Towards a Framework for Improving Goal-Oriented Requirement Models Quality

Carlos Cares
Depto. de Ingeniería de Sistemas
Universidad de La Frontera
Temuco, Chile
ccares@ufro.cl

Xavier Franch
Dep. Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Barcelona, Spain
franch@lsi.upc.edu

Abstract

Goal-orientation is a widespread and useful approach to Requirements Engineering. However, quality assessment frameworks focused on goal-oriented processes are either limited or remain on the theoretical side. Requirements quality initiatives range from simple metrics applicable to requirements documents, to general-purpose quality frameworks that include syntactic, semantic and pragmatic concerns. In some recent works, we have proposed a metrics framework for goal-oriented models, but the approach did not cover the cycle of quality assessment. In this paper we present a semiotic-based quality assessment proposal built upon the i framework and the SEQUAL proposal. We propose a simplification of SEQUAL which can be applied to i* models by defining semantic, pragmatic and social metrics. As a result, we obtain suites of metrics that can be applied to i* goal-oriented requirements models. This theoretical work is put into practice by using iStarML, a XML representation of i* models, over which XQuery sentences compute the proposed metrics.*

1. Introduction

Goal-Oriented Requirements Engineering (GORE) [1-3] is playing a relevant role in the Requirements Engineering (RE) discipline since the mid nineties. The simple question about the "reason why" behind some requirement has opened the door to Software Engineering (SE) not only to better understand specific business domains but also to propose different alternatives of achieving organizational goals. Besides, goal-oriented modeling provides an effective approach to understand, specify and design distributed information systems that need to operate in open, heterogeneous and evolving environments.

As remarked in [4] and easily checkable from the current state of the art, KAOS and the *i** framework are the two main goal-oriented modeling techniques. In this paper we focus on the *i** framework [5]. It aims at facilitating domain modeling, domain reengineering, and computer system requirements modeling. The *i** framework includes some procedural guides to build strategic and intentional models using the *i** modeling language. The *i** framework has spread and successfully been implemented in different contexts, e.g. organizational modeling [6, 7], requirements elicitation [8, 9], software design [10, 11] and security [12, 13]. Its explicit representation of goals and actors has allowed to use it in both GORE [1-3] and Agent-Oriented Requirements Engineering (AORE) [11, 14].

In any SE-related discipline, the issue of Quality Assurance and Measuring comes into discussion. RE is not an exception. Although software measurement has been proposed more than 30 years ago [15] it is far to be considered an accomplished task in the context of RE (and probably in most other SE contexts). On the practical side, quality in RE has been tackled by defining metrics related to requirements documents, such as number of text lines, number of imperatives ("must", "shall", "will"), continuances following imperatives (e.g., "the following"), number of ambiguities or weak phrases ("fast", "enough"), number of optional phrases ("can", "may", "it is desirable that") among others [16]. All these metrics have the relevant quality of being feasible and easy to apply at the requirements stage. However, at the same time, these metrics show just a few, and not too relevant, points where improving requirements quality.

Another topic on requirements quality is the measurement of requirement properties, e.g., it is recognized that traceability is a relevant positive requirements feature [17]. Also volatility is a feature that needs to be controlled [18]. Another set of desirable requirements properties have been presented in [19], where are mentioned completeness, consistency, correctness among many others. In the case of the *i** framework, this type of properties could be used as indicators that may help on finding methods and techniques to overcome some of the drawbacks identified when using i* models [20]. Being these quality features more relevant in terms of the possibilities of improving requirements quality, they present a greater difficulty to be measured.

In the *i** framework, we have recently proposed the *i** Metrics Definition Framework (*i*MDF) [21, 22]. This framework supports the definition of metrics through metric patterns [23] for measuring properties like the ones above. However, *i*MDF does not support the reasoning process that allows deducing the form that the metrics must take in order to measure the required properties. Furthermore, the metrics are defined as OCL formulae, which makes necessary to express *i** models as instances of the *i** metamodel and in our experience this may hamper its adoption, especially by practitioners. As a consequence, we may say that there is a gap between the theoretical framework represented by *i*MDF and the practical issue of how to make this framework close to the needs of the requirements engineer and easy to implement in terms of development costs.

This paper is intended to be a first step to fill this gap. On the one hand, we propose to adopt the SEQUAL proposal [24-27] as framework for reasoning about quality. SEQUAL has a strong theoretical foundation by considering semiotic concepts, i.e. not only documents or models but also actors as interpreters, informality and diversity of languages, and the social phenomena of agreeing. SEQUAL's structure focused recently in the particularities of the requirements stage [27] whilst keeping the original focus on three types of qualities, syntax quality, semantic quality and pragmatic quality.

On the other hand, we propose to use the iStarML interchange format [28] as the baseline for implementing the metrics. In fact, iStarML is an XML-based representation of *i** models conceived mainly to overcome the interoperability problem in the community of *i** tools. However, being XML-based, it is possible to take advantage of the big deal of services around this standard and in particular, to use some tools to analyze models with some purpose like for instance, computing the value of metrics defined as XML queries.

The rest of the paper is organized as follows. Section II provides the necessary background to understand the proposal, focusing on the *i** framework, the iStarML interchange format and the SEQUAL proposal (including a preliminary discussion on notions of semiotics). In Section III we develop a simplified SEQUAL-based framework whose metrics require interaction with stakeholders, and we show some samples of metrics and patterns that may be used to define them. Section IV shows how the framework can be put into practice by applying several steps whose practical result is the creation of a query expressed in Xpath over the iStarML representation of the model. Finally, Section V summarizes the proposal and identifies open issues on this research line.

2. Antecedents

2.1. GORE and the *i** Framework

GORE methodologies constitute a relevant trend in SE. GORE was proposed about fifteen years ago having a great impact on the RE community since that time, due to its crucial role on the understanding of stakeholders' business goals in order to model the domain and to identify the requirements of new software systems by stepwise decomposition of high-level goals [29, 30]. High-level goals capture the overall organizational objectives and key constraints; therefore they represent stable needs that are less sensitive to changes.

As mentioned in the introduction, the *i** framework is one of the most widespread GORE approach. It includes a modeling language and some reasoning techniques. In Fig. 1 we replicate Kavakli's view of the *i** framework as a GORE approach [1].

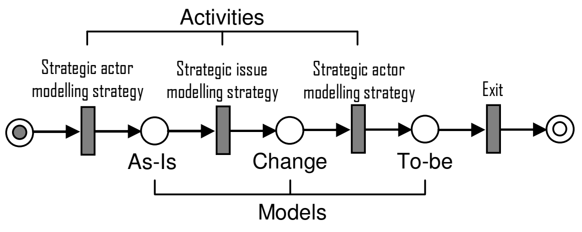

Activities
Strategic actor modelling strategy
Strategic issue modelling strategy
Strategic actor modelling strategy
Exit
As-Is
Change
To-be
Models

Fig. 1. Kavakli's view of the *i** scope into GORE processes

The *i** framework [5] proposes the use of two types of models, each one corresponding to a different abstraction level: a Strategic Dependency (SD) model represents the intentional level and the Strategic Rationale (SR) model represents the rational level.

A SD model consists of a set of nodes that represent actors and a set of dependencies that represent the relationships among them, expressing that an actor (depender) depends on some other (dependee) in order to obtain some objective (dependum). Depending on the dependum kind, the depender depends on the dependee to bring about a certain state in the world (goal dependency), to attain a goal in a particular way (task dependency), for the availability of a physical or informational entity (resource dependency) or to meet some non-functional requirement (softgoal dependency). It is also possible to define the importance of the dependency for each of the involved actors using three categories: open, committed and critical.

A SR model allows visualizing the intentional elements into the boundary of an actor in order to refine the SD model with reasoning capabilities. The dependencies of the SD model are linked to intentional elements inside the actor boundary. The elements inside the SR model are decomposed accordingly to two types of links: means-end links and task-decompositions.

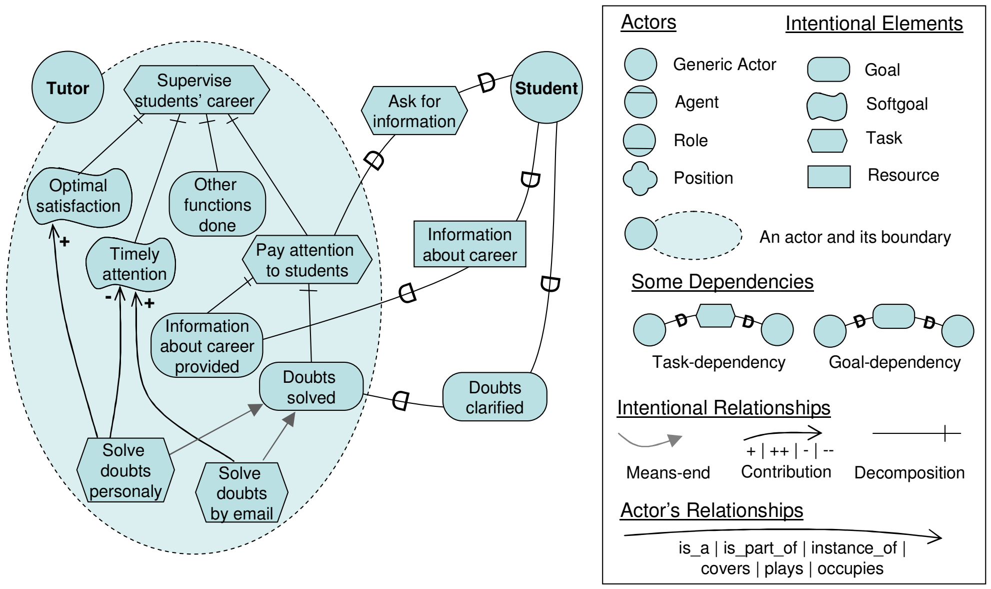


Fig. 2. The *i** framework: an example on tutoring.

Means-end links establish that one or more intentional elements are the means that contribute to the achievement of an end. The "end" can be a goal, task, resource, or softgoal, whereas the "means" is usually a task. There is a relation OR when there are many means, which indicate the different ways to obtain the end. The most used relationships are: Goal-Task, Resource-Task, Task-Task, Softgoal-Task, Softgoal-Softgoal and Goal-Goal (following usual conventions, the left-hand side of each pair represents the end and the right-hand side, the means). In Means-end links with a softgoal as end it is possible to specify if the contribution of the means towards the end is negative or positive.

Task-decomposition links state the decomposition of a task into different intentional elements. There is a relation AND when a task is decomposed into more than one intentional element. It is also possible to define constraints to refine this relationship. The importance of the intentional element in the accomplishment of the task can also be marked in the same way that in dependencies of a SD model.

Actors can be specialized into agents, roles and positions. A position covers roles. Agents represent particular instances of people, machines or software within the organization and they occupy positions (and as a consequence, they play the roles covered by these positions). Actors and their specializations can be decomposed into other actors using the *is-part-of* relationship.

These concepts and their graphical representation are illustrated in Fig. 2 using an example about academic tutoring of students. On the left-hand side, we show the SR model of a tutor and the hierarchical relationships among their internal intentional elements. On the right-hand side, we show the SD dependencies between a student and a tutor.

Different methodologies and language variations have been created based on *i** concepts and modelling techniques. A relevant proposal is *Tropos* [10, 11], an agent-oriented software development methodology that includes some language constructs or variations that are specific of this approach. Also relevant is GRL [31], an *i** variation which has been added as part of the industrial Telecommunications Standard Z.150 [32] for system specification. Besides these widespread proposals, there are others that have introduced different constructs in the language with their own research aims. Most of them have included software tools to support the underlying methodologies and techniques.

2.2. iStarML

In general, existing *i**-based tools and development frameworks are not capable to interoperate, i.e. to interchange models and diagrams, which prevents taking advantage of existing functionalities. One of the main reasons related to the lack of interoperability of different *i** frameworks is that the different *i**-based proposals use, define, or redefine the syntax or even the semantics of the seminal *i** constructs.

To confront the variability of models and in order to provide a common framework to enable interoperability among *i** researchers and users at the syntactic level, we have proposed iStarML [28], an XML format for representing *i** models. Its relevance is that the different *i** variants can eventually be translated into iStarML. Therefore iStarML allows a textual representation of domain models, requirements, actor relationships and a wide set of the different uses that *i** has covered as modeling language, particularly GORE and AORE aspects.

Previous work on the analysis of *i** metamodels [33, 34] has oriented us towards a core set of stable *i** abstract concepts which constitutes the basis of the existing *i** variations. We may distinguish up to six different parts that yield to six types of core concepts (see Areas in Fig. 3): (1) actor, for representing organizational units, humans or software agents; (2) intentional element, for representing the set of elements which give rationality to the actor's actions, e.g. goals and tasks; (3) dependency, for representing actors' dependencies in order to accomplish their own goals; (4) boundary, for representing the scope of actors; (5) intentional element link, for representing the relationships among intentional elements such as means-end or decomposition relationships; and (6) actor association link, for representing the relationships among actors such as is_part_of and is_a, among others. We have considered each area as a category of core concepts that drives the structure of iStarML. The definition of iStarML has allowed: *(i)* having a file format for diagrams interchanging among existing and new tools (see Fig. 4 for a sample); *(ii)* motivating the development and compliance of drawing tools to the defined format; and *(iii)* having a common way of representing the differences and similarities between existing *i** variations.

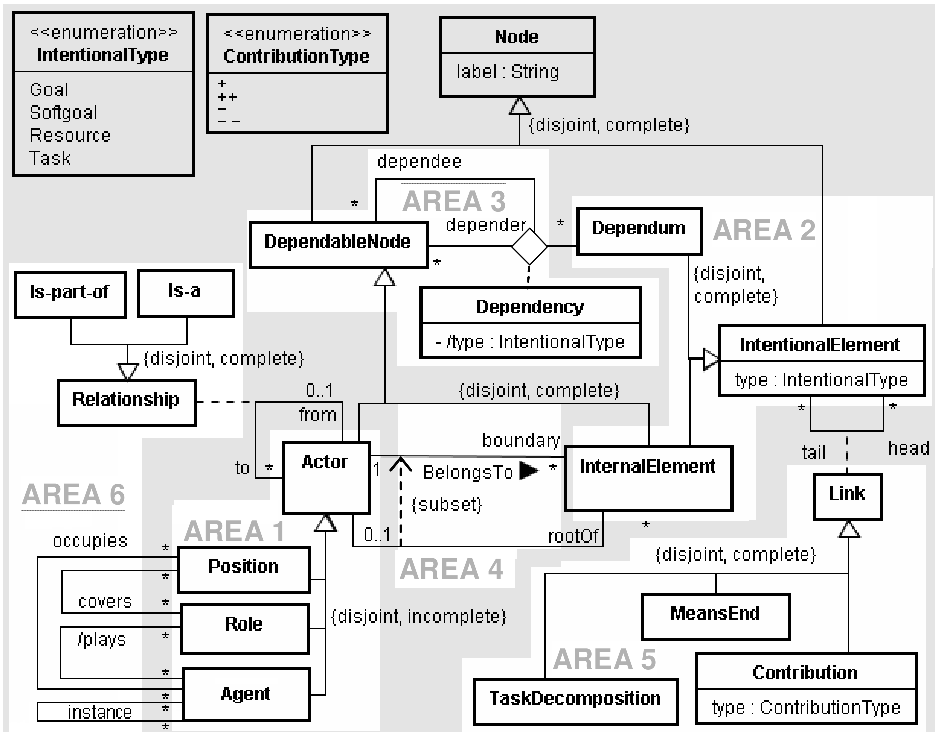
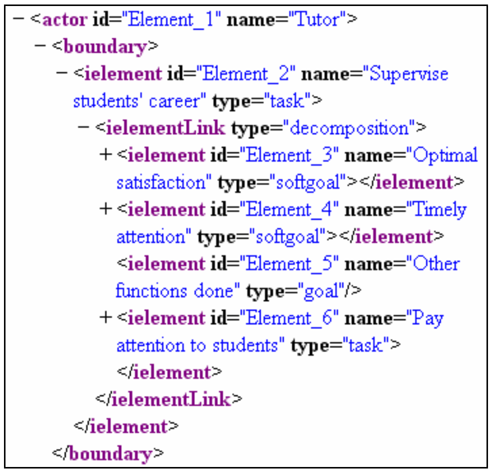


Fig. 3. Core concepts of *i**

It is worth to mention that the metamodel represents a basic conceptualization where *i** variants can be obtained by UML refactoring [33], being only a conceptual reference. This purely conceptual perspective would be Our proposal is to cover different conceptual frameworks, mainly Software Engineering Frameworks (e.g., Tropos), also in a more practical way. From this point of view, an OCL-based proposal would be only applicable to a specific *i** metamodel. However, if we formulate the calculable part on the iStarML interoperability proposal, we cover the family of *i** variants (different metamodels), moreover, if we add the current availability of software tools including iStarML, we facilitate the fact of adopting this quality proposal.



```
- <actor id="Element_1" name="Tutor">
  - <boundary>
    - <ielement id="Element_2" name="Supervise
      students' career" type="task">
      - <ielementLink type="decomposition">
        + <ielement id="Element_3" name="Optimal
          satisfaction" type="softgoal"></ielement>
        + <ielement id="Element_4" name="Timely
          attention" type="softgoal"></ielement>
          <ielement id="Element_5" name="Other
          functions done" type="goal"/>
        + <ielement id="Element_6" name="Pay
          attention to students" type="task">
          </ielement>
        </ielementLink>
      </ielement>
    </boundary>
```

Fig. 4. An excerpt of the iStarML of the tutoring example

2.3. SEQUAL

According to the Webster dictionary, semiotics is defined as “a general philosophical theory of signs and symbols that deals especially with their function in both artificially constructed and natural languages and comprises syntax, semantics, and pragmatics”. Pragmatics deals with the uses and effects of signs; semantics deals with the signification (meaning) of signs, that is the objects of the reality represented by signs; and syntax deals with combinations of signs regardless of their specific meaning or their relation to uses and effects [35]. A classical semiotic model is the Peirce’s model, which requires a repertory of signs (symbols) as communication elements, an ontology of objects and interpretants. The objects are elements of the “reality” for some community, also called cultural units, the classical example in semiotic is the conception of unicorn .The interpretants guarantee the validity of signs, normally there are alternative representations of objects such as synonyms, pictures, explanations or descriptions [36]. Although it may seem a little bit recursive, i.e. to explain a sign in terms of other signs at the same communication level, it is necessary to consider that some language sentences can reach stable meanings in the natural dynamic of meaning systems [37] and hence became (at least to a wide set of persons) a useful interpretant.

Therefore, a relevant semiotic idea is to include the process of meaning-making as part of the human communication phenomena [38]. In this conceptual framework, a sign represents an intended meaning (which implies that there is a referent) but it does not mean that the interpretation captures this meaning, even having access to some referent, i.e. it is accepted that the effect of the representation on the interpreter could not be the desired one. Therefore the sociolinguistic context, the knowledge and the experience of the interpreter become part of semiotic models because they are associated to the interpretation process. This perspective is used by U. Eco to state that semiotics is the Science of the culture and social conventions [36].

The SEQUAL (SEmiotic QUALity) framework is a semiotics-based reference model for assessing the quality of models, initially formulated at 1994 (with another name) [24]. As such, it aligns with the basic principles of semiotics and for instance, three types of qualities were initially recognized: syntax quality, semantic quality and pragmatic quality. Also, the elements of a model are signs that could have different interpretations depending on the sociolinguistic context and previous knowledge of their interpreters. This seminal proposal of SEQUAL was followed by a specific RE quality proposal [25] that incorporates into the framework the notion of the social process of agreement which was previously modeled by Pohl [39], yielding to a fourth dimension of quality, social quality. In Fig. 5 we reproduce the SEQUAL framework as it was illustrated in 1995.

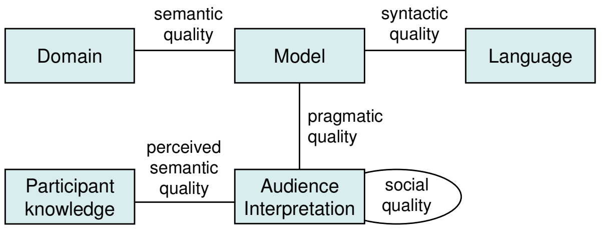


Fig. 5. The second version of SEQUAL (1995) [25]

Other remarkable characteristics of SEQUAL are explained next. First, in the 2002’s SEQUAL version [40], interaction principles were included. This extension meant adding new concepts into the model, such as modelers, the goal of the modeling process, and the appropriateness of the different actors over the representations, which means that actors feel having the control over the meaning system. Second, the evolution of the model and the articulation of the actors were also added into this version of SEQUAL. Third, in the

2004's SEQUAL version [26], the relevant semiotic concept of activation, intended to consider the impact of the models in interpreters, explicitly appeared in this model.

A revised SEQUAL model was proposed at 2006 [27]. This version included theoretical concepts such as the needed knowledge for optimal modeling and also the concept of a theoretical optimal domain, including the acquisition of knowledge of the participants in the process, the actions that change the domain, and the change of the domain model coming from a better understanding (interpretation) of the domain and also the evolution of the domain because the promoted changes. This model considered additional concepts as Ideal Semantic Quality in two versions (prescriptive and descriptive), among other extensions.

As a consequence of this evolution throughout the years, it may be said that the SEQUAL framework has become a rich model, complete enough to formulate a first SEQUAL-based approach to assess the quality of goal-oriented models.

3. *i** in a simplified SEQUAL framework

In this section we aim at formulating a SEQUAL framework for *i** with the purpose of assessing the quality of goal-oriented *i** models at the RE stage. With this purpose, we first formulate a simplified version of the SEQUAL framework oriented to get a first approach for improving goal-oriented models quality. As a second step we formulate some metrics related to the identified qualities considered in this simplified SEQUAL version. Concerning this second point, it is not an objective of this paper to provide a complete set of metrics but just to illustrate the way the engineer may proceed to produce them.

3.1. A SEQUAL simplified framework

Due to the great complexity of the SEQUAL framework as presented in the previous section, we have considered a reduced version of the model (see Fig. 6) in order to tackle our first practical approach. First, we have considered the most consolidated SEQUAL concepts, namely: *Domain*, *Model* and *Language*, and *Syntactic Quality*, *Semantic Quality* and *Pragmatic Quality* as quality aspects. These elements are also the used ones in the SEQUAL RE-oriented proposal [25] (see Fig. 5). Although *Audience Interpretation* has not been longer kept in the last SEQUAL proposals, we have kept it because it represents a relevant concept in the semiotics theory. In addition we have also considered the concept of *Model Evolution* from [26]. From the most recent proposal [27] we have included the ideas of learning, modeling and domain changes in a summarized way, that is adding the edges *Domain Evolution* and *Interpretation Evolution*. Finally, although language normally appears as static in the previous models, we have considered *Language Evolution* which is a classic assumption in the semiotics theory. It appears relevant to us because it implies to evolve also in the modeling necessities by extending the ontology that the language represents.

Since we are interested in measuring the quality of goal-oriented models, we focus next on the four dimensions of quality identified above. We have slightly reformulated some of the recent definitions of qualities from [27] with two objectives: (1) to be clear enough to facilitate the engineering process of measuring, and (2) to go back towards fundamental semiotic concepts which have been considered the relevant part of our simplified version of SEQUAL.

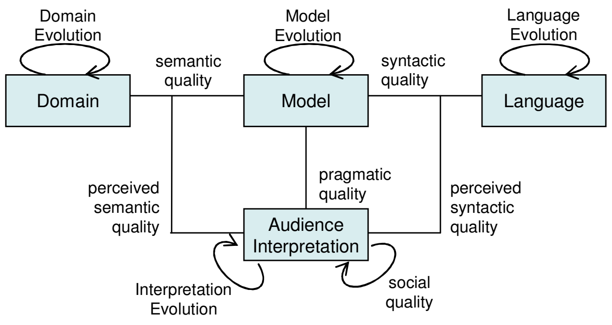


Fig. 6. A simplified SEQUAL framework for the *i** case

Syntactic quality has the goal of syntactic correctness, which means that the statements in the model must be compliant to: 1) the language syntax, e.g. task decomposition always with tasks as target, and possibly 2) some notational conventions of the modeling language, e.g., as done in [41] where some linguistic patterns are proposed.

This definition of syntactic quality has kept the SEQUAL definition that appears in this way along the evolution of SEQUAL framework. It may involve some degree of interaction as explained later, which means that it may be partially justified in terms of perceived syntactic quality.

Semantic quality has the goal of semantic validity. If we assume that semantics is the meaning of signs then it implies to relate signs (the model) to objects. Semantic validity means that the model elements have the meaning (objects) expected by the audience. From this perspective it is not possible to check semantic validity without interaction with the audience, therefore, in our practical proposal we approach semantic quality entirely as perceived semantic quality (see below).

According to this last fact, we have reformulated this definition taking out the property of completeness from the semantic part. Completeness of the model has a strong relationship with pragmatics, i.e. the objective of the attempted model. This fact has consequences on the concept of quality, because it supports separation of concerns and hence to obtain a clearer quality assessment procedure (as the one proposed in the next section).

Perceived semantic quality fully represents semantic quality in our framework, as stated above. It covers the correspondence between actors' interpretation of the model and their current knowledge about the categories of represented objects. We propose to ask for these categories in base to (at least ideally) known interpretants. Here the set of available interpretants becomes relevant, for example the same English word "goal" is the first interpretant, but also the definition of goal taken from the *i** framework or what is implied by something being a goal, for example that it must have different alternatives to be achieved or which are the questions that are addressed [39].

In our reformulation, we consider the semantic concept only as a match with the different interpretants, i.e., what a goal is supposed to be, which should be its properties, etc. Other questions about relevance of a model element or use of a model element are considered in pragmatics.

Pragmatic quality covers the correspondence between the actors' interpretation of the model and their current knowledge about the domain. It has the goal of semantic completeness and minimalism in the context of the model interpretation (use, domain implications, activations, etc.). Therefore the focus here is what the participants are interpreting from the model. For example if "Doubts solved" (Fig. 2) is a real goal of the current system (e.g., in the case of an As-Is model), if it is a needed goal, what implies that it has been recognized, what implies that it has been represented in the model, etc. Note that this question is meaningful only if it has been previously checked that "Doubts solved" is a goal.

Social quality covers the global agreement among participants about the previous measured qualities. It includes the process of negotiation of meanings and global interpretation (meanings and interpretation of the domain, model and language). When there is disagreement we talk about low quality whilst high social agreement means high quality. However, it seems clear that a total agreement could be never produced. Therefore, as a general fact, we can pretend only to evolve in agreement up to a point that does not admit more advances.

The social quality will be then the capacity to evolve on the model, the domain and the language to points of greater social agreement. As a result, these metrics will usually incorporate the time dimension in their definition. Therefore modifications to the model are not the only way of improving agreement, also it is relevant to reinterpret the domain, to modify the domain, to reinterpret the language and even to modify the modeling language, which is represented by the evolution arrow in the simplified model (Fig. 6).

Interaction. Although we have not explicitly considered interaction in the model, it is present in all quality aspects. In the case of syntax quality, interaction could be needed, even having a well-defined syntax and a parser for it, because there are notational suggestions on engineering models that may not have an easy automatic verification, e.g. that terms that appear in different elements are synonymous. In the case of semantic quality, as explained above in more detail, interaction is needed in order to match some model's elements with some of their interpretants. In the case of pragmatic quality, interaction is obviously needed, because it is not possible to have a measure of the perceptions without asking for these perceptions. Finally, in the case of social quality, if we consider the interaction of semantic and pragmatic levels then it could not be necessary having additional interaction in order to obtain social metrics. However, if we pretend to improve the agreement level about models, new interactions are the only way.

In spite of the mentioned adjustments, the conceptual baseline still comes from SEQUAL and this is the reason why we have labeled it as *simplification* instead of *variation*.

3.2. Application to the *i** Framework

Considering on the one hand the *i** language and their different models, and on the other hand this SEQUAL simplified framework, we propose to connect both proposals in order to have a practical application of a well-founded quality requirements proposal such as SEQUAL, applied to a widespread requirements modeling framework such as the *i** framework, with the purpose of improving the quality of goal-oriented requirements models.

At this point we use again the Kavakli's abstraction [1] to disaggregate the goal-oriented requirements process. It allows to obtain the models As-Is, Change and To-Be. The As-Is model represents a current state of the organizational domain involving human actors, their dependencies and their intentions. The Change model represents a modified organizational system considering the new system as an actor placed in the human system and connected to human actors through intentional dependencies. The To-Be system exclusively represents the new computational system in order to specify their goals and the strategy to pursue its goals. All these models are expressed by means of SD and SR diagrams. They could include specific tasks, resources or even quality features or quality constraints (normally in the form of softgoals).

Therefore, we have different models and also different qualities to measure, as illustrated in Fig. 7. It may be noticed that the qualities are proposed to be checked through some measuring points in a given ordering. The ordering is not accidental. Just as semantics cannot be verified without previous syntax verification, pragmatics cannot be verified without previous semantic verification (e.g., how can we verify the relevance of the goal G, if G is not a goal?). Thus, the verification ordering is established as: starting from syntax, then semantics and then pragmatics. As a consequence, specific semantic interaction on a correct syntax model can evolve the model into an improved semantic version and, moreover, a specific pragmatic interaction on a correct semantic portion of the model can evolve the model into an improved pragmatic version. Finally social quality can be tackled when individuals have their own interpretation of the domain, the model, and the modeling language. Concerning models, Kavakli's order is the obvious choice: first As-Is, then Change, finally To-Be.

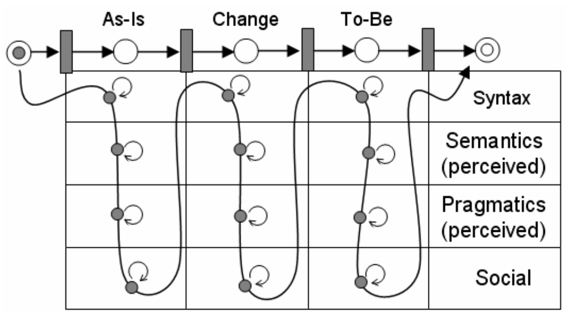


Fig. 7. The space of qualities to measure

In order to show how our simplified SEQUAL framework can be applied to *i** models, we have built Table 1, where the main concepts of the simplified SEQUAL framework are matched with the appropriate language constructors of the *i** modeling language.

As a natural consequence of the role that perception plays in our framework, most identified qualities need a certain level of interaction in order to be determined. An exception is syntactic quality that can become automatic when there is a complete formal specification. We do not consider syntactic quality in this paper due to its simplicity, since most of this quality concern can be solved used some iStarML parser for the syntax itself, and a simple natural language checker for language conventions.

Table 1. Correspondence between *i** and the simplified SEQUAL framework

Simplified SEQUAL concept	*i** framework
The model (M) represents the domain	The *i** language is oriented to organizational modelling including socio-technical systems. Therefore it can represent the domain As-Is, Changed and including the system To-Be.
The audience (A) represents organizational actors and technical actors. It is a subset of the stakeholders	It models the concept of actor. In fact, a complete *i** model should have all stakeholders represented in the domain model. The *i** language allows representing actors and there is a special language constructor to delimitate the organizational scope namely *boundary*.
The language (L) can be divided into formal, semi-formal and informal	The *i** language and the iStarML format constitute semi-formal languages. Informal language would be natural language in the process of requirements engineering.
Audience interpretation (I). There are as many interpretations as actors.	Not considered. Although *i** can represent intentionality, it does not allow specifying that the specified intentionality is the perception of a specific modeller or informant.

Therefore we will focus on interaction-based measurements to calculate some metrics that may serve as indicators to semantic quality, pragmatic quality and social quality. Table 2 summarizes some examples of metrics applied to the example of Fig. 2.

Semantic metrics point to approach semantic quality, which means to measure the relationship between the interpretants and the model symbols. It can be only checked by asking for the precise match among them. We envisage four possibilities (semantic quality interaction type of patterns) here: (i) to try to match a symbol (instance of an *i** model element) with the set of their interpretants, which can be a "good" indicator; (ii) to try to match a symbol with some set of other interpretants, which can be a "bad" indicator (for example to try to match a goal with the task definition); (iii) to try to match the symbol's properties with some of their interpretant's properties, which can be also a "good" indicator (for example checking that a specific softgoal represents a fuzzy concept) and, finally (iv) to try to match symbol's properties with other different properties, which can be a "bad indicator" (for example if someone said that a softgoal can have a mean that implies that the softgoal is completely achieved).

From a formal point of view, we can say that: having a set P of semantic quality interaction patterns (like the above ones), a set of interpretants I, and a set of symbols S which correspond to a language constructor (or group of language constructors) of the modeling language L, then m is a semantic metric iff the objective of m is to measure the gap between each s $\in$ S and the set of its interpretants of s, namely i(s) $\in$ I, i.e. m is a function m: $\{s\} \times i(s) \rightarrow V$, where V is a set of quality values. Thus m' is a measure that approaches m iff m' is a function m': $\{p\} \times \{s\} \times \{i\} \rightarrow V$, where p $\in$ P and i $\in$ i(s).

Table 2. Some examples of metrics for the tutoring *i** model

Name	Definition
Semantics metrics	
Proportion of tasks that are considered tasks	Amount of positive matches between model's tasks with corresponding interpretants over the total amount of negative and positive matches.
Proportion of resources that are not considered resources	Amount of positive matches between model's resources with non-corresponding interpretants over the total amount of negative and positive matches.
Pragmatics metrics	
Proportion of non-relevant tasks in the model	Amount of positive answers to negative considerations of tasks in the model over the total amount of considered tasks.
Proportion of unconsidered relevant actors	Amount of unidentified relevant actors over the total of actors in the model
Social metrics	
Agreement level about goals in the model are really goals	Average of level of agreements (percentage) on the individual considerations of each goal in the model
Velocity of agreement on domain models	Average over the total amount of time that the group has arrived to an As-Is model over the 80% of agreement in past modeling activities.

Pragmatics metrics point to measure pragmatic quality, which means that there is a match between object elements and symbols. Objects are assumed to be part of the reality of each participant. A high pragmatic quality means that the model is complete and has only the necessary modeled objects, i.e. we should consider to model only "relevant" objects. As in the previous case, interaction is absolutely necessary to obtain what is relevant to each participant. We envisage three basic ways (pragmatic quality interaction type of patterns): (i) to try to match a relevant object with an existing symbol in the model, which can be a "good" indicator; (e.g., to try to match a domain task that needs computational support with a task in the model); (ii) to try to match a irrelevant object with an existing symbol in the model, which can be a "bad" indicator (e.g., to identify a resource in the model which has not implications over the social organization); (iii) to try to identify a relevant object in the domain which is not present in the model, which can also be a "bad" indicator, (e.g., to identity an actor which should change its practices with the new software system).

From the formal point of view, we can say that: having a set P of pragmatic quality interaction patterns (like the above ones), a set O of perceived objects from the domain, and a set of symbols S which corresponds to a language constructor (or group of language constructors) of the modeling language L, then m is a semantic metric iff the objective of m is to measure the gap between s $\in$ S and the set of objects from domain that could be represented by s, namely o(s) $\in$ O, i.e. m is a function m: $\{s\} \times o(s) \rightarrow V$, where V is a set of quality values. Thus m' is a measure that approach m iff m' is a function m': $\{p\} \times \{s\} \times \{o\} \rightarrow V$, where p $\in$ P and o $\in$ o(s).

Social metrics point to measure the social quality, which means that there is a match between the interpretation (reading) of domain, model and language. A high social quality means that there is a general agreement about individual interpretations. Therefore we can approach at least high social quality through the level of agreement and the dynamics of the quality evolution on the base of the answers already formulated in the semantic and syntactic pattern-based interactions.

From a formal point of view we can say that: having a set of functions M of metrics, a set P of participants, a set V of quality values, an ordered set T of time values, and the definition of A^+ as the power set of A without the empty set, then we define a social metric m as a function m: $T^+ \times M^+ \times P^+ \rightarrow V$. Therefore social metrics will be focused on the level of agreement (of some community of participants) with respect to symbols or group of symbols in the model, e.g. level of agreement in goal representation (semantic perspective), level of agreement in relevance of goals in the model (pragmatic perspective), etc. Besides, the social dynamics of evolution is also part of social metrics, e.g. Average of needed amount of model versions to reach agreement levels over 80%.

Therefore the proposed general social quality interaction patterns depend on the access and evolution of the audience to the reasons of others. High agreement and fast convergence will mean high confidence and good social learning (high social quality).

Finally, we propose to trigger interaction by questions to the users of the models; the already established correspondence between metrics and quality patterns enable the feasibility to obtain a set of values which can be used to approach a metric value (depending on the selected scale). The model evolution will occur when we correct the model in order to minimize “bad” measurements (not modeled elements) or maximize “good” measurements (modeled elements).

3.3. Implementing quality measurements

In this section we show how to implement the proposed simplified SEQUAL framework for *i** over the previous tutoring system example (Fig. 2). We focus on the Change stage, where some needs have already been established (by modelers) but there is not yet a software system taking any specific responsibility. We assume that the model of Fig. 2 is a first version of the change model, therefore the issue of quality for improving (evolving) the model becomes crucial.

To implement the framework applied to semantic metrics we propose the following steps that can be taken iteratively: (1) to select a type of symbol *s* of the modeling language, i.e. an *i** element (as “goal”) or group of elements (such as “intentional element”); (2) to define a metric *m* on the basis of *s* and its generic interpretants, using techniques as GQM [42]; (3) to identify the interpretants *i*(*s*) of *s*, i.e. definitions or properties; (4) to define a set of measures on the base of *s* and *i*(*s*); (5) to select one or more semantic quality interaction patterns; (6) to define the values of measurement to be obtained; (7) to produce a high-level parameterized query that, by applying this pattern, asks for a particular quality value; (8) to generate the implementation of that high-level query.

In the last step, the iStarML format comes into existence: metrics may be applied on models represented in *i**, typically generating a XQuery sentence.

To implement the framework applied to pragmatic metrics we propose the same steps than before but, instead of considering the pair symbol-interpretant, we consider the pair symbol-object, where object is a relevant object in the domain, and hence it must be present in the model.

Table 3. Example of metric implementations

Step	Select	Example of Semantic Metric	Example of Pragmatic Metric
(1)	Symbol	*i** goal	i* goal
(2)	Metric	Proportion of goals that effectively are goals	Proportion of main goals placed in wrong actors
(3)	Pair symbol-x	Interpretant: (From [5].) A goal must be reached by some of its means	Object: A “wrong” (to the community) goal of the actor
(4)	Measure	Proportion of goals in the model that have identified means which are not enough to ensure satisfaction of the goal	Proportion of main goals by each specific actor which have identified other actors where they (goals) could be placed
(5)	Pattern type	Type (i)- by trying to match a goal with some of its interpretants	Type (ii)- by trying to match a goal which belongs to other actor
(6)	Scale	yes/no	yes/no
(7)	Query	Is it possible to reach *%goal* when *%means attached to the goal*	Is %goal one of the main goals of the %wrong actor

(8)	Xquery	for g$ in document("Tutoring.istarml")
		//ielement[@type="goal"]
		LET m$:=g$/ielementLink[@type="means-end"]
		/ielement[@name][@type]
		return <question type="yes-no">
		It is possible to reach {g$/@name} when
		{m$/@name} ?</question>
			for $a in xmlmem($bibo)//actor
			for $b in xmlmem($bibo)//actor
			for $g in $a/boundary/ielement[@type="goal"]
			where $b/@name<>$a/@name
			return <question type="yes-no">
			Is {$g/@name}a relevant goal for the {$a/@name}?

Interaction example		Is it possible to reach *Doubts solved* when *Solve doubts by email*?	Is *Knowledge of courses acquired* one of the main goals of the *Tutor?*

To implement the framework applied to social metrics we propose the following steps: (1) to select a set of time instances, for example next Monday; (2) to identify the group to measure, for example the tutors p1, p2 and p3; (3) to identify the metrics to apply in defined time instances, e.g. “Proportion of goals that effectively are goals”; (4) to identify the measurements collected, for example m’1: *Proportion of goals in the model that have identified a false means* and m’2: *Proportion of goals that have been identified as reachable objective*; (5) to collect the measurements grouped by symbol and by time. A valid sequence could be:

<"Doubts solved", p1, Monday(<m'1=yes, m'2=yes>)>
<"Doubts solved", p2, Monday(<m'1=no, m'2=yes>)>
<"Doubts solved", p3, Monday(<m'1=yes, m'2=yes>)>
<"Other functions done", p1, Monday(<m'1=yes, m'2=yes>)>

(6) to define and apply the social metric function, for example average of agreement in all symbols. So, if we have the agreements m'1("Doubts solved") = 66%, m'2("Doubts solved") = 100%, m'1("Other functions done") = 33%, and m'2("Other functions done") = 80%, then the final result is 70%.

Finally, to conclude the technical proposal, in Table 3 we have summarized these steps applied to produce two metrics: a semantic metric namely "proportion of goals that effectively are goals" and the pragmatic metric namely "proportion of main goals placed in worng actors". In both cases we have used the tutoring system example and the XQuery sentence assumes that the model representation is in iStarML.

4. Conclusions and future work

In this paper we have proposed a theoretical and practical framework for quality assessment of goal-oriented models at the RE stage that may eventually help on improving the quality of deliverables of each of the internal phases identified by Kavakli [1]. To do so, we have complemented the *i*MDF framework for defining metrics on *i** with two already existing proposals. Firstly, we have considered the quality assessment and measurement proposal named SEQUAL. We have configured a simplified version of SEQUAL based in its semiotic principles and analysed its meaning in *i**. As part of this analysis, we have illustrated how to elicit semantic, pragmatic and social metrics for providing a practical framework of question patterns to obtain goal-oriented process measurement values. Secondly, in order of being able to implement these patterns easily, we have used the iStarML as language for representing *i** models making then possible the expression of the patterns as XML-based queries, which is a very simple way to implement the framework, far more than our previous versions using OCL. Besides, the use of iStarML allows applying the quality framework also over existing *i** variants because these can be represented by iStarML too. Even as future work, one could think of abstracting iStarML into something like GOREontologyML to host KAOS and other formalisms.

In fact, we claim that our work is not just a contribution for the *i** community or even the GORE community, but to the RE community in general. We sustain this claim by considering the dissertation given by Abran et al. about the existing gap between theoretical frameworks for requirements measuring and requirements metrics [43]. This work shows a comparative study between measuring engineering frameworks and the SWEBOK (Software Engineering Body of Knowledge) initiative. As a way of summarizing the findings, they have built a table crossing knowledge areas (from SWEBOK) and a generic four-step generic measurement model. It is shown that RE topics do not present research activity on the first three steps of the model that are: design of measurement methods, application of measurement methods and analysis of measurement results. The fourth step, in which some activity has been recognized, deals with exploitation of measurements.

In terms of future work we have planned to cope two new theoretical problems: (1) to quantify the amount of interaction needed to obtain trustable goal-oriented process metrics; (2) to propose ways of enriching interaction in order to allow a more rapid evolution of models and social quality. Our next step in this sense is to confirm these theoretical problems testing this practical framework on some study cases and also get some feedback of the assumed benefits.

5. Acknowledgments

This work has been partially supported by the Spanish project ref. TIN2007-64753.

6. References

[1] E.Kavakli, "Goal Oriented Requirements Engineering: A Unifying Framework", *Informatica*, vol. 6, pp. 237-251, 2002
[2] A. v. Lamsweerde and E. Letier, "From Object Orientation to Goal Orientation: A Paradigm Shift for Requirements Engineering", *LNCS (RISSEF'02)*, vol. 2941, pp. 325-340, 2004.
[3] E. Yu and J. Mylopoulos, "Why goal-oriented requirements engineering?", Proc. of the 4th Int.Workshop on Requirements Engineering, 1998, pp. 15-22.
[4] N. Maiden, "What has requirements research ever done for us?", *IEEE Software*, vol. 22, pp. 104-105, 2005.
[5] E. Yu, "Modelling Strategic Relationships for Process Reengineering", PhD Thesis, Computer Science D., University of Toronto, 1995.
[6] P. Donzelli and P. Bresciani, "Domain Ontology Analysis in Agent-Oriented Requirements Engineering", *LNCS (KES'03)*, vol. 2773, pp. 1372 - 1379, 2003.
[7] M. Kolp, P. Giorgini, and J. Mylopoulos, "Organizational Patterns for Early Requirements Analysis", *LNCS (CAiSE'03)*, vol. 2681, pp. 617-632, 2003.

[8] P. Donzelli, "A goal-driven and agent-based requirements engineering framework", *Requirements Engineering*, vol. 9, pp. 16-39, 2004.
[9] P. Donzelli and R. Setola, "Handling the knowledge acquired during the requirements engineering process -a case study-", Proc. of the 14th Software Engineering and Knowledge Engineering Conference (SEKE'02), Ischia, Italy, 2002, pp. 673-679.
[10] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology", *Autonomous Agents And Multi-Agent Systems*, vol. 8, pp. 203-236, 2004.
[11] J. Castro, M. Kolp, and J. Mylopoulos, "Towards requi-rements-driven information systems engineering: the Tropos project", *Information Systems*, vol. 27, pp. 365-389, 2002.
[12] F. Massacci, J. Mylopoulos, and N. Zannone, "An Ontology for Secure Socio-Technical Systems", in *Handbook of Ontologies for Business Interaction*, P. Ritggen, Ed.: IDEA Group, 2007.
[13] H. Mouratidis, M. Weiss, and P. Giorgini, "Security Patterns Meet Agent Oriented Software Engineering: A Complementary Solution for Developing Secure Information Systems", *LNCS (ER'05)*, vol. 3716, pp. 225 - 240, 2005.
[14] E. Yu, "Why Agent-Oriented Requirements Enginee-ring", Proc. of the 3rd Int. Workshop on Requirements Engineering: Foundations for Software Quality, Barcelona, Spain, 1997.
[15] B. W. Boehm, *Software engineering economics*. Englewood Cliffs, N.J.: Prentice-Hall, 1981.
[16] R. J. Costello and D.-B. Liu, "Metrics for Requirements Engineering", *Systems Software*, vol. 29, pp. 39-63, 1995.
[17] O. C. Z. Gotel and C. W. Finkelstein, "An analysis of the requirements traceability problem", Proc. of the 1st Int. Conf. on Requirements Engineering, Colorado Springs, CO, USA, 1994, pp. 94-101.
[18] N. Nurmuliani, D. Zowghi, and S. Powell, "Analysis of requirements volatility during software development life cycle", Proc. of the Australian Software Engineering Conference, 2004, pp. 28-37.
[19] R. Matulevicius, "How Requirements Specification Quality Depends on Tools: A Case Study", *LNCS (CAiSE'04)*, vol. 3084, pp. 353-367, 2004.
[20] H. Estrada, A. Martínez, O. Pastor, and J. Mylopoulos, "An Experimental Evaluation of the i* Framework in a Model-based Software Generation Environment", *LNCS (CAiSE'06)*, vol. 4001, pp. 513-527, 2006.
[21] X. Franch, "On the Quantitative Analysis of Agent-Oriented Models", *LNCS (CAiSE'06)*, vol. 4001, pp. 495-509, 2006.
[22] X. Franch, "A Method for the Definition of Metrics over i* Models", *LNCS (CAiSE'09), in print*, 2009.
[23] X. Franch and G. Grau, "Towards a Catalogue of Patterns for Defining Metrics over i* Models", *LNCS (CAiSE'08)*, vol. 5074, pp. 197-212, 2008.
[24] O. I. Lindland, G. Sindre, and A. Solverg, "Understanding quality in conceptual modelling", *IEEE Software*, vol. 11, pp. 42-49, 1994.
[25] J. Krogstie, O. I. Lindland, and G. Sindre, "Towards a Deeper Understanding of Quality in Requirements Engineering", *LNCS (CaiSE'95)*, vol. 932, pp. 82-95, 1995.
[26] H. D. Jorgensen, "Interactive Process Models", PhD Thesis, D. of Computer and Information Science, Norwegian University of Science and Technology 2004.
[27] J. Krogstie, G. Sindre, and H. Jørgensen, "Process models representing knowledge for action: a revised quality framework", *European J. of Information Systems*, vol. 15, pp. 91-102, 2006.
[28] C. Cares, X. Franch, A. Perini, and A. Susi, "iStarML: An XML-based Model Interchange Format for i*", Proc. of the 3rd Int. *i** Workshop, Recife, Brazil, 2008, pp.
[29] A.v.Lamsweerde, "Requirements Engineering in the year 00: A Research Perspective", Proc. of the Int. Conf. on Software Engineering (ICSE'00) Limerick, Ireland, 2000, pp. 5-19.
[30] J. Mylopoulos, L. Chung, and E. Yu, "From Object-Oriented to Goal-Oriented Requirements Analysis", *Communications of the ACM*, vol. 42, pp. 31-37, 1999.
[31] "GRL - Goal Oriented Requirement Language", http://www.cs.toronto.edu/km/GRL/, last visited Apr. 2009.
[32] "Z.150: User Requirements Notation (URN) - Language requirements and framework", http://www.itu.int/rec/ recommendation.asp?type =folders&lang =e&parent= T-REC-Z.150, last visited Apr. 2009.
[33] C. Ayala, C. Cares, J. P. Carvallo, G. Grau, M. Haya, G. Salazar, X. Franch, E. Mayol, and C. Quer, "A Comparative Analysis of i*-Based Agent-Oriented Modeling Languages", Proc. of the Conf. on Software Engineering and Knowledge Engineering (SEKE2005), 2005, pp. 43-50.
[34] A. Susi, A. Perini, J. Mylopoulos, and P. Giorgini, "The Tropos Metamodel and its Use", *Informatica*, vol. 29, pp. 401-408, 2005.
[35] C. W. Morris, *Signs, language and behavior*. New York: Prentice-Hall, 1946.
[36] U. Eco, *La estructura ausente: introducción a la semiótica*, 3rd ed. Barcelona: Editorial Lumen, 1986.
[37] P. B. Andersen, "Dynamic semiotics", *Semiotica*, vol. 139, pp. 161-210, 2002.
[38] U. Eco, *Tratado de Semiótica General*. Barcelona: Lumen, 1995.
[39] K. Pohl, "The three dimensions of requirements engineering: A framework and its applications", *Information Systems*, vol. 19, pp. 243-258, 1994.
[40] J. Krogstie and H. D. Jorgensen, "Quality of interactive models," Proc. of the 1st Int. Workshop on Conceptual Modelling Quality (IWCMQ'02), Tampere, Finland, 2002, pp. 115-126.
[41] X. Franch, G. Grau, E. Mayol, C. Quer, C. Ayala, C. Cares, F. Navarrete, M. Haya, and P. Botella, "Systematic Construction of *i** Strategic Dependency Models for Socio-technical Systems," *Int. J. of Software Engineering and Knowledge Engineering*, vol. 17, pp. 79-106, 2007.
[42] V. R. Basili, G. Caldiera, and H. D. Rombach, "The Goal Question Metric Approach", in *Encyclopedia of Software Engineering*, G. Caldiera and D. H. Rombach, Eds. New York City: John Wiley and Sons, 1994, pp. 528-532.
[43] A. Abran, A. Sellami, and W. Suryn, "Metrology, Measurement and Metrics in Software Engineering", Proc. of the 9th Software Metrics Symp., Sydney, Australia, 2003, pp. 2-11.