# Similarity-based Reasoning with Quality Goals

Feng-Lin Li[1,2,3], Lin Liu[2] and John Mylopoulos[1]

[1] University of Trento, Trento, Italy
[2] Tsinghua University, Beijing, China
[3] Alibaba Corp., Hangzhou, China

**Abstract.** During the years, *softgoals* have enjoyed much attention in the requirement engineering (RE) community for handling non-functional requirements (NFRs). As pointed out by RE researchers in recent years, softgoal may not be the right concept for non-functional requirements since it also captures early functional requirements (FRs) that are vaguely defined, e.g., "increase profit". Alternatively, *quality goals* have been proposed. However, existing reasoning techniques on softgoals also have some deficiencies and can not be directly adopted: qualitative approaches lack sufficient accuracy and expressiveness for decision making, while the numbers used in existing quantitative procedures are subjective. This paper proposes a two-staged approach to tackle the problem of reasoning with quality goals: in the first stage we measure the satisfaction of leaf-level quality goals based on prototype values and graded membership, and in the second stage we propagate the satisfaction degree using similarity measurement. Our approach aims to address the aforementioned issues in the current reasoning procedures of quality goals.

**Keywords:** Quality Goals, Satisfaction Analysis, Graded Membership, Similarity Measurement.

## 1 Introduction

Non-functional requirements (NFRs) have been commonly acknowledged as a critical class of requirements in RE as it significantly influences user satisfaction and the success of software/services. In the RE community, many techniques were proposed for dealing with NFRs, and goal-oriented approaches have been argued to be the major effort in dealing with NFRs in depth [3].

In goal-oriented requirements engineering (GORE), requirements are captured as *goals* that represent state-of-affairs in the world and need to be brought about by a system-to-be, and non-functional requirements are captured as *softgoals* that are vague for success. As one of the key concepts in the NFR Framework [2] and $i*$ [12], softgoal has been popular in dealing with NFRs since its proposal in the 90's. However, it turns out that non-functional requirements are not always "soft", e.g., "the processing time of a product search shall be less than 1 second" is clear for success. On the other hand, early functional requirements can also be vague, e.g., "help administers to resolve problems".

That is, softgoals constitute a useful abstraction for early requirements, both functional and non-functional, rather than just non-functional ones [10]. To address this issue, we have proposed to treat NFRs as qualities and capture them as *quality goals* (QGs) [10]. Also, we have proposed a syntax for representing, a set of operators and a systematic process for refining quality goals [9].

With goal-oriented techniques, the AND/OR structure of goal models naturally lends itself to an analysis of goal satisfaction [7], which has been advocated to be able to facilitate design decision and alternative selection. During the years, many goal-oriented frameworks and accompanying reasoning procedures have been proposed (e.g., the NFR Framework [2], $i^*$ [12], and Tropos [1]). However, these techniques are not flawless and can not be directly adopted to our quality goals. For example, the qualitative procedures in the NFR Framework and $i^*$ lack sufficient expressiveness as they use only five levels of satisfaction labels and linear aggregation in their propagation rules (e.g., minimum of AND), and the quantitative procedure in Tropos needs the numbers that indicate the satisfaction of leaf-level goals to be assigned subjectively and manually.

In this work, we focus on goal satisfaction analysis, more specifically, the forward satisfaction analysis of quality goals. We propose a two-staged reasoning procedure: first, we measure the satisfaction of leaf-level QGs based on prototype values and graded membership; then, we propagate the satisfaction of lower-level QGs to higher-level QGs based on similarity measurement.

The rest of this paper is organized as follows. Section 2 briefly reviews related work on goal satisfaction analysis, Section 3 describes our treatment of NFRs as requirements referring to qualities, Section 4 presents the two-staged reasoning procedure, Section 5 concludes the paper and offers suggestions for future work.

## 2 Related Work

In general, goal satisfaction analysis can be categorized into two classes: *qualitative* and *quantitative* [7]. The former approach uses qualitative labels to represent the satisfaction of goals, e.g., satisfied, partially satisfied, unknown, partially denied, and fully denied, and the contribution strength of a goal to another, e.g., help, hurt, make and break. The latter uses numeric values within an interval, e.g., [0, 1], to represent the satisfaction of and the contribution among goals.

Both qualitative and quantitative approaches use some propagation rules to propagate initial satisfaction labels of lower-level goals to higher-level ones. For example, the NFR Framework [2] propagates the minimum value for AND contributions and the maximum value for OR. Quantitative procedures differ from qualitative ones by assigning numeric values to the satisfaction degree of goals and the contribution strength of links, then calculating a (normalized) linear combination of their arithmetic products (multiplying the satisfaction value of contributing goals by its contribution strength) [7]. A comprehensive comparison over goal satisfaction analysis procedures is conducted in [7].

In our observation, the aggregation from the satisfaction of lower-level goals to that of higher-level ones is often non-linear. For example, supposing that we have a softgoal “enjoyable trip” being AND-decomposed into “low cost”, “enjoyable activities” and “delicious food”, how about if we finally have “enjoyable activities” and “delicious food” but a “high cost”? It is probably not the minimum satisfaction value in this case, at least for some classes of users.

A second issue here is how to determine the satisfaction degree of leaf-level goals, which are often manually assigned in existing qualitative and quantitative approaches. Letier et al. [8] have also argued against the subjective numbers used in quantitative approaches, and proposed an approach to reasoning over the probabilistic satisfaction of non-functional properties. We agree that probabilistic satisfaction is an important characteristic of NFRs, but we also should not leave out the gradability of NFRs. For example, if we require the cost of a trip to take less than 600 Euros, how about it turns out to be 620 Euros?

## 3 The Theory – NFRs as Qualities

Our treatment of NFRs [9] is based on the Unified Foundational Ontology [5]: a quality (e.g., “cost#1”, the cost of a trip) is taken as a mapping function that maps the subject it inheres in (e.g., “trip#1”, the particular trip) to a value (e.g., “550€”), and an NFR is a requirement that requires the value of a quality to be in a desired region (e.g., “the cost of a trip shall be low”).

Vague NFRs are captured as ***quality goals*** (QGs) and denoted in the form of “$Q\ (SubjT) ::\ QRG$”, which is an abbreviation for “$\forall q \in Q, x \in SubjT, inheres\_in(q,x) \rightarrow has\_value\_in(q, QRG)$”. Using this syntax, the trip example can be captured as a QG “$Cost\ (Trip)\ ::\ Low$”, which is vague and will be further operationalized as a ***quality constraint*** (QC) “$Cost\ (Trip) :: \leq 600\,(Euros)$”, meaning that “for each trip, its cost shall be low (less than 600 Euros)”. Note that QCs share the same syntax with QGs, but have measurable regions [10].

QGs can be refined via its quality, subject, desired region, or a combination of them [10]. Two kinds of operators, namely *reduce* and *operationalize*, are used to refine a QG: the former refines a QG (resp. QC) to a set of QGs (resp. QCs), and the latter operationalizes a QG to a set of QCs. In the case there is only one element in the resulting set, the refinement is akin to a classical OR; if there is more than one resulting element, then the refinement emulates a classical AND.

We show some example snippets of QG refinement in Fig. 1. There are two points to be noted. First, we allow alternative operationalizations of a QG. This is particularly useful when dealing with context, which is captured as part of the subject of a QG, e.g., a low cost trip means differently to professors versus students. Second, we do not allow refinements from QCs, which are specification elements, to QGs, which represent requirements, because our main purpose is to turn informal stakeholder requirements into an eligible specification; we do support refinements of QCs to QCs after operationalization (e.g., weakening [9]).

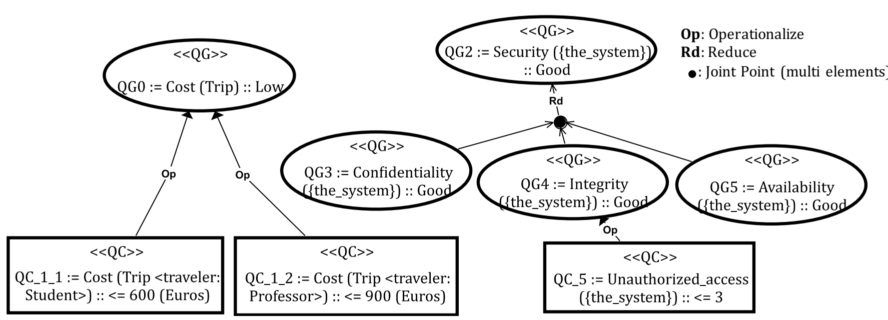


Fig. 1: Quality goal refinement snippets

## 4 Reasoning with Quality Goals

This section presents a two-staged reasoning procedure for the satisfaction analysis of quality goals. The first stage is previous work on measuring the satisfaction of leaf-level QGs [6], and the second stage is novel contribution on the propagation of the satisfaction degree of leaf-level QGs to higher-level QGs.

### 4.1 Measuring the satisfaction of leaf-level quality goals

In general, complex quality goals will be reduced to sub-QGs and finally operationalized as measurable quality constraints (QCs) during software design. And, a QC is satisfied if the perceived quality values fall into the expected region, or unsatisfied otherwise. That is, such a QC is akin to a binary function that returns 1 or 0. This may be too strong a condition since in many cases "good enough" is sufficient. For example, if one has operationalized "Low" as "$\leq 600\,(Euros)$", how about a cost of 620 Euros? As we can see, the satisfaction degree, not the make-or-break status, is what actually matters. So, when operationalizing a QG, we need to derive a smooth function, not a binary one.

To address this issue, we have proposed to measure the satisfaction degree of leaf-level QGs based on *gradable membership* [4,6]. This technique assumes that a conceptual space (e.g., a single-dimensional space for cost) consists of a set of regions (e.g., low, medium and high), each of which is associated with a concept C and represented by some *prototypical values* (e.g., 500€ and 700€ for low, 800€ and 1000€ for medium, 1200€ and 1500€ for high). It starts by considering the set of all possible selections of exactly one prototypical value from each region (e.g., the three points 500€, 800€ and 1200€ form a selection; and we will have $2^3 = 8$ such selections in this case). Each selection can be used to generate a Voronoi diagram [11] (a partition of a space into regions/cells based on distance to pre-set points in the space), in which a Voronoi cell is associated with a concept represented by a selected prototypical value. In this single-dimensional example, we use medians to generate partitions, e.g., $(500 + 800)/2 = 650$, $(800+1200)/2 = 1000$, and we have (0, 650] for low, (650, 1000] for medium, and (1000, $1000^+$) for high. Finally, it defines the membership of a value $p$ belongs to a concept C as the number of Voronoi diagrams that vote $p$ to the Voronoi cell associated with C. For instance, in our example, if a cost value is 740€, then it belongs to low with a degree of 0.75 (6 out of the 8 diagrams vote it to the cell associated with low) and belongs to medium with a degree of 0.25. Interested readers can refer to Li [9] for the calculation details and the derivation of math expressions for satisfaction degrees based on prototype values/regions.

The interesting point here is that we can use prototype values to represent a concept, and then use Voronoi diagrams to reason about the graded membership without the need of inventing made-up numbers as in existing quantitative goal satisfaction analysis procedures.

### 4.2 Propagating the satisfaction value of quality goals upstream

In practice, a QG may refer to a composite quality that has a set of sub-qualities as its offsprings, e.g., "Security" has sub-qualities "Confidentiality", "Integrity" and "Availability" (see the ISO/IEC 25010:2011 standard for a reference quality hierarchy), or a complex subject that has many different parts as it components, e.g, a system. In such cases, we need to first refine the complex QG to simple ones via its quality/subject, then use the technique discussed in Section 4.1 to measure the satisfaction of leaf-level QGs, and finally propagate up the satisfaction of lower-level QGs along the refinement hierarchy.

Recall that a QG can be reduced to a set of sub-QGs using our *reduce* operator. For such a refinement, we represent the QG as a vector of its sub-QGs. For example, we denote the refinement of a quality goal $QG$ to $QG_1$ and $QG_2$ as $QG \doteq [QG_1, QG_2]$. We set the expected satisfaction degree of each sub-QG to be 1.0, and express the expected satisfaction of QG as a $1 \times 2$ vector: $SatE_{QG} = [SatE_{QG_1}, SatE_{QG_2}] = [1.0, 1.0]$. Supposing that $QG_1$ and $QG_2$ are leaf-level goals associated with single-dimensional spaces, we can derive their satisfaction degrees by using the technique discussed in Section 4.1. On obtaining the satisfaction degree of $QG_1$ and $QG_2$, say $sat_1$ and $sat_2$, we have the vector $SatR_{QG} = [SatR_{QG_1}, SatR_{QG_2}] = [sat_1, sat_2]$. Finally, we use Eq. 1 to compute the the satisfaction degree of QG, where $n$ is the dimension of $SatE_{QG}$ (here, $n = 2$), $d$ is the Euclidean distance between $SatE_{QG}$ and $SatR_{QG}$, $d_{max}$ is their maximum distance when $SatR_{QG} = [0.0, 0.0]$ and is in fact a constant $\sqrt{n}$.

$$Sat_{QG} = 1 - \frac{d}{d_{max}} = 1 - \sqrt{\frac{\sum_{i=1}^{n} (SatE_{QG_i} - SatR_{QG_i})^2}{n}} \tag{1}$$

The rationale of this approach is to reduce a composite quality in a high-dimensional space to low-dimensional spaces (i.e., sub-qualities) and measure the similarity between reality and expectation. The geometric meaning of this approach is shown in Fig. 2. As shown in the figure, "Security" is represented as a point $E$ in the three-dimensional space and can be refined to "Confidentiality", "Integrity" and "Availability". We denote the expected satisfaction of "Security" as [1.0, 1.0, 1.0], and suppose that the real satisfaction $R$ to be [0.8, 0.4, 0.75]. The distance $d$ between the expectation and the reality is represented by $ER$, the maximal distance $d_{max}$ is $OE$, and the similarity between the reality $R$ and the expectation $E$ is $(1 - ER/OE)$. This approach can also be applied to subjects, which can be ref

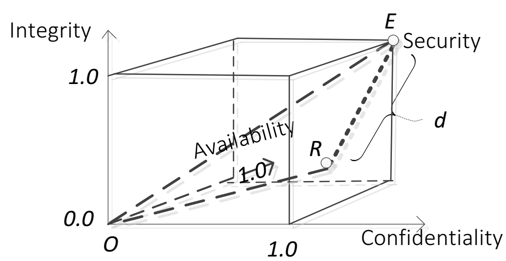


Fig. 2: The geometric meaning of similarity in our approach

The idea of similarity is borrowed from Machine Learning, where entities (e.g., document, images and words) are often represented as a vector of features. In a similar spirit, we denote the expected and actual satisfaction of a QG as vectors, with each of its sub-goals being a feature, and calculate their similarities using the metric defined in Eq. 1. Moreover, unlike Machine Learning techniques, our approach does not need to learn the satisfaction function of a leaf-level QG from big data; instead, we able to achieve the purpose by using prototype values, which are elicited from some stakeholders and are affordable in practice.

## 5 Discussion and Future Work

In this paper, we have proposed a two-staged procedure for the satisfaction analysis of quality goals. We first measure the satisfaction of leaf-level QGs based on graded membership and then propagate the satisfaction value upstream based on similarity measurement. Our approach is promising in addressing some of the deficiencies in existing reasoning procedures, e.g., the lack of sufficient expressiveness for qualitative approaches, and the meaningless of subjective numbers used in quantitative approaches.

We are going to implement this procedure in *Desiree* [4], a prototype tool that has been developed to support our requirement engineering framework [9]. We suggest that several interesting issues remain open for further investigation. For example, how to evaluate the effectiveness of this two-staged procedure using realistic case studies, how to set the weights of QGs in a QG hierarchy, and how to deal with conflicts in reasoning processes.

**Acknowledgments**. This work is supported by the European Research Council advanced grant (no. 267856), the Key Project of National Natural Science Foundation of China (no. 61432020) and the Intelligence Innovation Center, Alibaba.

## References

1. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. JAAMAS 8(3), 203–236 (2004)
2. Chung, L., Nixon, B.A., Yu, E.: Non-Functional Requirements in Software Engineering, vol. 5. Kluwer Academic Pub (2000)
3. Chung, L., do Prado Leite, J.C.S.: On non-functional requirements in software engineering. In: Conceptual modeling: Foundations and applications, pp. 363–379. Springer (2009)
4. Decock, L., Douven, I.: What is graded membership? Noûs 48(4), 653–682 (2014)
5. Guizzardi, G.: Ontological foundations for structural conceptual models. CTIT (2005)
6. Guizzardi, R.S., Li, F.L., Borgida, A., Guizzardi, G., Horkoff, J., Mylopoulos, J.: An ontological interpretation of non-functional requirements. In: FOIS. pp. 344–357 (2014)
7. Horkoff, J., Yu, E.: Comparison and evaluation of goal-oriented satisfaction analysis techniques. REJ 18(3), 199–222 (2013)
8. Letier, E., Van Lamsweerde, A.: Reasoning about partial goal satisfaction for requirements and design engineering. SIGSOFT SEN 29(6), 53–62 (2004)
9. Li, F.L.: Desiree - a Refinement Calculus for Requirements Engineering. Ph.D. thesis (2016)
10. Li, F.L., Horkoff, J., Mylopoulos, J., Guizzardi, R.S., Guizzardi, G., Borgida, A., Liu, L.: Non-functional requirements as qualities, with a spice of ontology. In: RE. pp. 293–302 (2014)
11. Wikipedia: Voronoi diagram, `https://en.wikipedia.org/wiki/Voronoi_diagram`
12. Yu, E.: Modelling strategic relationships for process reengineering. Ph.D. thesis (1995)

[4] The tool is available at https://goo.gl/oeJ9Fi.