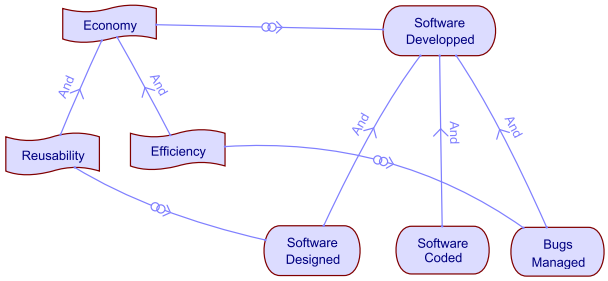

Figure 5: AND decomposition and extended quality requirement

4.2.3 *Contribution links and quality requirements*

Quality requirements are propagated through goal analysis and refinement. Contribution links can be used in the process to help take quality requirements into account.

Contribution links were introduced to help decompose soft-goals [4]. In i*, they are used only to decompose soft-goals. In our approach, they can also be used with quality requirements. The most common usage is to show how satisfying a hard-goal can contribute to satisfying a quality requirement that the hard-goal is required to fulfill.

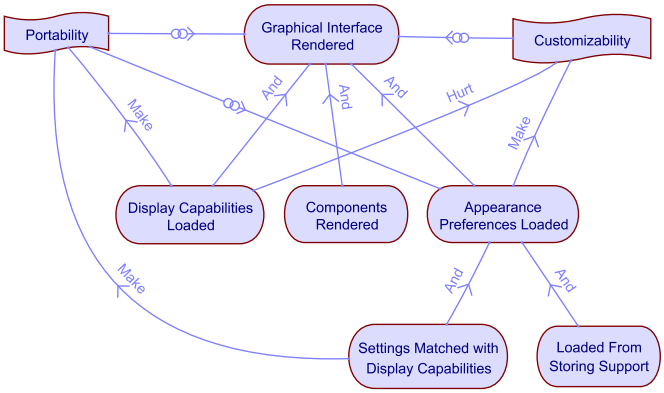

Figure 6: Contribution links and quality fulfillment

Figure 6 illustrates such a situation: a decomposition of hard-goal "Graphical Interface Rendered" constrained by quality requirements "Portability" and "Customizability". Hard-goal "Components Rendered" expresses that rendering the complete interface can be split into rendering its components. For quality"Portability" (to several interfaces like PC monitors, smartphones, etc.), hard-goal "Display Capabilities Loaded" requires that sufficient capabilities for those interfaces be made available. For quality "Customizability", hard-goal "Appearance Preferences Loaded" requires that the rendering system take into account existing user preferences for display settings (font face, color, font size, etc.). Contribution links of type "Make" (see Section 4.1) from the latter two hard-goals to their respective quality requirement make those contributions explicit.

In Figure 6, too much emphasis on the display capabilities of a specific platform may hamper the

"Customizability" of the overall platform, in the sense of making things more difficult for a device with insufficiently supported settings, a situation made explicit with a "Hurt" contribution link from "Display Capabilities Loaded" to "Customizability". The situation can be improved by further constraining the "Appearance Preferences Loaded" hard-goal with the "Portability" requirement. This new qualification link suggests the decomposition of "Appearance Preferences Loaded" into hard-goals "Settings Matched with Display Capabilities" and "Loaded from Storing Support". The idea is that "Settings Matched with Display Capabilities" will limit user settings to available capabilities and thus positively contribute to the "Portability" quality requirement. The analysis of the example should still be refined to further clarify and somehow sort the conflict between the contributions to quality requirements.

4.3 The objectives of goal analysis and refinement

The inputs to the qualitative process of goal refinement are a set of hard-goals expressing the important functional requirements for the system and a set of soft-goals in which quality requirements are more or less explicitly embedded.

The main objectives of goal analysis as presented here are: (1) to accompany developers and stakeholders when identifying and clarifying relevant requirements; (2) to organize the exploration and evaluation of alternative detailed requirements for the new or revised system; (3) to progressively make quality requirements explicit and to propagate the responsibility of satisfying them downwards to suitable goals of the analysis graph; (4) to "operationalize" those lower-level qualified hard-goals whenever possible, that is, to relate them to operations to be made available by the future system and that will ensure their best satisfaction.

Such system operations (or services or functions) invoked at the leaves of the goal analysis graph are intuitively similar to UML use cases, that is, specifications of operations that are expected to be provided by the system.

For example, a qualified goal "Message Sent" constrained by the quality requirement "High Confidentiality" could be adequately fulfilled by a system operation like "Send Encrypted Message".

V. FULFILLMENT OF QUALITY REQUIREMENTS

Figure 7 illustrates our techniques with a more substantial example. Soft-goal "Payment Immediately and Securely Sent" for an online shop is progressively transformed with elicitation links to extract quality requirements, qualification links to constrain derived goals, and contribution links to express contributions to the fulfillment of quality requirements and to the selection of goal decomposition alternatives.

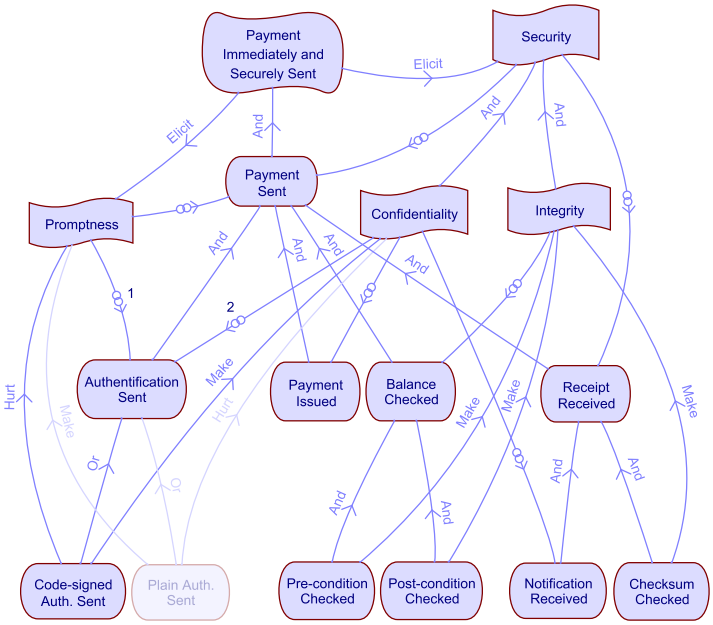

Figure 7: Immediately and Securely Sent soft-goal

Soft-goal "Payment Immediately and Securely Sent" is re-expressed as hard-goal "Payment Sent" constrained by elicited quality requirements "Security" and "Promptness". "Security" is AND-decomposed into "Confidentiality" and "Integrity" quality requirements. Hard-goal "Payment Sent" is AND-decomposed into hard-goals "Authentication Sent", "Payment Issued", "Balance Checked", and "Receipt Received". Hard-goal "Balance Checked" is AND-decomposed into hard-goals "Pre-condition Checked" and "Post-condition Checked". Hard-goal "Receipt Received" is AND-decomposed into hard-goals "Notification Received" and "Checksum Checked".

Hard-goal "Authentication Sent" is given two alternative subgoals: "Code-signed Auth. Sent" and "Plain Auth. Sent". The first one contributes positively ("Make") to "Confidentiality" and negatively ("Hurt") to "Promptness" as it requires user intervention for authentication, while the reverse holds for the second alternative. One way to solve the conflict, as suggested in Figure 7, is to assign a higher priority (say, value 2) to "Confidentiality" than to "Promptness" (value 1) thus privileging one of the subgoals.

It is interesting to compare the bottom part of Figure 7 to similar goal and soft-goal integration in the approach of [4]. There, soft-goals are mostly quality requirements; hard-goals and quality requirements are analyzed separately and correlated late in the analysis process. Our approach allows to analyze hard-goals, soft-goals, and quality requirements in an integrated scheme from the top-most and most abstract goals. If elicit and qualification links are removed from our analysis, then it becomes similar to the goal refinement of [4]. The price to pay for this simplification is the suppression of the traceability of quality requirements, analysis rationale, and late operationalization of quality requirements.

Thus, compared to modeling approaches relying on just hard-goals and soft-goals, our approach is semantically richer, in line with a model-driven emphasis. Quality

 ISBN: 978-1-61208-185-4

requirements are decoupled, as analysis and refinement proceed, from the functional part thanks to the additional links (elicitation, qualification, and contribution links).

The possible outcomes for an analysis graph is a list of alternative AND-sequences of leaf nodes together with the quality requirements whose satisfaction is still undecided. In Figure 7, there are two possible sequences of leaf hard-goals, each corresponding to one of the alternatives for handling the "Authentication Sent" hard-goal as described above. If "Confidentiality" is privileged as suggested, the sequence of leaf hard-goals is: "Code-signed Auth. Sent", "Payment Issued", "Pre-condition Checked", "Post-condition Checked", "Notification Received", "Checksum Checked", where two hard-goals ("Payment Issued" and "Notification Received") are still constrained by the "Confidentiality" requirement. Ensuring this quality requirement relies on third-party payment services. If not enough information is available, then handling it can be deferred until enough additional information becomes available.

In some cases, ensuring an acceptable degree of fulfillment must be postponed to run time and programmed with adhoc solutions. Such a late operationalization is typically necessary when the actual load of the system cannot be estimated with sufficient precision at the analysis stage. An example is the estimation by a service provider of an adequate frequency of checking the queue length of waiting customers for addressing the quality requirement of "High Availability". A set of social design patterns to help address such programming tasks is proposed in [9].

VI. SUMMARY AND CONCLUSIONS

This paper formulates a new definition and metamodel of quality requirements treated not as special cases of non-functional requirements or of soft-goals, but as constraints on goals, and it explores some of consequences of that definition. Three important aspects of our approach and its effect on goal refinement can be summarized as follows. First, quality requirements are typically embedded (i.e., implicit) within high-level soft-goals at the start of the analysis process. They are made progressively explicit during goal refinement. Second, with elicitation and qualification links, all soft-goals can be, earlier or later, re-expressed as a combination of lower-level hard-goals constrained by quality requirements. Third, quality requirements appear explicitly throughout the goal analysis process. As a result, quality requirements are more decoupled from the functional part, which enables a more flexible treatment of them.

The paper also proposes extensions of the usual process of goal refinement of software development by defining revised transformation of goal graphs in order to accommodate the presence of quality requirements that constrain goals and soft-goals.

Our approach was applied in [9] to enhancing the Tropos methodology [3,14] as QTropos (quality-aware Tropos) where quality requirements are independent notions constraining dependencies. Quality requirements can thus be taken into account during all phases of QTropos (early requirements, late requirements, architectural design, and detailed design). Also described in [9] are QCase, a tool to help apply our metamodel to the development of realistic projects, and a case study to help validation. Subsequent work will strengthen QCase with code generation and broaden its applicability to concepts from other development methodologies.

More perspectives for subsequent work include refining our metamodel at the light of substantial case studies and of further critical comparisons with other languages and methods, e.g., [1,2,12], and the abundant literature on quality in requirement engineering and model-driven development. Clearly, our approach to quality as constraints on goals enhancing conformance to requirements is just one possible approach among many. More work can be done on assessing more finely the locus and strength of its relevance.

REFERENCES

[1] D. Amyot, J. Horkoff, D. Gross, and G. Mussbacher. A Lightweight GRL Profile for i* Modeling, in Advances in Conceptual Modeling – Challenging Perspectives, Springer LNCS 5838, pp. 254-264, 2009.

[2] C. Ayala, C. Cares, J. Carvallo, G. Grau, M. Haya, G. Salazar, X; Franch, E. Mayol, C; Quer, A Comparative Analysis of i*-Based Agent-Oriented Modeling Languages, Procs 17th Int. Conf. on Software Engineering and Knowledge Engineering (SEKE'05), pp. 43-50, 2005.

[3] J. Castro, M. Kolp, and J. Mylopoulos. Towards Requirements-Driven Information System Engineering: the Tropos Project, Information Systems 27(6), pp. 365-389, 2002.

[4] L. Chung, A. Nixon, E. Yu, and J. Mylopoulos. Non-Functional Requirements in Software Engineering, Kluwer Academic Publishers, 2000.

[5] L. Chung and J.C.P. Leite, 2009. On Non-Functional Requirements in Software Engineering. Conceptual Modeling: Foundations and Applications – Springer LNCS 5600, pp. 363-379.

[6] R. Darimont and A. Van Lamsweerde. Formal Refinement Patterns for Goal-Driven Requirements Elaboration, ACM Sigsoft Notes, 21(6), pp. 179-190, 1996.

[7] M. Glinz. On Non-Functional Requirements. In: Proceedings of the 15th IEEE International Requirements Engineering Conference. IEEE, pp. 21-26, 2007.

[8] T. T.H. Hoang and M. Kolp. Goal, Soft-goal and Quality Requirement. In: Procs 12th ICEIS (Int. Conf. on Enterprise Information Systems)., pp. 11-22, 2010.

[9] T.T.H. Hoang. Quality-Aware Agent-Oriented Information-System Development. Ph. D. Thesis, Louvain, School of Management, Louvain-la-Neuve, Belgium, 2010. Available at http://www.isys.ucl.ac.be/ThesisHoangTTH.pdf

[10] C. Jones. Software Engineering Best Practices. McGraw-Hill, New York, 2009.

[11] R. Ramsin and R. Paige. Process-Centered Review of Object-Oriented Software Development Methodologies, ACM Computing Surveys, 40(1), pp. 1-89, 2008.

[12] 12. URN (User requirement Notation), Language Definition: Recommendation Z.151, International Telecommunication Union (ITU), Geneva, Switzerland, November 2008.

[13] A. van Lamsweerde. Requirements Engineering: From System Goals to UML Models to Software Specifications, Wiley, 2009.

[14] E. Yu. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering, Proc. 3rd IEEE Symposium on Requirements Engineering, pp. 226-235, 1997.

 ISBN: 978-1-61208-185-4