Non-Functional Requirements Revisited

Feng-Lin Li[1], Jennifer Horkoff[1], John Mylopoulos[1], Lin Liu[2], Alexander Borgida[3]

1: Dept. of Information Engineering and Computer Science, University of Trento, Trento, Italy
2: School of Software, Tsinghua University, Beijing, China
3: Department of Computer Science, Rutgers University, New Brunswick, USA

Abstract. Goal-Oriented Requirements Engineering (GORE) is founded on the premise that functional and non-functional requirements (NFRs) are stakeholder goals to be fulfilled by the system-to-be. Moreover, functional requirements are “hard” goals with clear-cut criteria for fulfillment, while traditionally NFRs are usually “soft” goals (aka softgoals) lacking a clear-cut criterion for success. We argue against this distinction and in favor of a different one: traditional NFRs (e.g., security, reliability, performance, usability etc.) are requirements for ***qualities*** that existentially depend on the subject they qualify. We give examples in support of our argument, and sketch an abstract syntax and semantics for goal models that follow our proposal.

Keywords: Goal Model, Softgoal, Quality, Quality Constraint, Ontology

1 Introduction

The rise of goal orientation as a research paradigm for Requirements Engineering (RE) is founded on the premise that functional and non-functional requirements can be modeled and analyzed as (stakeholder) goals. According to this view, functional requirements are modeled as hard goals with a clear-cut criterion for fulfillment, while non-functional requirements (hereafter NFRs) are modeled as soft goals (aka softgoals) with no such criterion [1], hence their name. This paradigm served as research baseline for *i** [2], but has also enjoyed much broader attention within the RE community during the past 20 years.

In this position paper we challenge this orthodoxy. In particular, we argue that traditional NFRs, such as performance, reliability, maintainability, etc., are not softgoals, but rather requirements for *qualities* that existentially depend on the subject they qualify [3]. Moreover, as quality requirements, these kinds of NFRs are very special optative statements: they constrain the space of allowable quality values for their subject. This key insight, missing from earlier treatments of NFRs, influences in important ways both the abstract syntax and the semantics of requirements models.

To capture this missing existential dependence, we distinguish *quality requirements* from NFRs and promote them to first-class status in our framework. At the same time, to allow the “defuzzification” of vague quality requirements, we follow Techne [4], which made the satisfaction of softgoals measurable by operationalizing them as “quality constraints” – though at the time the term “quality” did not have the ontologically technical meaning to be introduced in this paper. Based on these, we revisit the requirements model and give examples in support of our proposal.

2 Quality Requirements

DOLCE [3] is an ontology that aims at capturing the ontological categories underlying natural language and human common sense. It provides a rich theory of qualities, which we adopt here. According to DOLCE, a ***quality*** is a particular which applies to a particular subject, and *inheres in* that subject, unable to exist independently from it. This existential dependence makes the semantics of qualities different from those of softgoals: we can only assess the satisfaction of a quality if its subject exists. For example, we cannot assess the security of the goal "*Message be Sent*" unless the message has been sent (this is also noted in [5]). However, this important existential dependence of qualities has not been captured in previous GORE techniques, such as *i** [2], the NFR framework [6], Tropos, and Techne [4].

The original proposal in DOLCE associates a particular quality, say *c*#1, of quality type *Cost*, to an individual, say *trip*#1, in class *Trip*; and then associates with *c*#1 the particular cost value, say 1000€ , in the *Cost* quality space *EuroValues*. We rephrase this by using a single, higher-order function *hasQualityValue*, which takes as argument the quality type *QT*, and returns a function that maps particular subjects to their quality values. Thus *hasQualityValue*(*QT*) can be used as in *hasQualityValue*(*Cost*) (*trip*#1) to obtain the cost value 1000€. Meanwhile, we overload this function to apply to a class/type *Subj*, by applying *hasQualityValue*(*QT*) to all instances of *Subj*, obtaining the set of quality values for all these individuals. E.g., *hasQualityValue*(*Cost*) (*Trip*) returns the set of all trips' costs. Note that in order to accommodate different degrees of accuracy, DOLCE views a quality space as consisting of regions with sub-regions. So a trip might have "*Low*" cost, with "*Very Low*" being a sub-region.

We view a ***quality requirement*** (*QR*) as a constraint over the region of expected quality values (e.g. low, fast, or high) of relevant *QT* (e.g. cost, speed, or performance) for a subject (type). On the basis of the above defined function, this can be formalized as $hasQualityValue(QT)(Subj) \sqsubseteq RG$, where *QT* is a quality type, *Subj* is a class of subject individuals, *RG* is a possibly underspecified region of desired quality values. To make the quality region *RG* measurable, quality constraints (*QCs*) are introduced to precisely define its boundary [4][7]. For example, the quality requirement "*the Cost of Trip should be Low*", formally written as $hasQualityValue(Cost)(Trip) \sqsubseteq Low\ Cost$, can be made more precise by specifying a *QC*: $Low\ Cost = \{y \mid 0 \leq y \leq 500€\}$.

On this view, we can express the vagueness of quality requirements using the desired regions of quality values (of corresponding qualities), which are fuzzy (e.g. low, fast). QRs can be made measureable by operationalizing them as quality constraints. Keep in mind that NFRs are not only quality requirements; e.g., the NFRs "*Response Time* ≤ 5sec" and "*Take a Nice Trip*" can be modeled as hard goals and softgoals resp. The foundational concepts related to NFRs in our framework are defined as follows.

- A ***softgoal*** is a goal without a clear-cut definition of satisfaction [2], but an *independent* existence. Its satisfaction is determined by its refinements, which can include softgoals, hard goals and quality goals.
- A ***quality goal*** (*QG*) represents a quality requirement, which consists of a quality type, a desired quality region and a subject, with a different representation; e.g., a *QG* "*Low Cost*[*Trip*]" represents a *QR* "*the Cost of Trip should be Low*". A quality goal can be refined as quality goals or operationalized by a quality constraint. Its satisfaction is determined by its subject and refinements/operationalization.

- A ***quality constraint*** precisely defines a desired quality region in the quality space of a quality type and is achieved or denied by tasks. In our framework, quality constraints are metrics, but they are ontologically broader than metrics according to Techne [4], from which this concept is adopted.

3 A Framework for Goal Models with Quality Goals

To address the observed deficiency, we propose a revisited goal modeling framework on the basis of Techne [4], which extends *i** with quality constraints, preferences and inconsistency handling. We use extended EBNF (Extended Backus-Naur Form) to sketch an abstract syntax, wherein "+" means one or more, "∗" means zero or more, "·" means 'and', "|" means 'exclusive or', "→" means substitution, and "→" with a label at the top indicates particular relationship. We intend to describe a graphical notation and don't impose an order for elements on the right-hand side of the rules.

(1) $Goal \overset{is}{\rightarrow} Hardgoal \mid Softgoal \mid QualityGoal$
(2) $Hardgoal \xrightarrow{refine} Hardgoal^{+}$
(3) $Hardgoal \xrightarrow{operationalizeAs} Task^{+}$
(4) $Softgoal \xrightarrow{refine} Hardgoal^{*} \cdot Softgoal^{*} \cdot QualityGoal^{*}$
(5) $QualityGoal \xrightarrow{refine} QualityGoal^{+}$
(6) $QualityGoal \xrightarrow{operationalizeAs} QualityConstraint^{+}$
(7) $QualityGoal \rightarrow Quality'[Subj]$

Fig.1 The Abstract Syntax of the Revisited Goal Modeling Framework

The revisited requirements model is shown in Fig. 1 and visualized in Fig. 2. As the baseline, we use goals as a means to represent requirements (thus a quality requirement is now represented as a quality goal). *Quality Goal* is the key concept in our framework: it is expressed in the form of *Quality'*[*Subj*] (*Q'* [*Subj*] for short, e.g. "*Low Cost*[*Book Flight*]"), in which *Q'* is an ***intentional quality*** with both quality type and the desired region of quality values (e.g. *Low Cost*) and *Subj* is the type of subjects (e.g. *Book Flight*) that *Q'* inheres in; its operationalization, namely quality constraint, clearly specifies the region within the entire quality space.

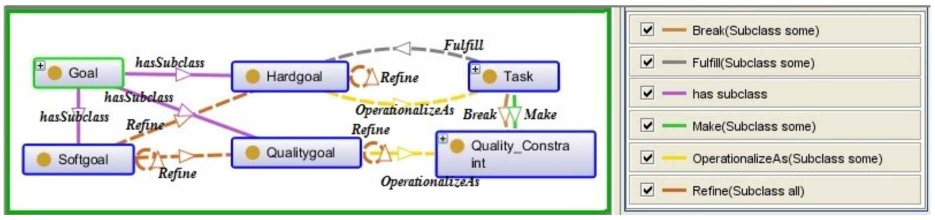

Fig.2 The Visual Representation of the Revisited Goal Modeling Framework

A travel example is used to illustrate the concepts in the framework. In this scenario, an employee *Lily* plans to take a leisure travel in Europe. As shown in Fig. 3, the top-level goal is a soft goal "*Take a Nice Trip*", which is refined to a hard goal "*Take a Trip*" and a quality goal "*Nice*[*Take a Trip*]". The hard goal is iteratively refined and finally operationalized as tasks; the quality goal is refined to other quality goals which are operationalized by quality constraints.

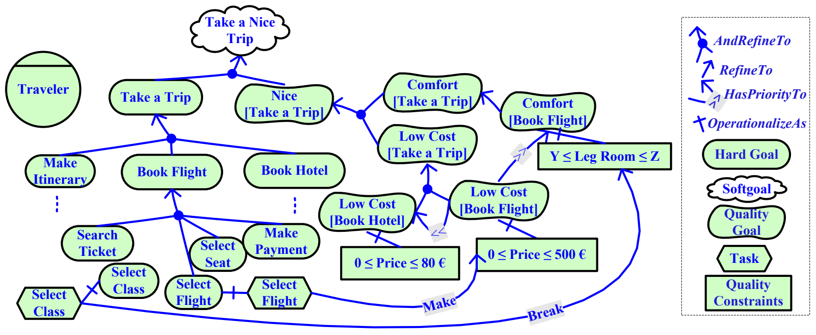

Fig.3 A Travelling Example

The refinement of a quality goal *Q'* [*Subj*] has two forms: via the intentional quality *Q'* and via the subject *Subj*. The refinement of *Q'* can be performed by following the existing quality type/dimension standards (e.g. ISO standards 9126 and more recently 25010), or using domain-specific knowledge. The refinement of *Subj* depends on either the hierarchy of hard goals or the ontology of subjects in Fig. 4. For example, refining by quality, *Nice* for subject *Take a Trip* usually includes *Low Cost*, *Comfort*, *Convenient* and *Timeliness*. In our example, only a subset of them is considered: "*Nice*[*Take a Trip*]" is refined as "*Low Cost*[*Take a Trip*]" and "*Comfort*[*Take a Trip*]". Refining by subject, taking a trip is composed of booking flight and hotel in the hard goal hierarchy, so the quality "*Low Cost*[*Take a Trip*]" is refined as "*Low Cost*[*Book Flight*]" and "*Low Cost*[*Book Hotel*]". In brief, the refinement of quality goals can be summarized as two rules:

$(Rule \cdot A{:}\ Refine\ by\ Quality)Q' \xrightarrow{x} \{Q'_1, Q'_2 \dots Q'_n\} \Longrightarrow$

$\quad Q'[Subj] \xrightarrow{x} \{Q'_1[Subj], Q'_2[Subj] \dots Q'_m[Subj]\}, (x = AND-refine|refine, 0 < m \le n)$

$(Rule \cdot B{:}\ Refine\ by\ Subject)Subj \xrightarrow{x} \{Subj_1, Subj_2 \dots Subj_n\} \Longrightarrow$

$\quad Q'[Subj] \xrightarrow{x} \{Q'[Subj_1], Q'[Subj_2] \dots Q'[Subj_m]\}, (x = AND-refine|refine, 0 < m \le n)$

In trying to understand the semantics of quality goals, we adopt a holistic perspective: a software system isn't just code, or code plus design and requirements; rather, it includes all the things that concern its existence, its makeup, development processes and history. Accordingly, quality goals can have any of these aspects as subjects. To clarify this, we give a simple ontology in Fig. 4, which defines the scope over which qualities apply. We intend for such subjects to cover both the problem and solution domain, allowing us to express both "as-is" and "to-be" quality requirements. Our ontology is not claimed to be complete and can be extended on demand.

(1) $Subject \xrightarrow{is} SoftwareSystem$

(2) $SoftwareSystem \xrightarrow{hasParts} Goal^{+} \cdot Process^{+} \cdot Thing^{+}$

(3) $Process \xrightarrow{hasParts} DevelopmentP \cdot BusinessP$

(4) $DevelopmentP \xrightarrow{hasParts} ElicitReqP \cdot DesignP \cdot ImplementationP \cdot TestP$

(5) $Thing \xrightarrow{hasParts} Artifact^{+} \cdot Object^{+}$

(6) $Artifact \xrightarrow{hasParts} Architecture \cdot Behavioural \cdot Feature \cdot Code$

Fig.4 An Ontology for the Subjects of Qualities

In our proposal, the partial contribution links *help* and *hurt* are excluded, only *make* and *break* are preserved to represent the achievement of quality constraints by tasks.

We omit help and hurt because they force on us a four-valued logic (satisfied *S*, denied *D*, partially satisfied *PS* and partially denied *PD*) [8]. As such, potential conflicts arise: if a quality goal receives both help and hurt or two competitive quality goals are both partially satisfied/denied, it is problematic to identify which alternative solution (i.e. a set of tasks) better satisfies given hard goals, along with concerned quality goals [5].

Meanwhile, make and break allow us to deal with the achievements of quality constraints using a simpler two-valued logic (*S*/*D*). By incorporating preferences and priorities, our framework is able to facilitate the selection of the best alternative requirements. In our goal models, the "*HasPriotityTo*" links will be drawn from preferred quality goals to less preferred ones, and priorities will be assigned to concerned quality goals. In this way, we are able to obtain an ordered sequence of quality goals based on stakeholder preferences. Future work will focus on describing use of our framework for the requirements selection problem.

4 The Semantics of Quality Goals

Reasoning about the satisfaction of goals can be performed on goal models by using inference rules and label propagation algorithms [8][9]. The core of existing goal semantics can be summarized as [8]: if a goal *G* is AND-refined to a set of goals G_1, G_2 …G_n $(n \geq 1)$, then *G* can be satisfied only if all the sub-goals are satisfied; if *G* is OR-refined, *G* will be satisfied if any of the sub-goals is satisfied; if a goal G_1 makes (resp. breaks) a goal G_2, then G_2 will be satisfied (resp. denied) if G_1 is satisfied.

In our framework, the semantics of quality goals are two-fold: (i) they existentially depend on subjects (ii) they constrain regions of quality values, which are clearly specified by quality constraints. The existential dependence indicates: if the subject *Subj* of a quality goal *QG* is not satisfied, then the satisfaction of *QG* is unknown. We use symbol *N* for this situation, which means "*no evidence*" (axiom 1). If the subject *Subj* is satisfied, then the satisfaction of *QG* is determined by its refinements (axiom 2 ~ 4). Take axiom (2) as an example, given that *QG* = *Q'* [*Subj*], and *QG* is AND-refined to G_1…G_i …G_n $(1 \leq i \leq n)$, then *QG* will be satisfied if its subject *Subj* is satisfied and all of the sub-goals are achieved. The other axioms are structured similarly.

(1) $QG = Q'[Subj]: D(Subj) \Rightarrow N(QG)$

(2) $QG = Q'[Subj], QG \xrightarrow{AND-refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge \left(\wedge_i S(QG_i)\right) \Rightarrow S(QG)(1 \leq i \leq n)$

(3) $QG = Q'[Subj], QG \xrightarrow{AND-refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge \left(\vee_i D(QG_i)\right) \Rightarrow D(QG)(1 \leq i \leq n)$

(4) $QG = Q'[Subj], QG \xrightarrow{refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge \left(\vee_i S(QG_i)\right) \Rightarrow S(QG)(1 \leq i \leq n)$

(5) $QG = Q'[Subj], QG \xrightarrow{refine} \{QG_1 \ldots QG_i \ldots QG_n\}: S(Subj) \wedge \left(\vee_i D(QG_i)\right) \Rightarrow D(QG)(1 \leq i \leq n)$

In accordance with the ontology in Fig. 4, the subjects can be processes or things instead of goals. In such cases, we can speak of process execution or thing existence instead of goal satisfaction. Moreover, in our framework, we use AND-refinement and refinement, and avoid the use of explicit OR.

(6) $QG = Q'[Subj], QG \xrightarrow{OperationalizeAs} QC:$
$\forall x: Subj, S(x) \wedge hasQualityValue(QT, x) \subseteq QC \Rightarrow S(QG)$

(7) $QG = Q'[Subj],\ QG \xrightarrow{OperationalizeAs} QC, T \xrightarrow{make} QC: S(T) \Rightarrow S(QC);\ S(Subj) \wedge S(QC) \Rightarrow S(QG)$

$(8)\ QG = Q'[Subj],\ QG \xrightarrow{OperationalizeAs} QC, T \xrightarrow{break} QC: S(T) \Rightarrow D(QC); S(Subj) \wedge D(QC) \Rightarrow D(QG)$

The axioms (6 ~ 8) deal with the satisfaction of quality goals at the bottom of the quality goal hierarchy. A quality goal *QG* will be satisfied, if the actual quality value of each individual in *Subj* on quality type *QT* belongs to the region *RG* being specified by a quality constraint *QC* (axiom 6). Axiom (7 ~ 8) show the semantics of make and break: if a task *T* makes (resp. breaks) a quality constraint *QC*, then the corresponding quality goal *QG* of *QC* will also be satisfied (resp. denied). It is worth mentioning that to deny *QG*, we also need its subject *Subj* to be achieved; otherwise the satisfaction of *QG* is unknown.

5 Conclusions and Future work

In this paper, we identified from NFRs quality requirements and model them as quality goals. Accordingly, we proposed a revisited goal modeling framework, sketching an abstract syntax and semantics. There are several interesting and challenging problems open for discussion: How to properly and systematically handle ambiguous qualities (e.g. inexpensive and low cost)? How will tasks influence the quality values of the concerned qualities? How to incorporate context in our goal models? These will be the key concerns in our next steps.

References

1. J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using nonfunctional requirements: A process-oriented approach," *Software Engineering, IEEE Transactions on*, vol. 18, no. 6, pp. 483–497, 1992.
2. E. S. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, 1997, pp. 226–235.
3. C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari, "WonderWeb Deliverable D18, Ontology Library (final)," 2009.
4. I. J. Jureta, A. Borgida, N. A. Ernst, and J. Mylopoulos, "Techne: Towards a new generation of requirements modeling languages with goals, preferences, and inconsistency handling," in *Requirements Engineering Conference (RE), 2010 18th IEEE International*, 2010, pp. 115–124.
5. J. Horkoff and E. Yu, "Comparison and evaluation of goal-oriented satisfaction analysis techniques," *Requirements Engineering*, pp. 1–24, 2012.
6. L. Chung, B. A. Nixon, and E. Yu, *Non-Functional Requirements in Software Engineering*, vol. 5. Kluwer Academic Pub, 2000.
7. I. J. Jureta, J. Mylopoulos, and S. Faulkner, "Revisiting the core ontology and problem in requirements engineering," in *International Requirements Engineering, 2008. RE'08. 16th IEEE*, 2008, pp. 71–80.
8. P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with goal models," *Conceptual Modeling—ER 2002*, pp. 167–181, 2003.
9. J. Horkoff and E. Yu, "Analyzing goal models: different approaches and how to choose among them," in Proceedings of the 2011 ACM Symposium on Applied Computing, 2011, pp. 675–682.