

A core ontology for requirements

Ivan J. Jureta^{a,c,*}, John Mylopoulos^b and Stéphane Faulkner^a

^a *Louvain School of Management, University of Namur, Namur, Belgium*

^b *Department of Computer Science, University of Toronto, Toronto, Canada*

^c *Fonds de la Recherche Scientifique – FNRS*

Abstract. In their seminal paper (*ACM T. Softw. Eng. Methodol.*, 6(1) (1997), 1–30), Zave and Jackson established a core ontology for Requirements Engineering (RE) and used it to formulate the “requirements problem”, thereby defining what it means to successfully complete RE. Starting from the premise that the stakeholders of the system-to-be communicate to the software engineer the information needed to perform RE, Zave and Jackson’s ontology is shown to be incomplete, in that it does not cover all classes of basic concerns – namely, the beliefs, desires, intentions, and evaluations – that the stakeholders communicate. In response, we provide a new core ontology for requirements that covers these classes of basic stakeholder concerns. The proposed new core ontology leads to a new formulation of the requirements problem. We thereby establish a new framework for the information that needs to be elicited over the course of RE and new criteria for determining whether an RE problem has been successfully addressed.

Keywords: Requirements engineering, ontology for requirements engineering, requirements problem

1. Introduction

Software engineering is “the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software” (Abram & Moore, 2004). At the very outset of any software development project, requirements are elicited, analyzed, validated, and specified in order to improve as much as feasible the understanding of the real world goals for, functions of, and constraints on the software system-to-be (Zave, 1997). To say that requirements are *engineered* is currently more of an ideal than the actual state of affairs in the requirements engineering (henceforth RE) field. In fact, the rigor, discipline, and precision with which requirements are manipulated vary strongly, depending to a large extent on the maturity of the requirements in a given project. Once requirements are, among others, precise and stable enough, and agreed upon, they become amenable to mathematical formalization, and can be subjected to the engineering approach. It is rather unsurprising then that the actual development of RE is skewed towards the study of these so-called *late requirements*. Dealing with such requirements at least has the benefit of being an engineering problem where both the tools and the material are understood well enough to warrant novel contributions over rather stable theoretical foundations.

Late requirements are distilled from *early requirements*, to which they stand in stark contrast. Obtained via communication with the stakeholders of the system-to-be, early requirements take the form of variously detailed, potentially ambiguous and vague natural language statements given without much consideration for some shared vocabulary about the system-to-be or its relevant surroundings. Eliciting,

*Corresponding author: Ivan J. Jureta, Louvain School of Management, Namur Campus (FUNDP), Rempart de la Vierge 8 (b. 423), B-5000, Namur, Belgium. E-mail: ivan.jureta@fundp.ac.be.

modeling, bringing together disparate and incomplete early requirements, and obtaining late requirements from there via an engineering approach is a longstanding concern in RE. Difficulty lies not only in discovering which engineering methods to devise in order to guide the performance of the said tasks, but also in establishing what kinds of information these methods are supposed to manipulate in the first place. This last issue is the focus of the present paper.

A decade ago Zave & Jackson (1997) observed that the RE field left behind simplistic approaches to understanding what a system will do in favor of novel and varied terminology, methods, languages, tools and issues considered to be critical. Their discussion was seminal for they defined a core ontology that established fundamental distinctions for the kinds of information that needs to be elicited over the course of any RE project. This in turn allowed them to formulate the “requirements problem”, which determines exactly what it means for RE to be successfully completed. Various interpretations of their core ontology have since been suggested and used. There is in fact limited consensus on the precise meaning of the core ontology in RE. Rather, a requirement is variously understood as (describing) a purpose, a need, a goal, a functionality, a constraint, a quality, a behavior, a service, a condition or a capability. Terms such as “nonfunctional requirement”, “softgoal”, “preference”, “priority” only add to the confusion. It is difficult to compare or combine contributions, and identify conceptual overlaps.

The only relevant core ontology for requirements is one which helps the software engineer in solving the requirements problem. Zave & Jackson’s (1997) proposal provided an elegant characterization of the requirements problem by relying on three concepts that constitute their core ontology. A “requirement” is an optative (i.e., desired) property of the environment, which includes the system-to-be and its relevant surroundings. A “domain assumption” is an indicative property, describing the environment as it is and in spite of the system-to-be. An optative property, intended to be directly implementable and support the satisfaction of requirements, is a fragment of a “specification”. From there on, Zave and Jackson suggest that *the requirements problem amounts to finding the specification S that for given domain assumptions K satisfies given requirements R* . If all three are written in a mathematical logic, the problem is solved once the engineer finds S such that $K, S \vdash R$. This characterization is, however, deficient for the following reasons:

1. Satisfaction in $K, S \vdash R$ is binary: K and S cannot satisfy R to some extent only. This is an oversimplification, for it is impossible to consider the software resulting from S as being of higher or lower quality – its quality is instead either acceptable ($K, S \vdash R$) or unacceptable ($K, S \not\vdash R$). In stark contrast to reality, it is impossible for stakeholders (i.e., users, owners, etc.) to be more or less satisfied with the system.
2. The proposed formulation leads us to conclude that any two distinct specifications S_1 and S_2 , such that $K, S_1 \vdash R$ and $K, S_2 \vdash R$, are equally desirable. This is again an oversimplification: e.g., say that the stakeholders prefer lower response times to a query and certainly expect the response time to be below 2 s, then if S_1 and S_2 result in, respectively, systems that respond to the query in less than 1 s and less than 0.5 s all else being equal, we will evidently choose S_2 over S_1 .
3. The proposed characterization is overly generic to be of use, e.g., in guiding the construction of RE (modeling) languages, that is, formalisms for the representation of information relevant during RE. While high abstraction is certainly desired, we see from points 1 and 2 above that Zave and Jackson’s formulation stays at too high a level. We cannot say with K , S , and R alone, if the set of concepts in one RE language is more appropriate to resolve the requirements problem than that of another language.
4. The formulation of the requirements problem depends on the concepts chosen for the core ontology, and *vice versa*: the understanding of the requirements problem influences the choice of concepts for

the core ontology. To escape this circularity, Zave and Jackson draw on common sense and experience. While this cannot be avoided altogether, moving towards *stable* conceptual foundations and problem formulation requires criteria for acceptance that are less dependent on the reader's opinion and the authority of the authors.

Our aim in this paper is to suggest a core ontology for requirements (henceforth CORE) that spans much of ongoing research in RE and is grounded in the communication between the stakeholders and the software engineer (Sections 2–5). This allows us to provide a new formulation of the requirements problem (Section 6) that responds to the limitations of Zave and Jackson's formulation. An exemplified walkthrough (Section 7) illustrates the instantiation of the concepts that constitute the ontology. We subsequently relate our results to comparable efforts (Section 8). The paper closes with a summary of conclusions and pointers to important directions for future inquiry (Section 9).

2. Baseline

“To study the most general features of reality and real objects” (Peirce, 1935) is to deal in matters of ontology in philosophy, entailing questions of what kind of entities exist, or what relationships there are between entities. A shared understanding of what exists, what properties it has, and what relationships it participates in, underlies any domain of inquiry – together, these form the conceptualization that acts as a foundation for knowledge sharing in the domain (Genesereth & Nilsson, 1987). Following the tradition of knowledge engineering, an ontology is an explicit specification, account, or representation of (a part of) a conceptualization (Gruber, 1993). An ontology will at least include a vocabulary of terms and some (more or less precise) delimitation of their intended meaning (Uschold & Gruninger, 1996). Given that a conceptualization embodies a particular view or understanding of the domain of interest, an ontology in the RE domain plays the role of a facilitator when comparing, integrating, or extending existing, and advancing new conceptualizations.

The purpose of this section is to explain the important choices we made when designing CORE. We start by discussing how we chose what goes in and what stays out of CORE (Section 2.1). We then discuss the consequences of the speech act-based understanding of communication on the design of CORE (Section 2.2), and recall the foundational ontology, to which CORE is related (Section 2.3).

2.1. Determining what goes in and what stays out

It is important at the outset to observe the distinction between calling an ontology minimal and/or core. Given the purpose at hand, which in our case is the design of an ontology used to characterize the requirements problem, a core ontology will carry only the necessary components that are needed to fulfill this purpose. A core ontology will be minimal *with regards to its purpose* only if it cannot be reduced further, that is, if none of its components can be redefined in terms of others. In this respect, it is important to avoid the misconception that a core ontology is minimal *with regards to its domain or field of use*. CORE is designed as a core ontology, and is minimal with regards to its purpose, but is not minimal for the entire RE field. While it will be apparent from the rest of the paper that CORE does span much of research in RE, we neither can, nor intend to argue that it spans *all* of research into the concepts and relationships relevant for RE. Being a core ontology, CORE incorporates only those concepts that are *necessary* for any knowledge representation formalism to carry if it is to be relevant in any way in the

resolution of the requirements problem. We argue so because of the method we chose in selecting the concepts that should go into CORE, and which we explain below.

Not all criteria are equally suited to guide the choice of the concepts an ontology for requirements. An obvious criterion is *intuitiveness*, whereby a concept is intuitive if its relevance seems self-evident from the very definition or understanding of the problem at hand. E.g., there is no doubt that a system-to-be will be transforming something in a given (human) setting, so that at least one concept is necessary for the representation of the anticipated transformations. Hence the use in RE of the *scenario*, *task*, *action* and/or *process* concepts, to name a few. Another common criterion for relevance is the practical value, or *utility* of concepts and relationships. E.g., in explaining the relevance of the goal concept, van Lamsweerde (2001) argues that goals can (i) provide a criterion for a sufficient completeness of a requirements specification, (ii) help in avoiding irrelevant requirements, (iii) facilitate the explaining of requirements to stakeholders, (iv) support the structuring of complex requirements documents in order to improve readability, (v) support conflict management, (vi) be the basis of a systematic requirements elaboration process, and so on. Intuitiveness and usefulness together lead to an approach where a concept is introduced for its intuitiveness, whereby its usefulness is evaluated through experience in order to reinforce or weaken the conviction that the concept is relevant. Where both intuitiveness and usefulness fail, however, is that it is impossible to use (either one individually or the two together) these criteria for *delimiting* the scope of an ontology for requirements. Failure arises from the fact that both very strongly rely on experience and common sense. The problem with the former in RE is that there is very little rigorous experimental work into the relevance of modeling primitives, not least because of the very real methodological and practical difficulties in acquiring rigorously the necessary observations. As for common sense, despite the insight it provides, it depends on personal factors that are hard to pin down. The fact that the “goal” concept itself is variously interpreted in RE research illustrates that what appears to be common sense to some is not so to others.¹

Intuition and usefulness were not the starting point in the design of CORE. Instead, it was a simple observation about the very process of information acquisition in RE, which is that *requirements-related information is elicited through the communication between stakeholders and software engineers*. This assumption makes the very methodology of choosing concepts and relationships different from the standard “come up with something intuitive, then check if useful” approach. To understand this shift, it is relevant to recall that RE developed as a subfield of software engineering, which makes it unsurprising that RE frameworks continue to be developed by drawing on software specification methods. An important side effect of this is that the role of the software engineer in applying software specification methods was carried over in the application of the RE frameworks. Now, applying software specification methods requires specific knowledge that is almost always well beyond the expertise and/or interest of the stakeholders. Stakeholders therefore have a very restricted role when software is being specified. The problem from our perspective is that reasoning by analogy led the RE field to carry over this role to RE. It follows that the software engineer is given during RE the communicated requirements-related information on one hand, and some “intuitive and useful” concepts and relationships on the other. The task then amounts to fit what the stakeholders communicate during elicitation to RE conceptualizations. In other words, the aim is to see what bits and pieces of communicated requirements instantiate which concepts and relationships. The problematic point is that stakeholders have no role to play in the choice of concepts and relationships. It might seem odd in RE to have stakeholders influence the choice of the components of an ontology, but this is exactly what happens in CORE.

¹Jureta et al. (2007) survey the literature and identify and discuss a number of definitions of the “goal” concept. A conclusion of that work that is important for the present discussion is that these definitions differ in the interpretation of that concept.

We do not suggest that the way to engage the stakeholders in the choice of conceptualizations is to make assumptions or attempt to figure out how they may understand the system-to-be and its surroundings. In other words, it is not to peek inside minds, because doing so brings us back where we started: namely, the case where the designer of a representation formalism for RE extrapolates that what is intuitive to her is so to others. Instead, a feasible solution is to design conceptualizations based on patterns of how stakeholders *reveal* their understanding of the system-to-be and its surroundings during RE. Because requirements-related information is elicited through the communication between the stakeholders and the software engineers, *it is via communication that stakeholders reveal their understanding of the system-to-be and its relevant surroundings*. To ground an RE ontology in how stakeholders reveal their understanding of the system and its environment is to ground the ontology in a conceptualization of the communication process. This calls for an altogether different approach to the design of RE conceptualizations and to the role of the software engineer during early RE. Namely, *the scope of RE conceptualizations should be determined by how and what can be communicated by the stakeholders during requirements elicitation*. Once the concepts are defined to cover the scope and depth of communicated information, then the very act of communicating during elicitation determines the classification of that communicated content within an ontology for RE. It is no longer the software engineer who chooses the classification of the communicated information. It is instead the stakeholder who determines if the communicated content is a goal or otherwise, and this by the very act of communicating it. It is also very convenient that for this to work, the stakeholder need know nothing of the RE method or representation formalism for RE.

In selecting and defining the components of CORE, we start from the simple premise that the RE process is one in which *stakeholders communicate information to the software engineer*, whereby stakeholders include people, but also, e.g., legacy systems, which are documented. The software engineer's task is to classify information as requirements or otherwise (e.g., domain assumptions), then use it when searching for an appropriate specification. While this observation appears to convey a straightforward understanding of the overall RE process, it has important consequences on our core ontology, and is the key source of novelty with regards to related research. *Utterances that stakeholders make in communicating with the engineer are actions intended to advance stakeholders' personal desires, intentions, beliefs, and evaluations*, in the aim of ensuring that the engineer can produce a specification that then leads to a system responsive to the communicated concerns. It ensues that, *if we know the various kinds of communicative actions at the disposal of stakeholders, we can delimit the scope of the core ontology to concepts that allow us to refer to all communicated concerns*.

The scope of CORE is such that the ontology carries only those concepts that are necessary to cover well established kinds of communicative actions. This leads to the discussion below (Section 2.2), in which we draw on research in linguistics in order to enumerate the kinds of communicative actions that have been established as present in communication.

To suggest an ontology is to suggest and argue for the appropriateness of a series of definitions. Since it is more the exception than the rule to agree on conceptualizations, it is necessary to ensure a certain degree of rigor for the definitions that we will suggest. Our second discussion below (Section 2.3) outlines the strategy that we chose to provide an acceptable degree of rigor to our definitions. In addition to arguments against the dominant contemporary conceptualization and those in favor of our proposal, we choose to discuss CORE in relation to a foundational ontology. This means that the ontological nature of our concepts is given in terms of the categories of the chosen foundational ontology. We therefore outline alternative foundational ontologies, then argue for one among these as the foundational ontology for CORE.

2.2. Delimiting the scope via speech acts

Content of communication can be conveyed in different ways by the speaker, entailing different effects on the hearer. This is reflected in linguistics by the separation, in any given statement, of the *dictum*, or what is said, from the *modus*, or how it is said in terms of speaker's attitude about what is said (i.e., the way it is said). This claim finds early support in Frege's (1879), *Begriffsschrift* where a distinction is made between the content of a sentence that expresses a judgment, and the act of judgment, that is, the assertion of the content. In the *Tractatus*, Wittgenstein also makes a distinction along the similar lines by stating that (Wittgenstein, 2001, Paragraph 4.022) "... A proposition shows how things stand if it is true. And it says that they do so stand". Although Wittgenstein later doubted that content can be sharply split from the purpose of a statement (Wittgenstein, 1953), Stenius (1967) elaborated on Wittgenstein's initial suggestion by distinguishing between a "sentence radical" – i.e., the content – from the "modal element" that amounts to the mood of the sentence. From the foundations set by Austin in his *How To Do Things With Words* (Austin, 1962), Searle developed a theory of speech acts (Searle, 1969) that seems to shape the current view of the separation between dictum and modus. For Searle, a meaningful utterance amounts to an attempt by the speaker to perform a speech act in order to convey something to the hearer. Elementary speech acts are of the form $F(P)$, where P is the content (the dictum) and F gives the illocutionary force, that is, the way in which P is communicated.

We turn to Searle's speech-act theory to better understand the communication process. Therein, communication is considered as action (Searle, 1969): a speaker makes an utterance in an attempt to change the state of the world. What distinguishes speech acts from non-speech actions is the domain of the speech act, that is, the part of the world that the speaker wishes to modify. With speech acts, the aim is to influence the mental state of the hearer. In the RE setting, the stakeholders and the engineers will be exchanging information, and will in turn take the roles of speakers and listeners. If we assume for simplicity the case of the speaker being a stakeholder, and the software engineer being the listener, upon being subjected to the speech act, *the engineer should distinguish its content from its illocutionary force in order to refer to the communicated content as a kind of requirement or otherwise*. For example, when someone says "The user should be able to book a plane ticket" (or if it is written in some documentation given to the software engineer), this may be expressing a desire that a user can book a plane ticket. By distinguishing the modus from the dictum, and following Searle's speech acts, we may conclude that the condition described in the content is desired. This would lead us to conclude that the content amounts to a requirement.

Depending on the illocutionary force, the engineer will classify the content of the communication as requirements or otherwise, and thereby differently refer to and act upon the communicated content: a core ontology for requirements should include concepts that cover all kinds of illocutionary force. Differences in illocutionary force lead Searle to distinguish assertive, directive, commissive, expressive, declarative, and representative declarative speech acts (Searle, 1975), as we recall below:

- *Assertive*. An assertive speech act conveys that the speaker believes in the truthfulness of the communicated content. It is unnecessary for the content to be actually true; what the assertive conveys is only that it is deemed true by the speaker. E.g., the speaker states "The door is closed" and believes that the door is closed.
- *Directive*. The content of the directive speech act describes conditions that the hearer desires to see become true. In contrast to the assertion, the speaker believes that the conditions do not hold and desires that they become so at some potentially undetermined time in the future. E.g., the speaker says "Paint the door in blue" and wants the door to be painted in blue. It is important to note that,

while a directive conveys a desire, it does not convey commitment to act in order to bring about the desired conditions. It is instead a commissive speech act that conveys intentions.

- *Commissive*. The commissive speech act indicates to the hearer that the speaker intends to perform actions described in the content; if the content describes conditions, then the commissive indicates that the speaker will perform actions needed to bring about states of affairs in which these conditions hold. E.g., the speaker says “I will paint the door” and intends to do it.
- *Expressive*. An expressive speech act conveys the speaker’s attitude, emotion, or feeling about a condition that may hold. An expressive thus amounts to an evaluation of the condition in terms of more or less intense favor or disfavor. Henceforth, we shall say that an expressive communicates *evaluations* to cover the evaluations that arise from attitudes, emotions, moods, or feelings. We shall, however, look into these concepts in more detail later on (Section 4.5). E.g., the speaker states “I like doors painted in blue more than those painted in red”.
- *Declarative*. A declarative speech act brings about the conditions specified by its content, provided that the role of the speaker allows the realization of the conditions. A president declaring war and a judge issuing a verdict both use declaratives, thereby bringing about the conditions conveyed by the speech act (i.e., the war is declared, the verdict applies). The content of a declarative speech act is believed to be true by the speaker.
- *Representative declarative*. A representative declarative speech act is used by the speaker to acknowledge that the conditions described by its content hold. As for the assertive and the declarative, the content of a representative declarative speech act is believed true by the speaker. E.g., a judge says “I find you guilty as charged”.

According to Searle’s speech acts, assertives, declaratives, and representative declaratives communicate beliefs of the speaker, directives convey desires, commissives intentions, and expressives evaluations. It is essential in RE to distinguish between the conditions that actually hold and those that are desired, but do not hold in the absence of the system-to-be. The very purpose of the system is to bring about the desired conditions. Consequently, the fundamental difference that any core ontology for requirements should make is that between beliefs and desires that are communicated. In order to do so, it is not necessary to finely separate assertives from declaratives, for they both convey beliefs. Instead, it is essential to distinguish assertives, declaratives, and representative declaratives from commissives: the former convey beliefs, while the latter convey desires. We will see below that, while Zave and Jackson’s ontology does distinguish between desired and believed conditions, and implicitly between these two and intentions, it abstracts entirely from the information conveyed by expressives. We will show that *evaluations*, which are conveyed by expressive speech acts, require a treatment distinct from that of beliefs, desires, and intentions. By giving evaluations their due place, we are brought to revise the currently accepted formulation of the requirements problem.

The assumption that communication plays a key role in information acquisition in RE led us to recall Searle’s taxonomy of speech acts, to distinguish dictum from modus, and to relate speech acts to notions of belief, desire, intention and evaluation. We can consequently observe that any core ontology for requirements, which admits the importance of communication for the RE process, should feature components that enable the classification of communicated content according to it being believed, desired, intended or giving an evaluation. Any proposal for a core ontology for requirements with a scope different than the one just highlighted should therefore show, not only that there is more (or less) in communication than the six fundamental kinds of speech acts, but that there is more (or less) to the fundamental kinds of mental attitudes and evaluations. This is a strong, yet very precise criterion that any proposal of a core ontology for requirements needs to meet in order to be taken as relevant.

2.3. Choosing a foundational ontology

Once we know what to cover with the core ontology, we are to produce definitions of its concepts. One commonly does so by mapping to others, hopefully more familiar and more precise in the reader's vocabulary. It is then hoped that the more these other concepts fit the reader's intuition, the more relevant in her view the definitions. To ensure an acceptable degree of rigor, we discuss the ontological nature of the components of CORE in relation to categories defined in a more restricted vocabulary, namely a *foundational* ontology, for which the underlying philosophical choices are clear and the ontological categories are well-delimited. This makes our assumptions clearer and easier to discuss and revisit in future efforts.

A foundational ontology is a theory about the abstract domain-independent categories in the real world. Its main purposes are to act as a starting point for building new ontologies, as a reference point for easy and rigorous comparisons among ontological approaches, and as a foundational framework for analyzing, harmonizing, and integrating existing ontologies. The choice of a foundational ontology in the present discussion depends essentially on how alternative foundational ontologies rank on two criteria. The first is the consistency between the ontological choices that underlie a candidate foundational ontology and the very general assumptions that we make about the RE process. In other words, the understanding of the overall RE setting and process ought to fit with the worldview reflected in the candidate foundational ontology. The second criterion is the relevance of the ontological categories with regards to the problem at hand. Given the scope of our RE ontology (Section 2.2), the question is whether there are ontological categories in the foundational ontology to which we can map the concepts of our RE ontology.

We make one straightforward assumption about the RE setting and process: Information communicated by the stakeholders over the course of RE reflects their perceptual bias, being influenced by individual professional, cultural, and other backgrounds, and social conventions. In this respect, concepts of the RE ontology ought to be grounded in ontological categories that underlie natural language and human commonsense. This leads to the first desideratum for the foundational ontology:

1. The foundational ontology should be descriptive (as opposed to revisionary – see further for details).

The second desideratum arises from the scope of CORE:

2. The foundational ontology of choice should have concepts that correspond to mental attitudes, that is, explicit components that cover the notions of belief, desire and intention.

The third desideratum arises from a longstanding issue in CORE, namely the separation between functional and nonfunctional requirements:

3. The foundational ontology should have concepts, in which we can ground the distinction between functional and nonfunctional requirements.

The common understanding is that the functional requirements describe what the system should do, whereas the nonfunctional ones describe how well (according some chosen criteria) the system should perform its functions. As we shall see in more detail later on, this common understanding gives a weak criterion for separating functional from nonfunctional requirements. The third desideratum above asks for a distinction between functional and nonfunctional requirements grounded in the foundational ontology.

Among the available foundational ontologies that we mention below, we chose the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) (Masolo et al., 2003), which addresses the given desiderata. DOLCE makes the following ontological choices:

- DOLCE is descriptive in that it aims to capture the ontological categories underlying natural language and human common sense. Categories in the ontology are therefore conceived as cognitive artifacts ultimately depending on human perception, cultural imprints, and social conventions. Descriptive contrasts to revisionary; choosing the latter commits to the aim of capturing the intrinsic nature of the world, that is, answering what exists, as opposed to what may be perceived to exist.
- DOLCE is multiplicative, allowing different entities to be co-localized in the same space-time. Broadly speaking, in a multiplicative ontology, we can say “the paper ticket is constituted of an amount of paper”, whereas in a reductionist ontology, we would say “the paper ticket is an amount of paper”. Consequently, if we speak of the departure date for some online booking system, we say in the former case “the paper ticket has a departure date” and not “the amount of paper has a departure date” as it would be more appropriate in the reductionist case.
- DOLCE takes a possibilist view: entities are allowed independently of their actual existence. At first sight, possibilism seems well adapted to RE for we are concerned with optative statements, which speak of what is presently not but is desired to be. DOLCE seems to adopt possibilism as a consequence of adopting modal logic for its formal characterization (Masolo et al., 2003). The actualism/possibilism discussion is considerably more elaborate, so that we may still properly deal with optative statements in (some variants of) actualism (for an introduction to the debate, see Menzel, 2007).
- DOLCE distinguishes between and allows both enduring and perduring entities assuming thus that entities have both temporal and spatial parts. The ontological choice at play here is that between endurantism and perdurantism, which has consequences on the questions of how identity and change are understood. DOLCE’s cognitive bias leads to allowing both endurants and perdurants, whereby they differ in that something is an endurant if and only if (i) it exists at more than one moment and (ii) statements about what parts it has must be made relative to some time or other. This is interesting in the present discussion mostly in that the cognitive bias is accounted for, so that statements about perdurants and endurants are equally accepted.
- DOLCE is an ontology of particulars. It takes the domain of discourse not to contain, e.g., the instantiation relation, so that no entity in the domain has instances. The consequence is that universals are not subject of being organized and characterized within, but are left outside DOLCE:

We take the ontology of universals as formally separated from that of particulars. Of course, universals do appear in an ontology of particulars, insofar they are used to organize and characterize them: simply, since they are not in the domain of discourse, they are not themselves subject to being organized and characterized. (Masolo et al., 2003, p. 13.)

The taxonomy of basic ontological categories in DOLCE is shown in Fig. 1. DOLCE distinguishes four basic categories:

- an *endurant*, whose proper parts are wholly present at any time;
- a *perdurant*, which accumulates parts over time (e.g., a process);
- a *quality*, which is a basic perceivable and measurable entity that inheres to other entities (e.g., the color of this desk);

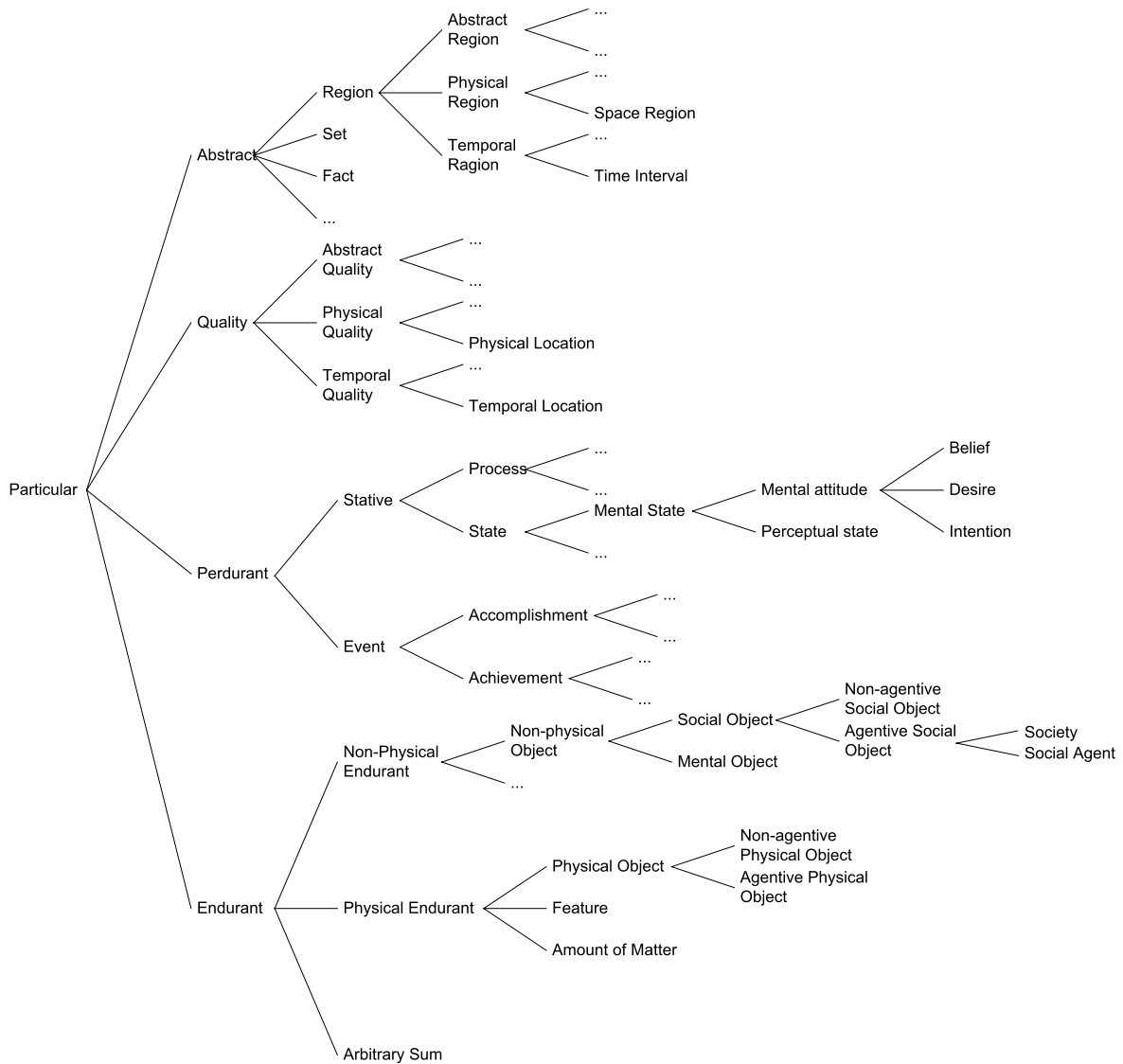


Fig. 1. Taxonomy of basic categories in DOLCE COM, borrowed from Masolo et al. (2003) and Ferrario & Oltramari (2004).

- an *abstract*, which has no spatial or temporal qualities, and is not a quality itself (e.g., fact, set, time interval, etc.).

DOLCE satisfies our first desideratum. Being a descriptive ontology, categories in DOLCE are defined to reflect concepts that depend on human perception, cultural imprints, and social conventions. Decisions taken over the course of RE are informed by stakeholders having perceptual bias, individual cultural and other backgrounds, and specific social conventions. Not recognizing such bias could hardly be an appropriate choice at present: the resulting RE ontology would be removed from the reality of the problem that we intend to resolve. As Masolo et al. (2003) point out, DOLCE has a clear cognitive bias, for its ontological categories are intended to capture ontological categories that underlie natural language

and human commonsense. This fits well with the approach adopted in defining CORE. We define the concepts of CORE by drawing from types of speech acts, which themselves are intended to reflect the communication of content expressed in natural language. Note that the RE effort involves the representation of information that is communicated by the stakeholders, and is therefore a process in which already formed conceptualizations are made explicit. Categories in DOLCE are defined to be descriptive notions that assist in rendering already formed conceptualizations explicit. By being descriptive, DOLCE is relevant to the RE process, as this process aims to capture what is perceived and communicated.

To satisfy our second desideratum, we adopt Ferrario and Oltramari's computational ontology of mind (COM) (Ferrario & Oltramari, 2004), which is an extension to DOLCE. COM defines mental attitudes within DOLCE, by considering them as kinds of Mental Attitude, which in turn is a kind of Mental State. Mental State therein is a kind of DOLCE State, which itself is a Perdurant.

It is DOLCE's Quality category that we use extensively below in CORE in order to address our third desideratum. In DOLCE, qualities are basic entities that one perceives and measures. A quality differs from "property" in that the former is a particular, while the latter is a universal. Every entity comes with some qualities. Also, qualities can inhere in other qualities. Qualities belong to a finite set of quality types, such as, e.g., size, color, etc. and are characteristic for – or inhere in – specific individuals. A quality (e.g., color) is distinguished from its "value" (e.g., some shade of red); the latter is called a "quale" and describes the position of an individual in a certain conceptual space, i.e., a "quality space". Two particulars having exactly the same color thus carry color qualities which have the same position in the color space, or, in other words, have the same color quale. We shall see later on that the distinction between functional and nonfunctional requirements is based in the Quality category.

When choosing a foundational ontology, alternative foundational ontologies are compared over their underlying ontological choices (Masolo et al., 2003). Among the dozen or so foundational ontologies that are available (for an overview, see Oberle, 2006), five are most relevant with regards to the desiderata identified above: the Basic Formal Ontology (BFO) (Spear, 2006), DOLCE (Masolo et al., 2003), OCHRE (Schneider, 2003), the Suggested Upper Merged Ontology (SUMO) (Niles & Pease, 2001) and Cyc (Lenat et al., 1990). While we briefly discuss these ontologies further, Cimiano et al. (2004) present and compare them in more detail. Each of the foundational ontologies shown in Table 1, has been defined with different purpose in mind and therefore subsumes different ontological choices.

RE aims to be consistent with human commonsense, so that the foundational ontology of choice should be descriptive. It should also be multiplicative, since reductionism seems further away from human commonsense (Oberle et al., 2007). RE involves to a considerable extent the reasoning about what may and will be the case in the future, so that the ontology ought to be possibilistic. Finally, we would hope for an ontology in which endurants are distinguished from perdurants, which again seems to fit intuition (Masolo et al., 2003). The closest with regards to the needs presented here are DOLCE and

Table 1
Comparison of main foundational ontologies and their ontological choices (from Cimiano et al., 2004)

	BFO	DOLCE	OCHRE	OpenCyc	SUMO
Descriptive	no	yes	no	yes	yes
Multiplicative	no	yes	unclear	unclear	yes
Possibilism	no	yes	yes	unclear	unclear
Perdurantism	yes	yes	no	unclear	yes

SUMO. DOLCE is preferred over SUMO. The latter aims to combine categories and relations coming from different top-level ontologies in order to improve interoperability, communication, and search in the Semantic Web field. Given that SUMO merges different upper level ontologies, it is not influenced by an overall theoretical approach, but adopts general categories from various sources. It is neither clearly multiplicative nor clearly reductionist, and the same dilemma applies to its position with regards to the choice of either possibilism or actualism. It is classified in Table 1 as descriptive, since it includes the commonsense distinction between objects and processes. Clear ontological choices adequate to the present discussion, explicit mental attitudes, and the convenient notion of Quality lead us to choose DOLCE and its extension COM.

3. Overview of the ontology

The purpose of this section is to provide a rough introduction to the concepts in CORE (Section 3.1) and relationships used in the definition of the requirements problem (Section 3.2). Detailed explanations of the concepts and relationships are relegated to subsequent sections (Section 4–5).

3.1. Overview of the concepts

The taxonomy of the concepts in CORE is given in Fig. 2. In introducing this paper, we observed that requirements obtained via communication with the stakeholders can take the form of variously detailed, potentially ambiguous and vague natural language statements given without much concern for a shared vocabulary about the system-to-be and its relevant surroundings. It is then clear that we will only obtain atomic speech acts if we impose constraints on the communication with the stakeholders. In the general

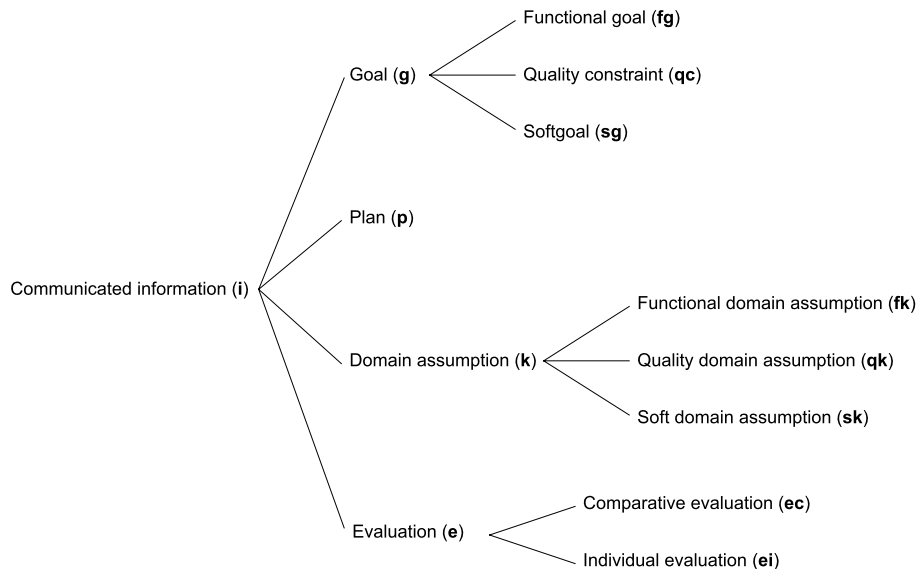


Fig. 2. Taxonomy of concepts in CORE. Letters in parentheses are abbreviations used throughout the paper (also see Appendix A).

case, such constraints cannot be assumed. Sentences may thus combine speech acts, as in the following example:

Marketing rules say that business seats are never less expensive than seats in economy, so that no business seat can be sold through the website at a price lower than the average price of the seat in economy. (Ex. 1)

The sentence above is intended to convey a belief and a desire, illustrating that potentially complex combinations of speech acts can be encountered. Any reference to (a part of a) communication, which may range from a simple written or otherwise recorded sentence to an elaborate discourse, falls in the root concept of CORE, called Communicated information. As a convention, we will denote some particular communicated information by $i\phi$, where ϕ symbolizes a sentence (or a discourse) and i indicates that ϕ is a particular, classified as Communicated information.² Since an $i\phi$ can involve conjunctive, disjunctive, conditional, or other combination of atomic speech acts, we need to decompose it and perform a finer classification of the parts obtained thereby. This finer classification, and the choice of the corresponding concepts in CORE is guided by the intent of distinguishing dictum on the basis of modus; that is, if the dictum of a speech act is believed, it is classified differently than if it is desired. We know from the earlier discussion (Section 2.2) that the content conveyed by a speech act will either be believed, desired, intended, or state an evaluation, and thus have four corresponding concepts in CORE: Domain assumption, Goal, Plan, and Evaluation. Given then some $i\phi$, in which we observe two atomic speech acts, say $i\phi_1$ and $i\phi_2$, we are interested in establishing, for each of the two, if it is believed, desired, intended, or conveys an evaluation. To classify any of ϕ_1 and ϕ_2 in one of the said four concepts, the following classification rules are applied:

- The content of an assertive or declarative or representative declarative speech act is believed by the speaker, and is classified as a Domain assumption.
- The content of a directive speech act is desired by the speaker, and is classified as a Goal.
- The content of a commissive speech act is intended by the speaker, and is classified as a Plan.
- The content of an expressive speech act states the speaker's evaluation, and is classified as an Evaluation.

The four concepts cited above cover the scope intended for CORE, for they allow us to classify the content of any type of speech act into categories, which are defined to match the mental attitude that the speech act associates to the content of communication. We thus classified believed content as Domain assumption, desired content as Goal, and so on. It may seem confusing to refer with the term Plan to the content of a commissive speech act, which conveys its speaker's intention. We know from Cohen & Levesque (1990) that an agent adopts some intention X if (i) it believes X is possible, (ii) it does not believe it will not bring about X, (iii) it believes it will bring about X, (iv) it does not intend all the side effects of bringing about X, and (v) it invests effort in trying to bring about X. Commitment thus plays a key role, but the plan may be missing; it may well be constructed along the way. While we do not contest that intentions, commitments, and plans are distinct, to capture commitment via the notion of Plan during early RE is to assume that the requirements engineer can elicit information about a potential plan from the stakeholder who expressed that commitment. This is important when searching for the solution to the requirements problem. If the stakeholder does have a plan for the realization of the

²Appendix A gives other abbreviations used throughout the paper.

commitment, then the requirements engineer can ask further questions so as to elicit the details of that plan, and perhaps advise on how that plan could be amended to include some interventions of the system-to-be. If the stakeholder happens to have no plan on how to realize the commitment, this signals to the requirements engineer the necessity to look into potential plans, which may be performed jointly by the system and the stakeholder in order to realize the stated commitment. We thus use of the term Plan to capture the content of commissives in order to highlight that a plan must somehow be identified, and that one of the very basic tasks in RE is indeed to identify plans, as we shall see further down in the definition of the requirements problem. The commissive thus signals that plans must be sought, not necessarily that they do exist. It is through RE that all the plans will need to be specified in order to determine whether they are part of a solution to the requirements problem.

It is interesting from a methodological standpoint to observe that a content ϕ of communication becomes classified in CORE by the very act of being communicated: ϕ starts as the dictum of a speech act, which immediately tells us the mental attitude ϕ is associated to, which in turn (using the simple classification rules above) puts ϕ into one of the four categories. As an example, consider the following atomic speech act:

No business seat has a lower price than an economy seat. (Ex. 2)

If the speaker/stakeholder believes the above is already true, the statement is an assertive speech act. If the stakeholder intends to institute the above as a rule, the statement is declarative speech act. If the stakeholder merely reiterates that the above applies, it is a representative declarative.

I hope/wish/desire/expect booking to be always confirmed with the new system. (Ex. 3)

I will ensure that booking is always confirmed. (Ex. 4)

It is preferred that the booking confirmation be sent quickly after booking. (Ex. 5)

Statement Ex. 3 is directive as it indicates what is desired. Ex. 4 is commissive for it points out the stakeholders' intention on bringing about something desired. Finally, Ex. 5 is expressive, as it makes explicit stakeholder's attitude on how the booking confirmation is to be sent to the user. While the illocutionary point is not apparent from written text (as in the examples above), it is not absent (Searle, 1980): when given documentation, the software engineer must determine if it expresses desires, beliefs, or otherwise.

Before outlining other concepts in CORE, consider the parallels with Zave and Jackson's ontology. Following their discussions (Zave & Jackson, 1997), we know that the critical distinction between their "requirement" and their "domain assumption" lies in that the former expresses something desired, while the latter something that is already the case, or is/will be the case regardless of the system. It is not difficult to see that Ex. 2 is thereby their "domain assumption", while Ex. 3 is their "requirement". Zave and Jackson's "specification" involves intentions: agents (including, e.g., stakeholders and the system-to-be) choose how to act and commit to do so, and this in order to bring about states of the world in which "requirements" are satisfied and "domain assumptions" are not violated. Hence, while desires give "requirements" and beliefs "domain assumptions", intentions give "specifications". Ex. 4 is part of a specification, in Zave and Jackson's terminology.

We see therefore that Zave and Jackson's domain assumption, requirement, and specification concepts cover mental attitudes (i.e., belief, desire, and intention) as they are usually conceptualized in philosophy (Oppy, 1998). More can be, and often is communicated than mental attitudes. Looking at the notion of attitude in psychology (Bizer et al., 2003) (which is some of what expressive speech acts can convey), attitude is identified most closely with affect, i.e., a general evaluative reaction (e.g., "I like Y" or "I like Y more than Z"). Ex. 5 is an expressive as it communicates an attitude. Zave and Jackson's core ontology is incomplete in that anything conveyed by expressives is missing. We show further on that this has significant consequences on the formulation of the requirements problem. CORE is more expressive than Zave and Jackson's already at the coarse-grained level, where we distinguish Goal, Plan, Domain assumption and Evaluation concepts.

The depth of CORE is delimited by the importance of distinguishing functional from nonfunctional requirements, and individual evaluations from comparative evaluations. These cannot be determined from the type of the speech act alone, but require the analysis of the content of the speech act. Depending on what ϕ states in a goal $\mathbf{g}\phi$, ϕ is classified as Functional goal, Quality constraint or Softgoal. Functional goals capture desired behaviors, while quality constraints refer to the quality criteria that these behaviors ought to satisfy. For example, a functional goal might say that the password be encrypted, while a quality constraint will specify the desired encryption level. The need for encryption might have come from the desire for sufficient user privacy. A desire for "sufficient privacy" is subjective, in the sense that "sufficient privacy" for one person may well be different than what someone else understands by that same expression. In the face of subjectivity, the best we can do is suggest an approximation of the desired subjective condition. Finer classification happens via an analogous rationale to domain assumptions, where we distinguish Functional domain assumption, Quality domain assumption and/or Soft domain assumption concepts.

Finer classification of evaluations takes place via Individual evaluation and Comparative evaluation concepts. Both may capture the content of attitudes, emotions, feelings, or moods. The difference lies in that an individual evaluation states the (un)desirability of a single particular, while a comparative evaluation compares in terms of (un)desirability two or more particulars. An example of the former is a stakeholder saying that she thinks it is good to be able to check-in to a flight via a web site. When she says that "It is better to receive a reminder of the flight via email a day or two before the flight date, than not to be reminded at all", she gives a comparative evaluation.

3.2. Overview of the relationships for the requirements problem

The relationships discussed further serve to relate instances of concepts in CORE in such a way as to highlight the salient qualities of the requirements problem. Relationships are not part of CORE, but remain outside of the ontology. There are four relationships: j-refine (Section 5.2), j-approximate (Section 5.1), e-compare and evaluate (Section 5.3).

To illustrate j-refine and e-compare, suppose that the following comparison is stated:

ec₁: It is better to receive a reminder of the flight via email
a day or two before the flight date, than not to be re- (Ex. 6)
minded at all.

We see that the above is (i) a comparative evaluation and (ii) compares two desires, and thus goals, which we denote $\mathbf{g}\phi_1$ and $\mathbf{g}\phi_2$, and abbreviate and rewrite as follows:

\mathbf{g}_1 : Send a flight reminder via email before the flight date. (Ex. 7)

\mathbf{g}_2 : Send no flight reminders. (Ex. 8)

The e-compare relationship stands between the two goals above, and is obtained from the statement in \mathbf{ec}_1 . j-refine will thus allow us to distinguish in a comparison, those which are being compared, from their relative desirability.

To understand the need for j-refine, observe that we rewrote the statement in \mathbf{ec}_1 via the two goals and the comparison between them. To do so, we must have performed some inference, in which \mathbf{ec}_1 (and potentially some other knowledge) acted as the premise(s), while the two goals and the comparison act as the conclusion. j-refine is designed to capture the relationship between premises and conclusions.

evaluate is a relationship intended to capture the individual evaluation of something (i.e., any DOLCE Particular). Given an individual evaluation, e.g., “It is good to be able to check-in to a flight via a web site”, we are interested in separating what is evaluated from the desirability qualifying it: we can obtain from the example that there is a goal “Traveler can check-in to a flight via a web site” and there is an evaluation by a stakeholder, which roughly says that this goal “is good”. Roughly speaking (details are in Section 5.3), evaluate is a unary relationship, which indicates whether what is evaluated is compulsory or optional. In the example, if further discussion with the stakeholder reveals that the system-to-be *must* satisfy the said goal, then evaluate will lead us to call that goal compulsory. If the user reveals that she would hope that the system can satisfy that goal, but that she would still use a system that does not satisfy it, then we will consider that goal as optional.

To see the purpose of j-approximate, recall the role of subjectivity observed earlier in relation to the Softgoal concept (Section 3.1). We noted that the desire for encryption of a stakeholder may come from the desire for sufficient user privacy, and that asking for “sufficient privacy” is subjective, in the sense that the evaluation that privacy protection is “sufficient”, is subjective. In the face of subjectivity, the best we can do is suggest an approximation of the desired subjective condition. Hence the j-approximate relationship. It relates quality constraints to softgoals: if the system satisfies the quality constraints that stand in the j-approximate relationship to the softgoals, then that is considered evidence enough to say that the softgoals are also satisfied to an acceptable level. In other words, this relationship says that a softgoal can for all practical purposes be approximated by one or more quality constraints. The term j-approximate abbreviates “justified approximation”, in which we say “justified” because subjectivity guarantees the absence of proof for the approximation between the quality constraints and the softgoals. j-approximate can stand between quality domain assumptions and soft domain assumptions.

4. Concepts

4.1. Communicated information

From a purely practical standpoint, Communicated information is a catchall concept, the instances of which should capture the content (dictum) communicated between the participants in RE. We do not impose constraints on the structure of that content: an instance may capture the content of an atomic speech

act (e.g., Exs 2–5 in Section 3.1), or the collection of content communicated via a potentially complex combination of speech acts of one or more types in Searle’s taxonomy (e.g., Ex. 6 in Section 3.2). Identifying and recording the instances of Communicated information thus seems rather straightforward: hear, restructure if needed, and record the sentences that the speaker is saying, whereby anything from a simple sentence to an elaborate discourse can be an instance of Communicated information. This, however, tells us nothing about the ontological nature of the instances of Communicated information, which is necessary to provide the definition of the Communicated information concept. We move step by step towards this definition below.

4.1.1. *Dependence of communicated information on mental states*

What is the “content” of communication, and what does it mean that an instance of Communicated information “captures” that “content”? To answer this question, we need to go back to Mental States in DOLCE COM. CORE takes that the participants in RE act as intentional agents, in the sense given to this term within the belief-desire-intention paradigm in artificial intelligence (Rao & Georgeff, 1991): the intentional agent is assumed to have beliefs about the conditions holding in her environment, desires about the conditions that she observes as unsatisfied and wishes to see satisfied in her environment, and intentions, which guide her means-ends reasoning about how to satisfy the desires to which she commits. She is also assumed capable of evaluating, in terms of desirability, these conditions individually or by comparing some to others. This places CORE in line with the understanding that Ferrario & Oltramari (2004) have of beliefs, desires, and intentions in the COM component of DOLCE. As illustrated in Fig. 1, Belief, Desire and Intention are Mental Attitudes in DOLCE COM, whereby a Mental Attitude is a subtype of Mental State. Mental State is a subtype of DOLCE State.

It is then assumed in DOLCE COM that an intentional agent has, at a certain time, a Mental State about a Mental Object. We agree with that assumption, and we see Mental Objects as being “nothing but representations of something else, both in the case they pertain to perceptual states and mental attitudes” (Ferrario & Oltramari, 2004, p. 5). Mental Object is in DOLCE a subtype of Non-Physical Object: “Non-physical Objects are divided into Social Objects and Mental Objects according to whether or not they are generically dependent a community of agents. A private experience, for instance, is an example of a mental object” (Masolo et al., 2003, p. 24). An intentional agent is thus assumed to have, at a certain time, a *Mental State about a Mental Object, whereby a Mental Object refers to other DOLCE Particulars*.

Just as Mental States are “about” other DOLCE Particulars, so are Communicated information in CORE “about” DOLCE Particulars. In the loose terminology used earlier, the “content” of communication that is “captured” by an instance of Communicated information is what this instance is “about”. Now, Communicated information are not Mental States, because of the following. If an intentional agent believes X , where X is some DOLCE Particular, then, so to speak, X which is the “content” of that belief, i.e., what this belief is about, will be what a Domain assumption from CORE will be about. If we establish through communication that the agent believes X , then we will classify X as a Domain assumption in CORE. This is in line with our earlier discussion (Section 3.1), where we noted that the content of a belief, desire, or intention, becomes the content of, respectively, a Domain assumption, Goal, or Plan. It is via communication that this content becomes amenable to classification “outside” the intentional agent, who believes, desires, or intends what the content refers to. CORE allows the classification of that content “outside” the intentional agent in a way that keeps the separation between beliefs, desires, and intentions, while communication is a means to obtain that content from the intentional agent.

Consider an example. A stakeholder desires that she can book flight tickets online. In the terminology above, she is an intentional agent, who at a certain time, has a Mental State, more precisely a Desire. The content of that Desire, i.e., what the Desire is about, can be understood as her being able to perform

a DOLCE Process, which amounts to book a flight ticket online.³ The software engineer can find out that this stakeholder has this Desire via communication with her over the course of RE. Since we recalled earlier (Section 2.2) the distinction between dictum and modus, we see that the engineer will acquire two kinds of information via communication: the dictum, which is the content of the stakeholder's Desire, and the modus, which will tell the engineer that the stakeholder in fact desires (as opposed to believes or intends) whatever DOLCE Particulars are referred to by that content. The next step for the engineer is to record these two information: the dictum becomes the content of an instance of a concept in CORE, and the modus determines the choice of the concept in CORE that this content will be an instance of. We thus started from the Desire of the stakeholder, and ended up with a Goal, whereby both have the same content, that is, they are both about the same DOLCE Particulars.

The observations above lead us to the following preliminary and incomplete identity criteria for Communicated information:

1. An instance x of Communicated information refers to a particular Mental Object m .
2. x cannot exist without there being a particular Mental State s , which is about m .
3. x cannot exist without there being a particular DOLCE Process p , through which it is communicated that s is about m .

4.1.2. Communicated information are Non-Agentive Social Objects

For simplicity here, we will speak of Communicated information both as a category and a concept. Particulars in Communicated information cannot be classified as DOLCE Abstract, because any communicated information is "in time", that is, has a creation time, before which they do not exist, and after which they do exist. Clearly, communicated information is not a DOLCE Quality, which leaves Endurants and Perdurants. We noted earlier (Section 2.3) that Endurants differ from Perdurants, in that the proper parts of the former are wholly present at any time, while those of the latter are accumulated over time. Particulars classified under Communicated information are consequently Endurants. To obtain a finer classification, observe that within the context of an RE project, Communicated information are not unlike laws (as in legislation) in several important respects: (i) both are non-physical, (ii) their identity depends on/is recognized within, a community of (intentional) agents and (iii) they are non-agentive, in that we cannot ascribe intentions, beliefs, and desires to them. E.g., a goal (i.e., instance of Goal in CORE) that the system-to-be should satisfy (e.g., enable users to book flight tickets online) will only be acceptable if the participants in RE agreed that this goal should be satisfied by the system-to-be. We are thus led to the following identity criterion that adds to the ones identified above (Section 4.1.1):

4. Instances of Communicated information are classified as DOLCE Non-Agentive Social Object (see Fig. 1 for the supercategories of the latter).

4.1.3. Communicated information are artifacts

Having Communicated information as a subcategory of DOLCE Non-Agentive Social Object does not indicate whether the creation of some Particular in Communicated information is an intentional act of some agent. The classification of some Particular as communicated information is intentional, whereby the intention is to determine the characteristics that the system-to-be and its environment should have, or in other words, to identify criteria against which the system-to-be and its environment are evaluated. It follows that communicated information are artifacts, in the sense that they are created via the intentional attribution of "capacities" to Particulars. According to Borgo and Vieu:

³We could go further in decomposing and classifying this content, by asking what "flight ticket" and "online" are, among others, but we shall stop at this level of precision as this does not affect our subsequent observations.

The capacity of an entity is an individual, just as its color is. This quality maps into a quale that is a region (possibly a sum of atomic qualia) in the capacity space, which can be seen as some sort of functional conceptual space. The quale corresponding to the capacity of an entity at a given time collects all the various dispositions or behaviors the entity is able to express at that time. [...] Although capacity and attributed capacity map into the same space of qualia, the former is a physical quality whereas the latter is an intentional quality as it depends on the intentions of the creator at the time of the creation event. Capacity and attributed capacity also differ in the following: the quale associated with the attributed capacity does not change with time since it is fixed by the creation event; moreover, this quale is a set of regions of the capacity space, since the intended behavior of the artifact needs not be specified in a precise way, and may present vagueness. (Borgo & Vieu, 2009, p. 22.)

Suppose that a stakeholder tells us that she would like to receive a booking confirmation via email upon the completion of the booking process. She is expressing a Desire, the content of which is a Mental Object, about the following DOLCE State: “having received by email a booking confirmation about the completed booking”. To record the content of this Desire, we will classify the reference to the Mental Object in question as a CORE goal. By doing so, we have created a particular (i.e., an instance of) Goal, which is a Non-Agentive Social Object with the attributed capacity of acting as a criterion against which the system-to-be and its environment are evaluated. Note that we are thus classifying in CORE with the intention that the system-to-be gets built so that it can satisfy the original desire.

Communicated information bears the traits of Borgo and Vieu’s Artifact and Non-Agentive Social Object in DOLCE. Borgo and Vieu do discuss physical artifacts, and consider these as a subcategory of DOLCE Physical Endurant and sibling to Amount of Matter, Physical Object, and Feature. They do not, unfortunately discuss non-physical artifacts, although they do acknowledge their relevance. Without extending DOLCE to non-physical artifacts, we can suggest the following identity criterion for Communicated information:

5. Instances of Communicated information are created from a Non-Agentive Social Object by an intentional act of an intentional agent, by which the agent attributed to the Non-Agentive Social Object the capacity of being a criterion against which the system-to-be and/or its operational environment are evaluated.

4.1.4. Definition of communicated information

Based on the five identity criteria discussed above (Sections 4.1.1–4.1.3), we offer the following definition of Communicated information.

Definition 1 (Communicated information). Any instance x of Communicated information is a DOLCE Artifact, such that:

- (1) an intentional agent a has Mental States about Mental Objects m_1, m_2, \dots, m_n ;
- (2) a Non-Agentive Social Object o refers to m_1, m_2, \dots, m_n ;
- (3) a creates x from a Non-Agentive Social Object o by performing speech acts;
- (4) the effect of speech acts is the attribution of the Quality q to o ;
- (5) q is the quality (the capacity, in Borgo and Vieu’s terms) of being a criterion against which the system-to-be and its environment are evaluated.

The following remarks are in order:

- Definition 1 indicates that for x to be an instance of Communicated information, a Non-Agentive Social Object o must be attributed the Quality q , where q is the quality of being a criterion against which the system-to-be and its environment are evaluated. x is thus created from o by the intentional act, which in this case amounts to speech acts, that attributes a capacity to o . Furthermore, o simply refers to Mental Objects, about which are Mental States of the intentional agent (i.e., DOLCE Agentive Physical Object), who performs the intentional act. In more intuitive terms, x : (i) is created via speech acts performed by an intentional agent; (ii) refers to some Mental Objects, about which are this agent's Mental States; and (iii) is relevant for RE in that it is communicated with the intention of being used as a criterion for the evaluation of the system-to-be and its environment.
- What is believed, desired, and intended in relation to the system-to-be, and can thereby be referred to via instances of Communicated information, can act as criteria for the evaluation of the system-to-be and its environment. A system is built according to assumptions (i.e., beliefs) about the operating environment, so that checking whether it violates or complies to such assumptions gives signals that affect its evaluation: e.g., a system-to-be violates some assumption, it may not be an adequate choice, and changing its behavior may be relevant. Same applies to desires and intentions: a potential system-to-be can be evaluated in terms of which of the desires it satisfies, and which of the intentions it conflicts or complies with. It is in this sense that communicating beliefs, desires, and intentions about the system-to-be and its environment assigns to the reference to the content of those mental states (i.e., the Mental Objects these states are about) the capacity of being criteria against which the system-to-be and its environment are evaluated.
- Definitions of the subcategories of Communicated information are variations on Definition 1. They reflect the finer classification of instances of Communicated information, and this depending on (i) the type of speech act that is used in communication by the intentional agent a , and (ii) the Qualities and quales of Mental Objects m_1, m_2, \dots, m_n .

4.1.5. Towards a mathematically formal account of communicated information

We draw freely below on axioms and definitions in DOLCE, and the components that extend DOLCE to (physical) Artifacts (Borgo & Vieu, 2009) and Mental States (Ferrario & Oltramari, 2004). To say that x is an instance of Communicated information, we use the predicate \mathbf{i} and define it as follows.

$$\begin{aligned} \mathbf{i}(x) =_{\text{def}} & \exists a, s, m, t AB(a, s, m, t) \wedge \exists o Refer(o, m) \\ & \wedge \exists e, q \text{IntentionalSelS}(e, a, x, o, q) \\ & \wedge \exists p \text{SACT}(p) \wedge \text{DHD}(e, p). \end{aligned} \quad (\text{Di})$$

Definition (Di) reads: x is Communicated information if and only if (i) the mental state s of the intentional agent a is about the mental object m at time t , (ii) non-agentive social object o refers to m , (iii) e is the event of the agent a intentionally selecting the non-agentive social object o and attributing to it the quality q , (iv) p is a speech act and (v) the event e is directly historically dependent on the speech act p . The predicates given in Definition 1 are discussed in turn below, along with the associated definitions and axioms.

$AB(a, s, m, t)$ is Ferrario and Oltramari's formalization of their "aboutness" relation between Mental States and Mental Objects, and reads "Mental State s of the intentional agent a is about a mental object m at time t ". Definition (D1) indicates that $AB(a, s, m, t)$ verifies if and only if the Agentive Physical Object (APO) a participates to the state s at t and the Mental Object (MOB) m also participates to s at t , while m

is one-sided specific constant dependent on a . $PC(a, s, t)$ reads “endurant a participates in perdurant s at time t ” (Masolo et al., 2003).

$$AB(a, s, m, t) =_{def} APO(a) \wedge ST(s) \wedge MOB(m) \wedge OSD(m, a) \wedge PC(a, s, t) \wedge PC(m, s, t). \quad (D1)$$

The reference relation, which is the intuitive relation that stands between, e.g., the photograph of a person and that photographed person, is not formalized in DOLCE. We do, however, use the predicate *Refer*, and read $Refer(q, r)$ as “ q refers to r ”.

The predicate *IntentionalSelS* is strongly based on Borgo and Vieu’s *IntentionalSel* predicate. The difference between them is that the artifact is created from a Non-Agentive Social Object in the former, while it is created from an Amount of Matter or a Non-Agentive Physical Object in the latter. This is also the only difference in the axioms that apply to *IntentionalSelS*, compared to *IntentionalSel*. The Social Artifact x , written $SART(x)$ is created by the intentional association of a Non-Agentive Social Object o ($NASO(o)$) and a Quality q , where q is of the type *AttributedCap*, a primitive predicate denoting attributed capacities. The intentional association is an Event of type *CreationEv*. Written *IntentionalSel*, intentional selection is a primitive relation taking as arguments an Event (*EV*) e , an intentional agent a (i.e., Agentive Physical Object, $APO(a)$), a Social Artifact x , a Non-Agentive Social Object o , and a Quality q . *IntentionalSel*(e, a, x, o, q) reads “ e is the event of the agent a obtaining the artifact x by intentionally selecting o and attributing to it the capacity q ”. Axiom (A1) indicates that a Social Artifact is created by intentionally selecting a non-agentive social object; Axiom (A2) restricts *IntentionalSel*.

$$SART(x) \leftrightarrow \exists e, a, o, q \text{ IntentionalSel}(e, a, x, o, q). \quad (A1)$$

$$\begin{aligned} \text{IntentionalSel}(e, a, x, o, q) \rightarrow & EV(e) \wedge APO(a) \wedge SART(x) \wedge NASO(o) \wedge \text{AttributedCap}(q) \\ & \wedge qt(q, \phi) \wedge \exists t (ql_T(t, e) \wedge PC(o, e, t) \wedge PC(x, e, t) \\ & \wedge a, e, t \wedge K(o, x, t)) \end{aligned} \quad (A2)$$

Predicates called in Axiom (A2) are defined in DOLCE (Masolo et al., 2003); we simply recall their reading here: $qt(q, \phi)$: q is a quality of ϕ ; $ql_T(t, e)$: t is the temporal quale of e (i.e., e occurs at t); and $K(o, x, t)$: o constitutes x at time t . Axioms and definitions related to *IntentionalSelS* are given in Appendix B; they are all straightforward rewritings of Borgo and Vieu’s axioms and definitions, with the difference that the axioms specific to artifacts created from Amount of Matter or a Non-Agentive Physical Object are absent.

Two predicates remain in Definition (Di): *SACT* and *DHD*. $SACT(p)$ indicates that p is a speech act. Speech acts have not been formalized in DOLCE. Following our earlier discussion of speech acts, it is good enough for the present discussion to assume that Speech Act (*SACT*) is a subcategory of DOLCE Process, and that there is one subcategory of Speech Act for each speech act type in Searle’s taxonomy (Section 2.2), which leads to the following two axioms; the predicate *PRO* indicates that its argument is a DOLCE Process.

$$\forall p, SACT(p) \rightarrow PRO(p) \wedge \exists a, (APO(a) \wedge GED(p, a)). \quad (A3)$$

$$SACT(p) =_{def} SAAS(p) \vee SADI(p) \vee SACO(p) \vee SAEX(p) \vee SADE(p) \vee SARD(p). \quad (A4)$$

Axiom (A3) restricts *SACT* to processes that are generally dependent on Agentive Physical Objects. Axiom (A4) indicates the subcategorization of speech acts along Searle’s taxonomy: assertive (*SAAS*), directive (*SADI*), commissive (*SACO*), expressive (*SAEX*), declarative (*SADDE*), and representative declarative (*SARD*). General Dependence (D2) and Direct Historical Dependence (D4) are defined as follows (Ferrario & Oltramari, 2004).

$$GED(x, y) =_{def} \Box(\exists t (PRE(x, t)) \rightarrow \exists t'(PRE(y, t'))). \quad (D2)$$

$$HD(x, y) =_{def} \Box(\exists t (PRE(x, t)) \rightarrow \forall t (PRE(x, t) \rightarrow \exists t' (t' < t \wedge PRE(y, t')))). \quad (D3)$$

$$DHD(x, y) =_{def} HD(x, y) \wedge \neg \exists z (HD(x, z) \wedge HD(z, y)). \quad (D4)$$

We see from the discussion above that the mathematically formal characterization of $i(x)$ in Definition (Di) is rather imprecise, in the sense that many options are left open. More can certainly be said of the axioms on speech acts, their classification in DOLCE, restrictions on the relationship between speech acts and intentional selection. This would, however, require the extension of DOLCE to speech acts and non-physical artifacts. Both remain open issues at this time.

4.2. Goal, functional goal, quality constraint and softgoal

Definition 2 (Goal). Any instance of Communicated information that is communicated via a directive speech act is an instance of Goal.

Because the Goal concept captures desired conditions, it seems to correspond to Zave and Jackson’s “requirement” concept. This correspondence is only superficial, as Goal has three subcategories in CORE: Functional goal, Quality constraint, and Softgoal. The consequence is that Goal is a more expressive concept than Zave and Jackson’s “requirement” concept.

The Functional goal and Quality constraint concepts are introduced in CORE to cover the classical taxonomic dimension for the “requirement” concept: namely, the distinction between functional and nonfunctional requirements.⁴ It is widely accepted in RE that functional requirements refer to what the system does, as opposed to how well the system does what it does. The latter is covered by nonfunctional requirements. Ex. 9 thus gives a functional requirement.

Once the payment for the flight is confirmed, book the flight. (Ex. 9)

It should be possible to fully book a flight through less than 5 different screens. (Ex. 10)

Ex. 10 indicates how well booking performs. It identifies a measure on the behavior of the system (i.e., the number of different screens the user needs to go through to book a flight). In other words, Ex. 10 characterizes flight booking: it points to the existence of qualities for which only some (sets of) possible values are desired. It follows that the functional vs. nonfunctional distinction is grounded in the notion of DOLCE Quality (Section 2.3). Namely, a goal that refers to qualities and constrains quality

⁴Zave and Jackson do not discuss this taxonomic dimension for their requirement concept. However, the distinction between functional and nonfunctional requirements is a widely accepted one in the RE field, as Van Lamsweerde observes in his overview of RE research (van Lamsweerde, 2001).

values is a quality constraint; otherwise the goal is a functional goal. This still remains within accepted intuitions in RE: any functional requirement will describe what the system does, whereas a nonfunctional requirement will refer to qualifications on the system's doings, thus allowing us to characterize how well the system behaves. Although uncontroversial, this separation becomes more precise herein, leading to the following definitions.

Definition 3 (Quality constraint). Any instance x of Goal is an instance of Quality constraint if and only if:

- (1) x refers to a Quality q' , and
- (2) the quality type of q' has an associated quality space with a structure that is shared among the participants in RE.

Definition 3 refers to the structure of quality spaces. Masolo et al. (2003) indicate that in DOLCE:

Each quality type has an associated quality space with a specific structure. For example, lengths are usually associated to a metric linear space, and colors to a topological 2D space. The structure of these spaces reflects our perceptual and cognitive bias [...]. (Masolo et al., 2003, p. 17.)

When we speak of the metric linear space, that quality space is generally recognized and thus can be assumed as shared. Definition 3 requires that the structure of the quality space be shared only among the participants in RE. This in practice means that they agree on the understanding of the qualities they recognize in the system-to-be and its environment. When a goal does not refer to Qualities, it is called a functional goal.

Definition 4 (Functional goal). Any instance x of Goal is an instance of Functional goal if and only if:

- (1) x refers to a Perdurant (i.e., either Event, State or Process), and
- (2) x does not refer to a Quality.

It should be possible to verify whether a system satisfies goals before delivering it to the stakeholders. Any functional goal and any quality constraint are intended to be verifiable, in the sense that the software engineer can check at any time whether the system satisfies the chosen functional goal or quality constraint. Functional goals are verifiable, because the software engineer can determine whether what is functionally required is indeed delivered by the system: whether the system-to-be is/will be able to make an Event occur, bring about a State, or perform a Process. Whether the flight is booked whenever the payment is confirmed is not a matter of cognitive bias or ill-defined verification criteria – the flight is either booked or not. The second condition in Definition 3 ensures the verifiability for quality constraints: the quality space must be shared among the stakeholders. Number of screens involved in flight booking is a quality for which there is no controversy on the structure of the quality space or on the association of the system behaviors to the values in the quality space.

Nonfunctional requirements are taken to include abstract considerations such as security, safety, maintainability, convenience, and so on (Boehm et al., 1976; Mylopoulos et al., 1992). The question then is: How do these notions relate to our quality constraint concept? In other words, does desiring, e.g., “high convenience in flight booking” give us a quality constraint or something else?

Given that a Quality (Section 2.3) is a perceivable and measurable entity that inheres in other entities, there must be a quality space and values for, e.g., “convenience” in order to call it a Quality, and thereby state that asking for, e.g., “high convenience” is an instance of Quality constraint. We know that people

have the ability to qualify convenience since we observe that they can convey more or less precisely to what extent they find a system convenient. We also know that such qualifications are not necessarily (or rarely are) shared: while one person gives a strong favorable qualification, another one may disagree. It is reasonable to argue that any qualification requires that a somehow structured quality space exists. We could consequently conclude that high convenience is a quality constraint. To do so is, however, a mistake for the following reasons:

- Divergent evaluations of the same phenomenon mean that the structure of the underlying quality space is dependent on each individual's cognitive bias. The structure of the quality space is subjective.
- While each individual may have some kind of quality space for convenience (or for security, safety, accessibility, and so on), it seems that people are rarely (if ever) capable of defining and/or conveying the structure of that quality space in a precise manner. Verifiability is elusive.
- Subjective structure of the quality space goes against the need for a quality space with a structure shared among the stakeholders, which would subsequently facilitate verification.

We conclude thus that while quality constraints cover some of nonfunctional requirements, other non-functional requirements on, e.g., convenience, security, maintainability, and so on, are *not* quality constraints. We call them *softgoals* instead.

Definition 5 (Softgoal). Any instance x of Goal is an instance of Softgoal if and only if:

- (1) x refers to a Quality q' , and
- (2) the quality type of q' has an associated quality space with a structure that is not shared among the participants in RE.

The following statement, communicated by way of a directive speech act gives us a softgoal.

Flight booking should be convenient. (Ex. 11)

Following Definitions 2–5, observe that any instance of Communicated information obtained via a directive speech act will be classified further depending on whether it refers to Qualities: if yes, then it is necessary to determine if its quality space is shared among the participants in RE. If it does not refer to Qualities, it is an instance of Functional goal.

4.3. Domain assumption, and functional, quality and soft domain assumptions

Beliefs communicated by way of assertive, declarative, or representative declarative speech acts constrain the possible states of the world only to those in which the beliefs are not violated. Such beliefs correspond to domain assumptions, which should not be violated by the system-to-be or its relevant surroundings. Domain assumptions can refer to Qualities, in the same sense as quality constraints and softgoals are. We therefore conceive domain assumptions by analogy to goals. CORE contains domain assumptions, functional domain assumptions, quality domain assumptions, and soft domain assumptions by analogy to, respectively, goals, functional goals, quality constraints, and softgoals.

Definition 6 (Domain assumption). Any instance of Communicated information that is communicated via an assertive, declarative, or representative declarative speech act is an instance of Domain assumption.

Definition 7 (Functional domain assumption). Any instance x of Domain assumption is an instance of Functional domain assumption if and only if:

- (1) x refers to a Perdurant (i.e., either Event, State or Process), and
- (2) x does not refer to a Quality.

Below is an example of a functional domain assumption. It arises from the introduction of the Electronic System for Travel Authorization (ESTA) by the U.S. Department of Homeland Security (Customs and Border Protection, Department of Homeland Security, 2008). The ESTA is one of the measures introduced in 2008 within the broader efforts intended to reduce the terrorist threats to the USA.

Following the implementation of the mandatory Electronic System for Travel Authorization (ESTA), citizens or nationals of countries that participate in the Visa Waiver Program (VWP), and traveling to the USA must go to <https://esta.cbp.dhs.gov/>, follow the instructions to answer all the required questions, and submit an application for a travel authorization. (Ex. 12)

The above is relevant for a flight booking system, in the sense that any traveler that fits the above description should be informed about this requirement, and given the link to the ESTA website before and after booking.

Definition 8 (Quality domain assumption). Any instance x of Domain assumption is an instance of Quality domain assumption if and only if:

- (1) x refers to a Quality q' , and
- (2) the quality type of q' has an associated quality space with a structure that is shared among the participants in RE.

The proper processing of the request for travel authorization requires that it be submitted within a given time frame, as the following illustrates.

To facilitate the authorization process, DHS recommends that ESTA applications be submitted no less than 72 hours prior to travel. However, applications may be submitted at any time prior to traveling to the United States under the VWP. (Ex. 13)

The excerpt above makes reference to a Quality, that we could call “ESTA request submission time”, and that would capture the length of the time period from the submission time to the departure time. The quality space has a precise structure, given that “time period” has a standard and established meaning. The constraints on the values of the said quality are precise. If these were not the case, we would be facing a soft domain assumption.

Definition 9 (Soft domain assumption). Any instance x of Domain assumption is an instance of Soft domain assumption if and only if:

- (1) x refers to a Quality q' , and
- (2) the quality type of q' has an associated quality space with a structure that is not shared among the participants in RE.

Following the 11 September 2001 terrorist attacks, the United States' Department of Homeland Security requires electronic access to Passenger Name Record (PNR) for flights to, from, or through the USA. An International Agreement between the European Union and the USA establishes a legal framework allowing airlines to transfer passengers' PNR (Anonymous, 2006). The EU also provides guidance on how passengers should be informed that their data is made available to the relevant authorities (EC Article 29 Data Protection Working Party, 2007). The following excerpt is a guideline for the display of information on PNR access, that is shown to the user booking a flight via a website.

On a website, the longer notice should have the same level of visibility and accessibility for passengers as general fare and travel conditions. (Ex. 14)

The above indicates that the information on the availability of PNR data should be displayed to the website user in a "visible" and "accessible" way. The latter two are Qualities, with quality spaces that are not shared.

4.4. Plan

Stakeholders commit to act in ways that allow them to achieve goals within the setting established by domain assumptions. We interpret the intentions, communicated via commissive speech acts, as stakeholders' commitments to act. We introduce the concept of Plan, and use it to capture the intentions that stakeholders communicate. However, the stakeholders' intentions alone are insufficient to achieve goals: the intervention of the system-to-be is also necessary. Not only can the instances of Plan be stakeholders' intentions, but also the behaviors of the system-to-be. In other words, an instance of Plan can be an intention of the stakeholder, or an "intention" of the system-to-be (i.e., built-in ability to perform a process under specific conditions).

Definition 10 (Plan). Any instance x of Communicated information is an instance of Plan if and only if:

- (1) either x is communicated by a commissive speech act, or
- (2) x refers to a Process that should be performed by the system-to-be.

In RE we are interested in plans that satisfy generic goals, such as "book a flight", "schedule a meeting" or "fulfill a book request". Satisfaction of a goal at system runtime equates to the execution of appropriate plans for particular cases: booking a user-specified flight, scheduling a particular meeting, or responding to a specific book request. Consequently, plans are generic in the same sense that goals can be generic. The instances of Goal, Functional goal, Quality constraint, Softgoal and Plan will in practice accept parameters, that is, they are *generic* instances. For example, in "Book a flight" goal may involve departure and arrival dates and locations; when scheduling a meeting, the parameters would include the initiator, the time of the request, the list of participants, as well as the purpose of the meeting; for book requests, parameters would refer the author and title of the book, also the requester and the time of the request. Plans that satisfy these goals will be correspondingly parameterized. Making explicit the parameters is very relevant from a methodological standpoint: it has been recognized over the years (e.g., Dardenne et al., 1993; Jiang et al., 2006) that such information proves useful in the construction of data models for the system-to-be.

Note that our ontology places constraints neither on the way that parameterization is specified in instances of the concerned CORE concepts, nor on how instances of Plan are specified. It is not necessary

in practice for an instance of Plan to describe the transformations (e.g., as some sequences of steps) that need to be executed: we encounter situations in which we know how precisely one of the parties ought to act, while the best we can do for other parties is to restrict potential ways in which they can act – e.g., we often treat third party web services as black boxes of functionality, so that our Plan instance might be written as constraints on the inputs and outputs of the service. Both the specification of parameters and the specification of instances of concepts in CORE are matters that require a separate discussion, in a different setting. We leave them for a future treatment of an RE specification language intended to accommodate CORE.

4.5. Evaluation, and individual and comparative evaluation

Expressive speech acts convey the speaker’s attitude, emotion, or feeling. Overall, we call “evaluation” the content of expressive speech acts; this choice of terminology arises from the conceptualization of attitudes, and their relationship to emotions, feelings, and moods in psychology. In order to understand the Evaluation concept in CORE, it is necessary to make a detour and consider the notions of attitude and emotion, and their role in decision-making. This is due to the fact that RE is a decision setting, in which choices are made ultimately by a software engineer, but under the constraint of satisfying the stakeholders. Consequently, the aim at present is to see how attitudes and emotions influence what stakeholders consider satisfactory, and how the attitudes and emotions that they convey can be referred to in CORE in order to be useful in RE.

Attitude. The notions of attitude, emotion, feeling, and mood are strongly related. Attitude has been defined as “a psychological tendency that is expressed by evaluating a particular entity with some degree of favor or disfavor” (Eagly & Chaiken, 1998). An attitude thus gives an evaluation in terms of degree of favor or disfavor. Such degrees vary in sign (positive or negative) and in intensity, whereby the intensity of the valuation is relative: considering an object of attitude on its own involves implicit comparison to a set of objects perceived by the evaluator to be of the same kind. As Kahneman et al. (1999) observe, “objects of attitudes [as the term is used psychology] include anything that people can like or dislike, wish to protect or to harm, to acquire or to reject”. Attitudes can arise from any combination of affective, cognitive and behavioral input (Bizer et al., 2003). Emotions, feelings (i.e., conscious subjective experience of emotion), and moods (i.e., long lasting, less intense, less specific affective states) associated with the entity being evaluated through past or current experience provide the affective basis of attitude. Attributes, or qualities associated to the entity give the cognitive basis of attitude, whereby it is the sign and intensity of evaluation associated with the individual qualities that are transferred on the entity. The behavioral basis of attitude is found in the individual’s interpretation of past behavior and the intentions to commit future acts. It is further relevant to note that separate attitudes on two entities do not necessarily predict the outcome of a choice or direct comparison of these entities (Hsee, 1996; Kahneman et al., 1999). Attitudes are richer and less well-behaving in mathematical terms than those of utility or conventional preference, used often in economics and artificial intelligence. It has been suggested, following Kahneman et al. (1999) that the consistency and short-term stability of preferences (Sen, 1973) are missing in attitudes. Moreover, attitudes violate extensionality, as the same entity may evoke different evaluations depending on how it is described and the context, in which evaluation takes place (Kahneman et al., 1999).

Emotion. Intuitively, emotions influence decision-making, in the sense that they influence evaluations of alternatives that are considered in a decision situation. In RE the stakeholders convey their evaluations

to the software engineer via expressive speech acts. The software engineer decides, while accounting for the evaluations of the stakeholders: the evaluations that they convey enter into the deliberation that the software engineer performs. Question is: how do emotions influence the ranking of alternatives in a decision situation? To say that they give evaluations is not precise enough. Economists have relatively recently started to study the role of emotions in decision-making. According to Elster's review of the "emotion" concept in economics research (Elster, 1998, p. 64), "[by] far the most common way of modeling the interaction between emotions and interests [i.e., values that the decision maker sees in alternative choices] is to view the former as psychic costs or benefits that enter into the utility function on par with satisfaction derived from material rewards". If we take emotions to shape the reward parameters in a decision situation, we can say that emotions are a source of economic preferences. Elster argues, however, that emotions "also affect the ability to make rational choices within those parameters [i.e., the parameters that describe a decision situation]" (Elster, 1998, p. 73). Emotions can thus lead to a bias towards particular alternatives, which finds support in neuroscience research (Dunn et al., 2006; Camerer et al., 2005). An important contribution with regards to emotions' role of bias in decision-making is Damasio's somatic marker hypothesis (Damasio et al., 1991; Damasio, 1994):

[A generic situation is described, in which a choice is to be made.] [...] imagine that *before* you apply any kind of cost/benefit analysis to the premises, and before you reason toward the solution of the problem, something quite important happens: When the bad outcome connected with a given response option comes into mind, however fleetingly, you experience an unpleasant gut feeling. Because the feeling is about the body, I gave the phenomenon the technical term *somatic* state [...] and because it 'marks' and image, I called it a *marker* [...] What does the *somatic marker* achieve? It forces attention on the negative outcome to which a given action may lead, and functions as an automated alarm signal [...] The signal may lead you to reject, *immediately*, the negative course of action and thus makes you choose among alternatives. The automated signal protects you against future losses, without further ado, and then allows you *to choose among fewer alternatives*. There is still room for using a cost/benefit analysis and proper deductive competence, but only *after* the automated step greatly reduces the number of options [...] This general account applies to the choice of actions whose immediate consequences are negative, but which generate positive future outcomes. (Damasio, 1994, pp. 170–175; italics in original.)

Choice behavior is thus seen as combining cost/benefit analysis with marker signals that indicate how rewarding or punishing an alternative choice is likely to be (Dunn et al., 2006).

Our intention with the Evaluation concept is to capture what expressives convey, which includes both attitudes and emotions. This is a safe choice, given the limited understanding of these phenomena in psychology, or in economics. Our definition of Evaluation is grounded in the following observations, which follow from the above discussions of attitudes and emotions:

1. There are no constraints on the entities that may be objects of attitudes (e.g., Eagly & Chaiken, 1998; Kahneman et al., 1999). Stakeholders can thus evaluate favorably or unfavorably instances of any concepts in CORE: e.g., Functional goal (Ex. 15), Quality constraint (Ex. 16), Softgoal (Ex. 17), Plan (Ex. 18), Functional domain assumption (Ex. 19), and so on.

It would be good if the user is informed of special flight offers. (Ex. 15)

It is preferred to split the flight booking form over several screens than to show it fully on a single screen. (Ex. 16)

Convenience of flight booking is more important than the speed of booking. (Ex. 17)

Having the system confirm booking is preferred to having a person do it. (Ex. 18)

Rule in Ex. 2 is unacceptable for it limits the pricing options in case of promotions. (Ex. 19)

2. Attitudes are intertwined with emotions. Communicating attitudes certainly conveys the underlying emotions. The latter give rise to somatic markers on alternative behaviors in decision-making (e.g., Damasio, 1994; Dunn et al., 2006), thereby effectively ranking the alternative courses of action. This is important here in two respects. Expressives may convey temporary preference orders among alternatives. An evaluation may thus capture a temporary preference order among two or more of the explicit alternatives. Because attitudes violate extensionality (among other typical characteristics of standard economic preferences – see (Kahneman et al., 1999)), the temporary preference order need not be stable, transitive, or necessarily obey some other property.
3. Following the preceding point, somatic markers may have much stronger effect on the set of considered alternatives: not only may they order alternatives, but the dominated alternatives may be entirely eliminated from those considered. In case of explicit alternatives, a single alternative is taken, all others discarded. An instance of Evaluation may thus cut through the set of considered alternatives.

As the software engineer hopes to construct high quality systems, she must be interested in evaluations: they reflect stakeholders' level of satisfaction with plans, functional goals, quality constraints, and so on, thereby guiding the engineer during the decision-making on whether and how to define plans to satisfy functional goals, softgoals, quality constraints, and avoid violating functional domain assumptions, quality domain assumptions, and soft domain assumptions. What the software engineer must have in order to rate alternative plans is certainly an explicit account of stakeholders' evaluations. In conclusion, we have the following definition of the Evaluation concept in CORE:

Definition 11 (Evaluation). Any instance x of Communicated information is an instance of Evaluation if and only if:

- (1) if x is communicated via an expressive speech act, and
- (2) either x conveys the evaluation, in terms of favor or disfavor, a single Particular, or
- (3) x conveys a comparison in terms of favor or disfavor of two or more other Particulars.

To reflect in CORE that an evaluation can be given on one Particular (e.g., Exs 15 and 19), or may amount to a comparison of instances (e.g., Exs 16–18), we introduce the Individual evaluation and Comparative evaluation concepts.

Definition 12 (Individual evaluation). Any instance of Evaluation is an instance of Individual evaluation if and only if it conveys the evaluation, in terms of favor or disfavor, a single Particular.

Definition 13 (Comparative evaluation). Any instance of Evaluation is an instance of Comparative evaluation if and only if it conveys a comparison in terms of favor or disfavor of two or more other Particulars.

Since we are referring to DOLCE Particulars in Definition 11, it should be noted that an evaluation can evaluate another evaluation, or compare other evaluations. In very practical terms, allowing this reflects a simple idea: if two orders of temporary preferences (i.e., comparative evaluations) are inconsistent,

choice may be guided by only one of the two, which entails that an order is established between the two temporary preferences. This is not to say that, for instance, emotions can be chosen for it appears agreed that they are passively undergone (e.g., Elster, 1998). Instead, in the RE setting, a stakeholder may acknowledge that considered alternatives ought to be ranked not according to her own temporary preference order but someone else's. Although the underlying emotion that gave rise to the former actor's temporary preferences may not have changed as the result of accepting externally to act otherwise, the actor has conveyed (possibly by lack of action) that one order of preference is (temporarily) acceptable over another.

For instance, lower payment verification time may correlate with lower payment security in a flight booking system. In that case, aiming to satisfy one preferred softgoal negatively affects the ability to satisfy some other softgoal; consequently, the involved comparative evaluations are conflicting. If the conflict cannot be alleviated through a different design of the system-to-be (e.g., new payment verification servers with better performance both in payment verification and security), the relative importance of the conflicting comparative evaluations should be determined. This relative importance will simply be given by another comparative evaluation, this time between the conflicting comparative evaluations. The following two comparative evaluations over functional goals give a conflict:

Issuing electronic flight tickets is preferred to issuing paper flight tickets. (Ex. 20)

It is preferred that all travelers have paper tickets than only those who have had access to a printer after the booking. (Ex. 21)

The answer is to choose which of the two is more favorably evaluated. This results in adopting plans (and thereby a system design) that satisfy the most favorably evaluated option (that is, functional goal in the above example) in the preferred comparative evaluation. In other words, if the comparative evaluation obtained from the second statement is more important (i.e., obtains a more intense favorable evaluation), then a system design obeying the comparative evaluation in the first statement will be evaluated to a less favorable (or higher disfavorable) extent than a design that obeys the first comparative evaluation.

Let **FG**, **QC**, **SG**, **P**, **FK**, **QK**, **SK** and **EC** be the sets of instances of, respectively Functional goal, Quality constraint, Softgoal, Plan, Functional domain assumption, Quality domain assumption, Soft domain assumption, or Comparative evaluation that are considered in an RE project. Also, let each **fg**, **qc**, **sg**, **p**, **fk**, **qk**, **sk** and **ec** be a typical member of, respectively, **FG**, **QC**, **SG**, **P**, **FK**, **QK**, **SK** and **EC**. Note the abbreviations: **fg** \equiv **fg** ϕ , **qc** \equiv **qc** ψ , and so on.⁵

Favorable or disfavorable evaluation of a single **fg**, **qc**, **sg**, **p**, **fk**, **qk**, **sk** or **ec** is of interest to the software engineer to the extent that it indicates whether it is necessary to specify the system that accords with the given **fg**, **qc**, **sg**, **p**, **fk**, **qk**, **sk** or **ec**. In other words, the engineer is interested in knowing if some **fg**, **qc**, **sg**, **p**, **fk**, **qk**, **sk** or **ec** is *compulsory* or *optional*. It is compulsory if the stakeholders cannot accept a specification that does not account for that particular **fg**, **qc**, **sg**, **p**, **fk**, **qk**, **sk** or **ec**; otherwise, it is optional.

Individual evaluations completely partition each of **FG**, **QC**, **SG**, **P**, **FK**, **QK**, **SK** and **EC** onto optional and compulsory parts: **FG** on **FG**^O and **FG**^C, **QC** on **QC**^O and **QC**^C, and so on. The dichotomy

⁵All notational conventions are summarized in the Appendix A.

optional/compulsory arises from the intensity of evaluation: favorable or unfavorable evaluation involves undoubtedly an ill-structured and subjective evaluation space, so that the chosen dichotomy serves to partition that space and thereby simplify the use of evaluations over instances of Functional goal, Quality constraint, Softgoal, Plan, Functional domain assumption, Quality domain assumption, Soft domain assumption, or Comparative evaluation. The actual intensity becomes of interest once tradeoffs appear – i.e., when it is necessary to determine which of two or more optional or compulsory, but conflicting softgoals, plans, or otherwise, need to be satisfied. Note that optionality is a means for comparing alternative (specifications of) variants of the system-to-be, whereby those that satisfy “more” among FG^O , QC^O , SG^O , P^O , FK^O , QK^O , SK^O and EC^O is more desired than another that satisfies “less” of these, all other things being equal.

Optionality also provides an interesting response to the problems of optimistic (i.e., idealistic) and pessimistic quality constraints. However systematic is the development process of any realistic system, uncertainty remains about how the system will actually operate once deployed (e.g., due to failures, introduction of new components, unanticipated interactions, etc.). This leads to the problems of idealistic and pessimistic quality constraints. A quality constraint that will not be satisfied by the system is *optimistic* (e.g., expecting a response time lower than the feasible minimum), while a quality constraint that can be satisfied “better” than what is expected is *pessimistic* (e.g., expecting a response time higher than what the system actually can achieve). Although we cannot entirely avoid either idealistic or pessimistic quality constraints, we can separate *optional* from *compulsory* quality constraints: if a quality constraint is likely to be optimistic, it may be possible to rewrite it as several quality constraints, some of which are compulsory while others are optional. Depending then on the actual system operation, we can have (some of) the optional quality constraints satisfied, so that we avoid (i) missing entirely the satisfaction of the initial optimistic quality constraint, or (ii) satisfying only the compulsory quality constraints and missing all optional ones (in this situation, we actually obtain a pessimistic quality constraint from the initially optimistic one). For example, the initial quality constraint “*Inform the traveler of changes in flight time and airport gate during the hour before departure of the booked flight*” is idealistic since we do not always have access to this information in a timely manner (though when we do, we can use it). If we were to consider this quality constraint as compulsory, we would hope to ensure during development that it is always satisfied by the system design. We would then undoubtedly replace it with a more realistic form, e.g., by hoping to inform the traveler of changes 6 hours before the flight. In many cases, the revised quality constraint is actually pessimistic since there are cases when we do have access to the relevant information in a timely manner and we can use it to inform the traveler within the hour preceding the flight. With optional vs. compulsory distinction, we keep the revised quality constraint as compulsory, while we add “*If the information is available on flight time and/or gate changes during the hour before the flight, inform the traveler of these changes*” as an optional quality constraint.

5. Relationships

This section introduces the relationships that we will use in order to obtain the definition of the requirements problem. Four relationships are discussed: j-approximate (Section 5.1), j-refine (Section 5.2), e-compare and evaluate (Section 5.3). The j-approximate and j-refine relationships share a common characteristic in that they both use defeasible reasoning. Appendix C recalls the basics of defeasible reasoning via argumentation.

5.1. Justified approximation

The so-called *justified approximation* is a relationship that we introduce between quality constraints and softgoals. The necessity for j-approximate arises from the salient characteristic of softgoals, which is that a softgoal cannot be satisfied to the extent hoped for by the stakeholders. This is due to subjective (i.e., not shared) quality spaces and the tendency to expect idealistic satisfaction levels, that is, those that are beyond the resources available to the stakeholders the system-to-be.

The original softgoal concept has been introduced in RE by Mylopoulos et al. (1992, 1999) to cover nonfunctional requirements that cannot be satisfied in an objectively verifiable way. That line of research concluded that the best a software engineer can do is obtain a specification that *satisfices* softgoals. *Satisficing* is a term that has, to the best of our knowledge, been introduced by Simon (1955) in his well-known critique of the assumptions behind the economic model of rational decision-making. Simon's behavioral model of rational choice challenges such an understanding of decision behavior, by arguing that the inherent limitations of the decision-maker restrict the information that can be considered and processed when making a choice: "actual human rationality-striving can at best be an extremely crude and simplified approximation of the kind of global rationality that is implied, for example, by game-theoretical models" (Simon, 1955, p. 101). He went on to suggest a decision procedure that is more realistic: given a minimal acceptable (i.e., threshold) level of satisfaction, the aim of the decision-maker is to identify some possible outcomes that are above the threshold, then identify actions that can bring about one of these outcomes. That is the essence of the satisficing approach to decision making. Instead of seeking to bring about the optimal outcome, the decision-maker seeks any one above a given threshold.

Following the general ideas discussed by Simon, Mylopoulos et al. argued that a softgoal can only be satisfied, for optimal solutions are beyond the reach in practice. However, Mylopoulos et al. associated a different meaning to satisficing, taking that it occurs when the "best" alternative (i.e., one that satisfies the softgoal to the highest desirable extent compared to other alternatives) is chosen among all *considered* alternatives. In practice, these are the alternatives identified over the course of decision-making, which in RE means that they are alternative (specifications of) variants of the system-to-be identified during RE.

The question that becomes relevant is how one can determine the extent to which a softgoal is satisfied by a specification. The answer from Mylopoulos et al. was to introduce the so-called "contribution links" between goals and softgoals to indicate that bringing about states in which the goal is satisfied contributes positively or negatively to the satisficing of the softgoal. The aim with contribution links is to enable the engineer to claim that satisfying functional goals in some particular way leads to some degree of satisfaction of the softgoals. A contribution link relates a goal to a softgoal, in order to state that to satisfy the former leads the latter to be satisfied to some extent. The extent, or degree of satisfaction is indicated by a label on the contribution link. The label is typically one of the following four: strong positive (i.e., if the functional goal is satisfied, the softgoal is satisfied to some significant extent), positive, negative, strong negative. Giorgini et al. (2003) subsequently suggested techniques for the automated propagation of satisfaction intensities through graphs in which goals and softgoals are nodes, and contribution links are edges. This gave a relevant approach to the reasoning with contribution links. What is still missing in RE is knowing *why* a contribution can be defined towards a softgoal from an instance of another concept, or, in other words: what evidence and rationale led to believe that it makes sense to define that contribution. It has been observed already (e.g., Letier & van Lamsweerde, 2004) that the rationale behind the definition of contribution links between goals and softgoals remains the discretion of the engineer or another participant in the RE process, meaning that choices and arguments thereto can

remain implicit, leading to the conclusion that reasoning by contribution links about the satisficing of the softgoals cannot be a rigorous decision process.

Given these remarks, we do not use the contribution relationship in its original form. The justified approximation relationship allows us to draw at least as elaborate conclusions about the satisficing of softgoals, as the contribution relationship does. The justified approximation relationship is designed to counter the deficiencies of the contribution relationship, and arises from the finer understanding of functional goals, quality constraints, and softgoals made possible with CORE. Recall that the softgoal constrains values of a quality defined over a subjective and/or ill-structured quality space. Because one cannot manage what one cannot measure, when the engineer aims to ensure the convenience of the flight booking process, she will try to understand how various values of measurable system behaviors affect stakeholders' perception of convenience. For instance, she may consider that six screens for flight booking correspond to a threshold level of convenience, and that convenience improves as the number of screens goes down, whereby anything less than three screens is unacceptable, for the user is expected to fill out too big a form over too few screens. *Doing this amounts to approximate a subjective quality space by one or more other, shared quality spaces.* There is therefore a relationship between quality constraints and softgoals reflecting that a quality constraint approximates a softgoal in the sense that there is correlation between the values in the quality space of the former and the quality space of the latter. Now, classical contribution links in RE are labeled: e.g., if we choose a system design in which there will be less than five screens for flight booking, we will label the contribution link as positive (we would draw the symbol '+' on it). The label is not itself related to the sign of the correlation (e.g., the number of screens for flight booking may be negatively correlated with the 'level' of convenience) and there is no absolute rule for deriving one from the other. However, it has not been very clear in RE – for lack of a precise definition of the softgoal concept – what it means for a goal to correlate with a softgoal. We see here that it is reasonable to argue that it is instead a quality constraint that can correlate with a softgoal, and not a functional goal.

Why approximate convenience with the number of screens required to perform some task, and not, say, the time it takes a typical user to complete that task? Not all approximations are equivalently appropriate or acceptable to the stakeholders. Evidently, the engineer cannot prove (in the formal sense) that it is indeed relevant to approximate convenience with the number of screens in flight booking and that the correlation is as described above; the closest feasible solution instead is to *justify* that a quality referred to in a quality constraint approximates a quality referred to in a softgoal. In other words, *we are interested in justifying that a quality constraint approximates a softgoal*, whereby the justification applies under two caveats:

- Approximation stands for all practical purposes within the given development project.
- Approximation is justified only on grounds of the information available to the software engineer (i.e., new information – e.g., more details on how stakeholders evaluate convenience – may lead the engineer to revise the approximation that was previously justified).

The first caveat means that approximating convenience by (among others) the number of screens for booking is justified only locally, within the given project – it may be appropriate in the case of another system, but that remains to be justified; the second caveat means that the approximation can become irrelevant, e.g., if the engineer finds out that stakeholders care in no way for the number of screens, and that convenience is merely related to the length of the form to fill when booking a flight.

To arrive at the justified approximation relationship, we need to combine the correlation between quality constraints and softgoals, and the need for the justification of that approximation. But there is more. Say that we have the following softgoal:

sg: Flight booking should be convenient. (Ex. 22)

It is perfectly acceptable to understand from the above that the “convenience” of flight booking should be optimized. Consequently, not only does a softgoal involve a subjective quality space, but it also incorporates information about a desired *direction* to pursue in that quality space. Of course, the direction will hardly be precisely known because the quality space is subjective. Still, information about the desired direction cannot be disregarded. Say that the number of screens (displayed to the user while booking a flight) correlates with the values in the quality space for “convenience”. The following quality constraint may then be given as a threshold for the convenience of flight booking.

qc: Flight booking takes at most 6 screens. (Ex. 23)

Any flight booking process that takes at most 6 screens is then deemed as convenient. However, not all booking processes that do fall within the convenient ones are necessarily equally desirable. Indeed, if convenience is to be optimized, then justifiably approximating **sg** with **qc** alone amounts to disregard an important piece of information. In other words, *softgoals incorporate comparative evaluations*. Above, **qc** should be combined with an instance **ec** of Comparative evaluation in order for us to claim that **sg** is justifiably approximated. Such an **ec** may be:

ec: For $X > 1$, number $X - 1$ of screens is preferred to number X screens for flight booking. (Ex. 24)

To arrive at the justified approximation relationship, we need to combine the correlation between quality constraints and softgoals, the need for the justification of that approximation, and the comparative evaluations. Following these remarks, we can define the j-approximate relationship. As the definition is rather elaborate, we start with several introductory remarks that will prove helpful:

- Definition 14 will point out that two ingredients go into any justified approximation. The first ingredient are the relata: we have the softgoal or soft domain assumption that is being approximated (\mathbf{z} in Definition 14), and the one or more instances of concepts in CORE, which we take to approximate the former ($\mathbf{x}_1, \dots, \mathbf{x}_n$ in Definition 14). Since alternative quality constraints can approximate a softgoal or soft domain constraint, we can have instances of comparative evaluations among the instances that approximate in order to compare the alternative quality constraints ($\mathbf{y}_1, \dots, \mathbf{y}_m$ in the same definition). The second ingredient are the justifications for two assertions. One of them says that values in the quality space of each quality referred to in the approximating instances $\mathbf{x}_1, \dots, \mathbf{x}_n$ correlate sufficiently with the values in the quality space of the approximated softgoal or soft domain constraint \mathbf{z} . The second assertion that needs to be justified relates the satisfaction of $\mathbf{x}_1, \dots, \mathbf{x}_n$ to the satisfaction of \mathbf{z} .
- Since we can very well have any or both $n > 1$ and $m > 0$ in, respectively, $\mathbf{x}_1, \dots, \mathbf{x}_n$ and $\mathbf{y}_1, \dots, \mathbf{y}_m$, it is necessary to organize these instances when relating them to \mathbf{z} . Consequently, Definition 14 says how to build what we call the justified approximation tree. For a softgoal or soft domain assumption \mathbf{z} , we denote $T_A(\mathbf{z})$ the justified approximation tree, where T refers to the tree,

the subscript in \mathcal{T}_A refers to “approximation”, the \mathbf{z} in $\mathcal{T}_A(\mathbf{z})$ refers to the instance being approximated and that acts as the root of the justified approximation tree. We write $\mathcal{T}_A(\mathbf{z}) \approx_{\triangleright} \mathbf{z}$ to state that $\mathcal{T}_A(\mathbf{z})$ justifiably approximates \mathbf{z} .

- When we say that $\mathbf{x}_1, \dots, \mathbf{x}_n$ and $\mathbf{y}_1, \dots, \mathbf{y}_m$ are organized in a justified approximation tree, this means that we use the tree to make explicit the relations between these instances. Hence the need for the means – namely, unary and n-ary modifiers – to denote these relations. What are these unary (written U) and n-ary (written N) modifiers in Definition 14? They are similar to unary and binary operators in a formal logic; the difference is that we do not require them to be defined via formal semantics to be used during early RE. This is due to the fact that we manipulate formally specified requirements only in the late steps of RE, if at all. Unary and n-ary modifiers allow us to have various kinds of relations between instances that participate in the justified approximation; we are not limited to conjunction and disjunction. For example, Always in the future, Eventually in the future, Always in the past, and Eventually in the past are useful to separate rough temporal determinants about a condition given in a concrete instance. More elaborate unary operators for time would follow classical developments in temporal logics (Bellini et al., 2000): for example, a metric can be introduced in order to express temporal constraints in a quantitative form (e.g., allowed duration of a condition). N-ary modifiers might take the form of conjunction, disjunction, conditional (e.g., if-then, while-do), temporal orderings (e.g., -before-, -meets-, -starts-, etc.), or synchronization structures. We do not limit the library of unary or n-ary modifiers in this paper, but leave it open: The relevance of various potential modifiers will be determined from the construction of, and experience in the use of RE methodologies built on top of CORE.

Definition 14 (Justified approximation). Let \mathbf{z} be an instance of either Softgoal or Soft domain assumption. Let any \mathbf{x} be an instance of some other concept in CORE. Let \mathbf{y} be an instance of Comparative evaluation. There is a j-approximate relationship from two non-empty sets of instances $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and $\{\mathbf{y}_1, \dots, \mathbf{y}_m\}$ to \mathbf{z} , written $\mathcal{T}_A(\mathbf{z}) \approx_{\triangleright} \mathbf{z}$ if and only if:

- (1) the assertion “values in the quality space of each quality referred to in $\mathbf{x}_1, \dots, \mathbf{x}_n$ correlate sufficiently with values in the quality space of the quality referred to in \mathbf{z} ” is justified; and
- (2) the assertion “satisfying $root(\mathcal{T}_A(\mathbf{z}))$ satisfies \mathbf{z} ” is justified.

where the justified approximation tree $\mathcal{T}_A(\mathbf{z})$ for \mathbf{z} is built by applying the following steps:

1. Set each \mathbf{x}_i in $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and each \mathbf{y}_j in $\{\mathbf{y}_1, \dots, \mathbf{y}_m\}$ as the leaf of the tree $\mathcal{T}_A(\mathbf{z})$. Call a node without outgoing edges a “top node”.
2. For each top node \mathbf{x}_i that has a unary modifier U for \mathbf{x}_i , create a node $U\mathbf{x}_i$ and an edge directed from the node \mathbf{x}_i to the node $U\mathbf{x}_i$. Label that edge with the unary modifier U.
3. For each top node \mathbf{y}_j that has a unary modifier U for \mathbf{y}_j , create a node $U\mathbf{y}_j$ and an edge directed from the node \mathbf{y}_j to the node $U\mathbf{y}_j$. Label that edge with the unary modifier U.
4. For each subset T of top nodes related by a single n-ary modifier N, create an edge $N(T)$ and create from each member of T an edge directed to $N(T)$. Label each of these edges with the n-ary modifier N.
5. Repeat steps 2–4 until $\mathcal{T}_A(\mathbf{z})$ is a tree and no unary or n-ary modifiers need to be applied to the root of $\mathcal{T}_A(\mathbf{z})$.

Several additional remarks are in order after the definition has been given:

- Given how the justification process proceeds, some **sg** is justifiably approximated if (i) there is no evidence to contradict the assertion “values in the quality space of each quality referred to in $\mathbf{x}_1, \dots, \mathbf{x}_n$ correlate sufficiently with values in the quality space of the quality referred to in \mathbf{z} ”, and (ii) there is no evidence to contradict the assertion “satisfying $root(\mathcal{T}_A(\mathbf{z}))$ satisfies \mathbf{z} ”. The point here is that we only proceed to the justification of any of the two assertions above if evidence to the contrary is made explicit.
- Definition 14 speaks of “sufficient” correlation, leaving the sufficient level to be fixed case by case, and for the given project. We impose no constraints within CORE with regards to the sufficiency of some correlation.
- We do not necessarily have a quantified correlation between a subjective and shared quality space. The only requirement is that all evidence to the contrary of that correlation be discarded via the justification process.

To claim that softgoals are satisfied, the engineer must identify instances of CORE concepts that justifiably approximate these softgoals. As a first illustration, consider the justified approximation tree shown in Fig. 3 for the softgoal in Ex. 22. Notice that the root of the justified approximation tree is:

$$root(\mathcal{T}_A(\mathbf{sg})) = \mathbf{qc} \text{ AND } \mathbf{ec}.$$

The approximation is justified if there is no evidence (i) that the quality “number of screens” does not correlate sufficiently with the quality “convenience of flight booking”, and (ii) that to satisfy **qc AND ec** does not satisfy **sg**. Assuming that such evidence is indeed absent, we can state that $\mathcal{T}_A(\mathbf{sg}) \approx \triangleright \mathbf{sg}$.

As another example, consider the following softgoal for the online flight booking application:

$$\mathbf{sg}_1: \text{ User registration should be usable.} \quad (\text{Ex. 25})$$

We are interested in quality constraints that stand in the justified approximation relationship with this softgoal. Drawing from an industry report Nielsen et al. (2001) on usability in e-commerce that is based on user observations, we can suggest a number of quality constraints that together approximate the above softgoal:

$$\mathbf{qc}_1: \text{ User needs not register to purchase.} \quad (\text{Ex. 26})$$

$$\mathbf{qc}_2: \text{ Benefits of registration are shown when registering.} \quad (\text{Ex. 27})$$

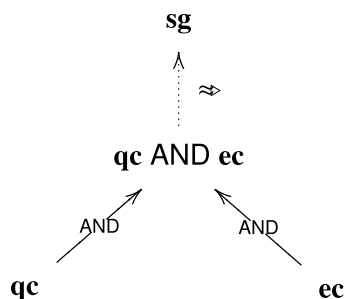


Fig. 3. Justified approximation of instance **sg** of the Softgoal concept, given in Ex. 22.

qc₃: Privacy policy is linked on every screen. (Ex. 28)

qc₄: The login process asks the user for her email, and not the username. (Ex. 29)

qc₅: Password recovery by "secret question" is not the only way to recover a forgotten password. (Ex. 30)

qc₆: Help on choosing the username and password is understood by at least 90% of test users. (Ex. 31)

We can add the following comparative evaluations:

ec₁: It is preferred to let unregistered users to purchase than to impose registration prior to purchase. (Ex. 32)

ec₂: The higher the proportion of test users that understand the help on choosing the username and password, the better. (Ex. 33)

Figure 4 shows the justified approximation tree $\mathcal{T}_A(\mathbf{sg}_1)$. Figure 5 provides the dialectical tree for the assertion "values in the quality space of **qc**₁ correlate sufficiently with values in the quality space of the quality referred to in **sg**₁". It is valid to write $\mathbf{qc}_1 \approx \triangleright \mathbf{sg}_1$ (or, in other words, it is valid to assert "values in the quality space of **qc**₁ correlate sufficiently with values in the quality space of the quality referred to in **sg**₁", because the only argument (i.e., Arg.1.6) that disagrees with that assertion is defeated by undefeated arguments Arg.1.6.1 and Arg.1.6.2.

Compared to the justified approximation relationship, contribution links in RE do not recognize the importance of justification and correlation. Classical contribution links are indirect: they link functional

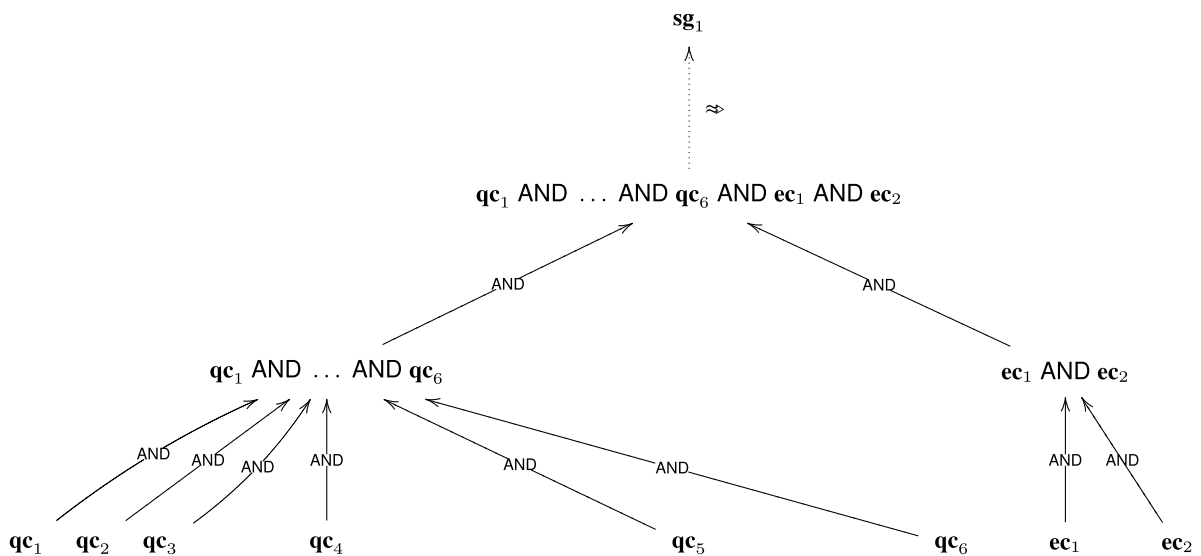
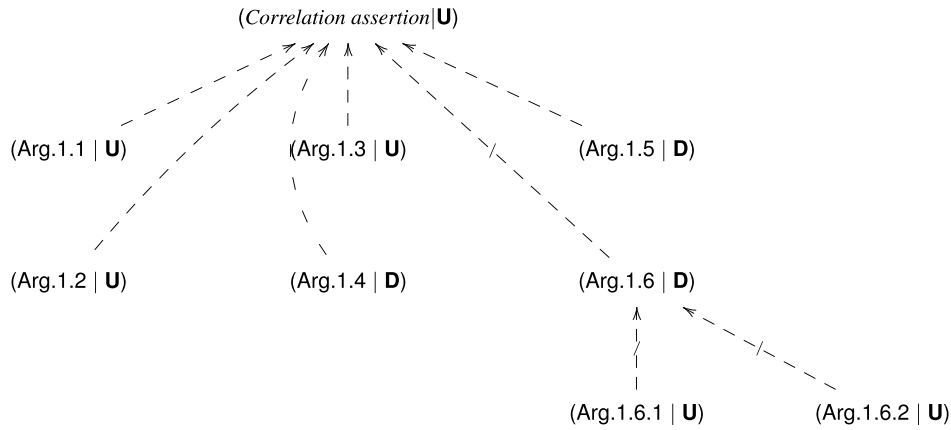


Fig. 4. Justified approximation of instance **sg**₁ of the Softgoal concept, given in Ex. 25.



Arguments used in the dialectical tree above:

- (Arg.1.1) Four of eight sites that required registration experienced complete purchase failures with at least one user because of login or registration.
- (Arg.1.2) Optional registration is perceived as less offensive than compulsory registration.
- (Arg.1.3) Registration is perceived as annoying.
- (Arg.1.4) Compulsory registration stimulates the fear of being charged something.
- (Arg.1.5) Compulsory registration raises privacy concerns and fear of unsolicited e-mails.
- (Arg.1.6) Registration is necessary for a secure checkout.
- (Arg.1.6.1) Except for a username and password, the user provides the same personal information on checkout as during registration.
- (Arg.1.6.2) Regardless of registration, SSL encryption makes it difficult to intercept the communication between the user and the server during the checkout process.

Fig. 5. Dialectical tree for the assertion: “values in the quality space of qc_1 correlate sufficiently with values in the quality space of the quality referred to in sg_1 ”. Content of arguments appearing in the dialectical tree is listed below the dialectical tree. *Correlation assertion* stands for the assertion “values in the quality space of qc_1 correlate sufficiently with values in the quality space of the quality referred to in sg_1 ”.

goals to softgoals. It is more appropriate to speak of correlation between observed qualities over the system behaviors that satisfy functional goals (given by qualities referred to in quality constraints) and softgoals, than to abstract from quality constraints, as is the case in classical contribution links. *j*-approximate highlights the importance of correlation and justification for the existence of a the approximation relationship, while not rejecting the classical contribution links. Classical contribution links remain as preliminary, unjustified approximations, which point to potential justified approximations.

5.2. Justified refinement

The process of *program refinement* amounts to replace a piece of abstract program with a piece of more concrete program (Dijkstra, 1972; Hoare, 1972; Wirth, 1971), thereby incorporating lower level detail. Considerations of lower levels can thereby be delayed to later steps of system development.

AND-refinement and *OR-refinement* relationships, which have by now become part of the standard body of knowledge in RE, aim to bring this same benefit. Just as program refinement, they allow detail to be incrementally added during RE, while ensuring that this additional detail remains consistent with the information found at more abstract levels. In *AND-* and *OR-refinement*, to refine means to replace the refined instance of a concept by other instances of the same concept. For example, refining a functional

goal involves the identification of other functional goals whose satisfaction equates to satisfying the refined goal. Hence, if there is an AND-refinement relationship between a functional goal and two other functional goals, satisfying the latter two guarantees that the former functional goal is satisfied. If there is an OR-refinement relationship between a goal and two other goals, satisfying either of the latter equates to satisfy the former goal. The relevant question at present is whether AND- and OR-refinement can apply to concepts in CORE. Does j-refine equate to either of AND- or OR-refinement relationships? To answer this question, we need to recall the precise definition of classical AND- and OR-refinement relationships in RE.

Darimont & van Lamsweerde's (1996) definition of a complete AND-refinement of a goal remains a standard one in RE. Namely, a set of goals g_1, g_2, \dots, g_n is a complete AND-refinement of a goal g if and only if the following conditions hold: (a) $g_1 \wedge g_2 \wedge \dots \wedge g_n \vdash g$, (b) for all $i \in \{1, \dots, n\}$, $\bigwedge_{j \neq i} g_j \not\vdash g$, (c) $g_1 \wedge g_2 \wedge \dots \wedge g_n \not\vdash \perp$, and (d) $n > 1$. The first condition ensures that satisfying all goals obtained by refinement leads to the satisfaction of the refined goal; the second condition ensures that the refined goals are only those necessary for the satisfaction of the AND-refined goal; the third condition indicates that the AND-refined goals must be consistent; the fourth avoids trivial refinement (e.g., rewriting the refined goal in an equivalent form). For OR-refinement, the only condition is that satisfying either of the goals obtained by refinement alone is enough to obtain the satisfaction of the OR-refined goal; there is no condition for minimality or consistency.

In attempting to reuse the standard AND- and OR-refinement, we face the following problems:

1. Instances of Goal, Functional goal, Quality constraint and/or Softgoal do not by default resemble instances of Darimont and van Lamsweerde's "goal" concept. While some members of **G**, **FG**, **QC** and/or **SG** may be known precisely enough by the software engineer to warrant a formulation in a formal logic, such degree of precision only comes late in the RE process. In other words, there are members of Goal, Functional goal, Quality constraint and/or Softgoal that do relate via some form of refinement, but that refinement does not satisfy the conditions of AND- or OR-refinement. A weaker kind of the refinement relationship is needed in CORE.
2. The term "refinement" has a well-established meaning in RE: it designates an analysis technique. An important, if not key benefit of the refinement technique is that it leads us to substitute a less concrete instance with a combination of more concrete instances. In this respect, AND- and OR-refinement are conceptualizations that concretize the result of applying the refinement technique. We see this in AND-refinement, where g_1, g_2, \dots, g_n substitute for g . Now, observe that the refinement technique gives two pieces of information: the more concrete instances *and* how these instances are combined. In AND-refinement, we know (a) that g_1, g_2, \dots, g_n substitute g , and (ii) g_1, g_2, \dots, g_n stand in conjunction, i.e., $g_1 \wedge g_2 \wedge \dots \wedge g_n$ substitute g . OR-refinement tells us that (a) that g_1, g_2, \dots, g_n substitute g , and (ii) g_1, g_2, \dots, g_n stand in disjunction, i.e., *either* g_1 , *or* g_2 , *or* ... *or* g_n substitute g . The question to ask here is: Why do we have only AND- and OR-refinement in RE? One could expect the answer to be that all possible relationships can be broken down to conjunction and disjunction between more concrete instances. That answer is, however, evidently false. We reuse here a statement from an earlier example:

At present, no business seat has a lower price than an economy seat if there are no promotions on business seats. (Ex. 34)

As we noted earlier, this statement involves two conditionally related assertives. If our toolset includes only AND- and OR-refinement, we are led to conclude that the above statement cannot be

refined, for there are no lines of conjunction or disjunction along which to decompose it. However, it may well be useful to consider separately the condition from the conditioned content in that statement. The reason is evident: we decompose the statement in order to focus on its pieces, one at a time. Conditionality, as in *if x then y* cannot be equated to conjuncts and/or disjuncts of *x* and *y*, unless the software engineer accepts the paradoxes that go with equating conditionality to material implication, i.e., $x \rightarrow y$, and then rewriting it as $\neg x \vee y$ or $\neg(x \wedge \neg y)$.⁶ A more general form of the refinement relationship is needed in CORE.⁷

3. A third problem arises from the fact that both AND- and OR-refinement are defined over instances of the same concepts: a goal can only be refined by goals. Consider the following directive statement:

*When the user completes a booking, a confirmation should be sent and enriched
with useful information relevant to the user's destination.* (Ex. 35)

The content of the above becomes the content of an instance of Goal. We can refine it by breaking it into three pieces: (i) the condition, which gives us an instance of Functional goal; (ii) the first conditioned part, which is an instance of Functional goal; and (iii) the second conditioned part, which gives an instance of Softgoal (notice the quality “useful”, which undoubtedly involves a subjective quality space). Each of these pieces is obtained by decomposing the original statement, but not all of the pieces are instances of the same concept: there are two functional goals and a softgoal. Neither AND- nor OR-refinement relationships can stand between these atomic statements and the composite one.

Let us start off from Darimont and van Lamsweerde’s AND-refinement, and consider how it could be adapted in response to the first issue raised above (point 1). We can loosen the relationship between the concrete and abstract instances. Namely, concrete instances defeasibly derive the abstract instance. Let x, x_1, x_2, \dots, x_n be instances of some concept in CORE. x_1, x_2, \dots, x_n is a complete *weak* AND-refinement of the instance x if and only if:

- $x_1 \wedge x_2 \wedge \dots \wedge x_n$ defeasibly derive x : $\mathcal{K}, x_1 \wedge x_2 \wedge \dots \wedge x_n \vdash x$;
- the set $\{x_1, x_2, \dots, x_n\}$ is minimal: $\forall i \in \{1, \dots, n\}, \mathcal{K}, \bigwedge_{j \neq i} x_j \not\vdash x$;
- $x_1 \wedge x_2 \wedge \dots \wedge x_n$ are consistent: $\mathcal{K}, x_1 \wedge x_2 \wedge \dots \wedge x_n \not\vdash \perp$.

Above, \mathcal{K} carries all instances of the concepts in our ontology *except* x, x_1, x_2, \dots, x_n . Together with x, x_1, x_2, \dots, x_n , \mathcal{K} is all that is explicitly given about a given system and its relevant environment. The above is a *weak* refinement because we adopt nonmonotony. Recall that we need not have x ’s written in a mathematical logic for the above definition to remain relevant: if we use an informal logic of argumentation, then the above definition indicates simply that $x_1 \wedge x_2 \wedge \dots \wedge x_n$ together with x form an argument, where the former are premises and the latter the tentative conclusion of these premises. \mathcal{K} intervenes in the above definition in order to acknowledge that the concept instances involved in the

⁶Sanford’s (1989) and Bennett’s (2003) discussions are relevant to the reader interested in the difficulties in dealing with conditionals.

⁷There is a nuance in this discussion that may not be immediately apparent: by claiming that we need a more general relationship, we are not arguing that, for instance, the notion of “conditionality” is somehow absent from RE (or at least from Darimont and van Lamsweerde’s discussions). If we have instances written in a formal logic, conditionality can be described to the extent that the formal apparatus – i.e., that formal logic – allows conditionality to be expressed. The problem is that the very notion of conditionality remains within the formulation of the content of the instance, and is thus invisible to the techniques applied in RE. In other words, we do not have techniques in RE to look *inside* the conditional structures – these structures are considered as internal to the atomic instances that RE methodologies manipulate.

refinement should be consistent with the instances of other concepts of the ontology. We thus have an argument in the case of weak refinement, where we can say that it is appropriate to believe that x will obtain when $x_1 \wedge x_2 \wedge \dots \wedge x_n$ obtain, given \mathcal{K} . In other words, the conclusion (x) holds when premises hold ($x_1 \wedge x_2 \wedge \dots \wedge x_n$) within the given “context” (\mathcal{K}) – the premises and the conclusions together give an argument. Moving from classical to weak OR-refinement is straightforward. A set of instances of the same concept in CORE, x_1, x_2, \dots, x_n is a complete weak OR-refinement of the instance x of that same concept if and only if either of the x_1, x_2, \dots, x_n defeasibly derives x , i.e., $\mathcal{K}, x_1 \sqcup x_2 \sqcup \dots \sqcup x_n \vdash x$, where $a \sqcup b$ stands for *either a or b*.

To illustrate the interest of weak over strong refinement, we take an example from the classical Software Quality Characteristics Tree (Boehm et al., 1976), where it is stated that maintainability decomposes onto testability, understandability, and modifiability, so that we can – if we follow Boehm et al. – AND-refine a maintainability softgoal onto softgoals on testability, understandability, and modifiability. Having strong AND-refinement links in this case is rather counterintuitive, for it would mean that there is proof that satisfactory testability, understandability, and modifiability entail satisfactory maintainability. As there clearly is no such proof, it turns out that our weak refinement links provide a more realistic account of the matter. If we write that testability, understandability, and modifiability weakly AND-refine maintainability, we are saying: to believe that a system satisfies the softgoals defined over these three quality considerations leads to conclude that the softgoals defined over maintainability are satisfied. We thus have the satisficing of the softgoals over testability, understandability, and modifiability as premises to the conclusion that the softgoals on maintainability are satisfied. There is no proof (i.e., no provability relation) in weak AND-refinement; instead, we have an argument (as explained above), which means that we can provide additional arguments in support for that argument, or counterargue it. Our weak refinement links are thus in line with recent contributions in software quality research, where it is recognized that no absolute quality ontology can be suggested (such an ontology indicates how to always and completely decompose quality considerations such as maintainability), but that local ontologies are usually built for a given project, or a given domain (Dromey, 1995; Kitchenham & Pfleeger, 1996).

To respond to the second issue raised above (point 2), it is necessary to remove the constraint on the relationships between concrete instances. We move from AND-refinement and OR-refinement to a more general form, in which AND or OR are but some among various kinds of relationships that may exist between the concrete instances. To resolve at the same time the third issue raised earlier (point 3), we will allow any instance of any concept in CORE to be refined by any other instances of any concepts in CORE. The justified refinement (i.e., j-refine) relationship is defined as follows.

Definition 15 (Justified refinement). Let \mathbf{x} be an instance of a concept in CORE. There is a j-refine relationship from a non-empty set of instances $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ to an instance \mathbf{x} , written $\mathcal{T}_R(\mathbf{x}) \asymp \triangleright \mathbf{x}$, if and only if:

- (1) the root of $\mathcal{T}_R(\mathbf{x})$ defeasibly derives \mathbf{x} : $\mathcal{K}, \text{root}(\mathcal{T}_R(\mathbf{x})) \vdash \mathbf{x}$;
- (2) for any node $\text{nonrootnode}(\mathcal{T}_R(\mathbf{x}))$ in $\mathcal{T}_R(\mathbf{x})$ other than the root of $\mathcal{T}_R(\mathbf{x})$, the following holds: $\mathcal{K}, \text{nonrootnode}(\mathcal{T}_R(\mathbf{x})) \not\vdash \mathbf{x}$;
- (3) $\text{root}(\mathcal{T}_R(\mathbf{x}))$ is consistent: $\text{root}(\mathcal{T}_R(\mathbf{x})) \not\vdash \perp$;

where the justified refinement tree $\mathcal{T}_R(\mathbf{x})$ for \mathbf{x} is built by applying the following steps:

1. Set each \mathbf{x}_i in $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ as a leaf of $\mathcal{T}_R(\mathbf{x})$. Call a node without outgoing edges a “top node”.
2. For each top node \mathbf{x}_i that has a unary modifier U for \mathbf{x}_i , create a node $U\mathbf{x}_i$ and an edge directed from the node \mathbf{x}_i to the node $U\mathbf{x}_i$. Label that edge with the unary modifier U .

3. For each subset T of top nodes related by a single n -ary modifier N , create a node $N(T)$, and create from each member of T an edge directed to the node $N(T)$. Label each of these edges with the n -ary modifier N .
4. Repeat steps 2–3 until $\mathcal{T}_R(\mathbf{x})$ is a tree and no unary or n -ary modifiers need to be applied to the root of $\mathcal{T}_R(\mathbf{x})$.

The following remarks are in order:

- Standard AND- and OR-refinement relationships in RE are represented as trees, where the root is an instance considered less detailed, and the leaves are more detailed instances. We keep this representation in justified refinement by building the justified refinement tree $\mathcal{T}_R(\mathbf{x})$.
- Condition 1 in Definition 15 indicates that $\langle \text{root}(\mathcal{T}_R(\mathbf{x})), \mathbf{x} \rangle$ is an argument. In other words, if the condition given by the root of $\mathcal{T}_R(\mathbf{x})$ holds, then it is reasonable to believe that \mathbf{x} holds as well.
- Condition 2 ensures that there is no subtree of the justified refinement tree, such that its root defeasibly entails \mathbf{x} . This condition follows the minimality condition from AND- and OR-refinement, but is adapted to the terminology here.
- Condition 3 indicates that justified refinement cannot result in an inconsistent combination of concrete instances.
- Unary and n -ary modifiers in Definition 15 have the same meaning as those in the justified approximation relationship (see Definition 14 in Section 5.1). Conjunction and disjunction, as in AND- and OR-refinement amount to two kinds of n -ary modifiers in justified refinement: namely, binary modifiers AND and OR. A weak AND-refinement thus amounts to a justified refinement tree, in which the root is of the form $\text{AND}(T)$, where T is the set of instances that stand in conjunction.

To illustrate the use of justified refinement, consider the problem of defining a registration form for new users of an online flight booking application. Say that we are given the following statement:

If a user is not registered and selects to purchase a flight, then a clear and comprehensive registration form should be displayed. (Ex. 36)

We instantiate the Communicated information concept to capture the content of the above statement:

i₁: If a user is not registered and selects to purchase a flight, then a clear and comprehensive registration form should be displayed. (Ex. 37)

Following the definitions of the concepts in CORE, we obtain the following instances rather straightforwardly:

k₁: A user is not registered. (Ex. 38)

k₂: A user selects to purchase a flight. (Ex. 39)

g₁: Display a comprehensive registration form. (Ex. 40)

g₂: Display a clear registration form. (Ex. 41)

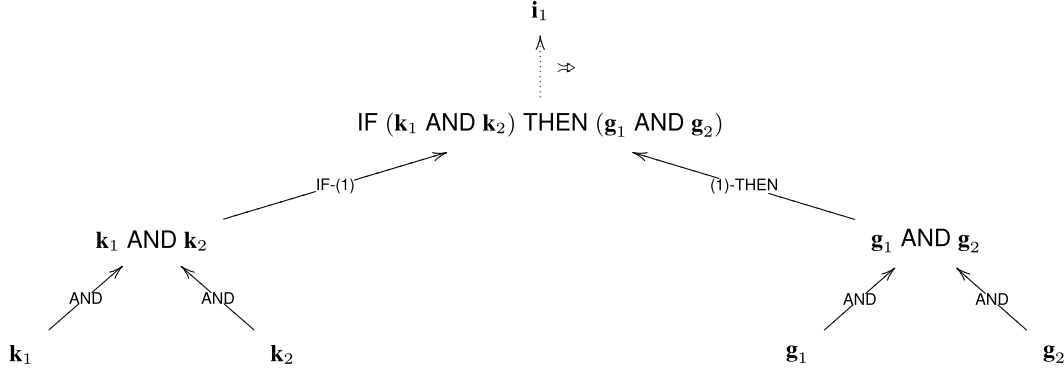


Fig. 6. Justified refinement of instance i_1 of Communicated information – see text for the content of the instances. $IF (k_1 \text{ AND } k_2) \text{ THEN } (g_1 \text{ AND } g_2)$ is the root of the justified refinement tree $\mathcal{T}_R(i_1)$. We connect with a dotted arrow the root of $IF (k_1 \text{ AND } k_2) \text{ THEN } (g_1 \text{ AND } g_2)$ to the abstract instance i_1 and label that link with the justified refinement relationship. Remark that the IF-THEN is broken down onto two parts, and that these parts are related by simply placing on IF the suffix identical to the prefix on THEN, namely “(1)”.

We apply the procedure in Definition 15 to build the justified refinement tree: we set k_1, k_2, g_1, g_2 as leaves of $\mathcal{T}_R(i_1)$ and add modifiers according to our understanding of i_1 . We get the structure shown in Fig. 6.

The refinement in Fig. 6 is justified if we provide no evidence to the contrary. Consider now the leaves of $\mathcal{T}_R(i_1)$, and proceed to justifiably refine the goals g_1 and g_2 . We can refine g_1 onto a functional goal and a softgoal:

fg_1 : Registration form is displayed to the user. (Ex. 42)

sg_1 : Registration form is comprehensive. (Ex. 43)

fg_1 and sg_1 are in conjunction, as shown in Fig. 7. The justification for this refinement is given in Fig. 8. We can justifiably refine g_2 as in Fig. 9, with the functional goal fg_1 and the following softgoal:

sg_2 : Registration form is clear. (Ex. 44)

Say that experience leads us to observe that a clear registration form is one in which each of the non-straightforward input fields carries a textual help displayed to the user. Moreover, the purpose of the registration form should be clearly explained. We can use that observation to justify $\mathcal{T}_R(sg_2) \succ \triangleright sg_2$, where $root(\mathcal{T}_R(sg_2)) = sg_3 \text{ AND } sg_4$:

sg_3 : The purpose and expected content of each input field is clearly explained. (Ex. 45)

sg_4 : The purpose of the registration form is clearly explained. (Ex. 46)

Provided that no counterargument is voiced against the said observation, $\mathcal{T}_R(sg_2) \succ \triangleright sg_2$ is a justified refinement.

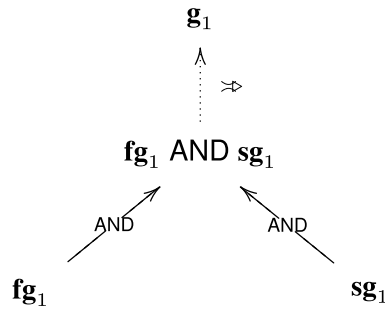


Fig. 7. Justified refinement of instance g_1 of Goal – see text for the content of the instances. Figure 8 gives the justification for this refinement.

g_1 : Display a comprehensive registration form.

$\triangleleft \succ fg_1 \text{ AND } sg_1$

$\leftarrow \leftarrow$ (Arg.1 | **U**) a softgoal refines g_1

$\leftarrow \leftarrow$ (Arg.1.1 | **U**) Comprehensiveness of a registration form is a quality for which we have no well-defined quality space.

$\leftarrow \leftarrow$ (Arg.1.2 | **U**) Comprehensiveness of a registration form is a quality for which the quality space varies among stakeholders (i.e., the quality space is subjective).

$\leftarrow \leftarrow$ (Arg.2 | **U**) a functional goal refines g_1

$\leftarrow \leftarrow$ (Arg.2.1 | **U**) Once we have made explicit the softgoal sg_1 , what remains in g_1 is that the registration form should be displayed. Displaying a registration form does not refer to qualities of the registration form.

Fig. 8. Arguments in favor of $\mathcal{T}_R(g_1) \triangleleft \succ g_1$.

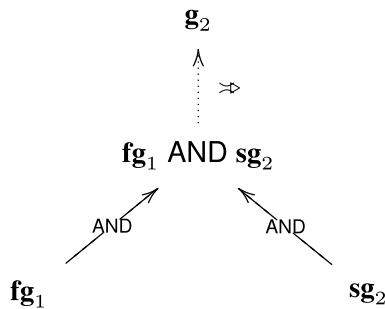


Fig. 9. Justified refinement of instance g_2 of Goal – see text for the content of the instances.

5.3. E-compare and evaluate

The *evaluate* relationship is introduced in order to relate Individual evaluation to the instance that it evaluates. There are no particular remarks to make for that relation, so that we move on to *e-compare*.

We have argued above (Definition 13) that an instance of Comparative evaluation compares in terms of favor or disfavor two or more instances of CORE concepts. The compared instances are related by the *e-compare* relationship. Two questions first need to be dealt with. First, why not rely on some form of

utility, and second, why not use some kind of preferences, such as, e.g., revealed preferences.

The central argument against utility at present, and from there to any form of quantitative interpersonal comparison of values of desirability measures, has been much discussed in welfare economics. Robbins summarizes it as follows (Robbins, 1981, p. 5):

... suppose that A and B fall into conversation about their respective enjoyments and A says to B, ‘Of course I get more satisfaction than you out of music’, and B vigorously asserts the contrary. Needless to say, you and I as outsiders can form our own judgments. But these are essentially subjective, not objectively ascertainable fact. There is no available way in which we can measure and compare the satisfactions which A and B derive from music. Intelligent talk? But that may be misleading. Facial expression? That too may be deceptive. Willingness to make sacrifices of other things? But that clearly shifts the emphasis to the satisfactions derived from other things; and we are left with the ultimate difficulty of interpersonal comparisons that, as Jevons put it, ‘Every mind is thus inscrutable to every other mind and no common denominator of feeling seems to be possible’. Jevons’ emphasis may be a bit extreme; we certainly think we know what other people are feeling, though opinions notoriously differ in different cultural settings and between different people. But it is surely incontestable where scientific proof or measurement is in question.

Once quantitative interpersonal comparison based on one form or another of utility is out of the way, classical (economic) preferences – such as those between commodity bundles (Varian, 1984) – may appear as a possible option. Arguments for discarding that option were voiced above (Section 4.5): evaluations can be expressed individually or comparatively over possibly anything, they have no internal consistence, are context dependent, and so on. Evaluations are thus less demanding. In being such, they are also considerably less convenient, for regularity and consequently prediction of choice seems to become impossible. Evaluation-based comparison herein certainly rejects internal consistency that is postulated for classical preferences. Doing so does not entail calling comparisons inconsistent: as Sen observed (Sen, 1993, p. 501) “inconsistency may be easily disputed, depending on the context, if we know a bit more about what the person is trying to do”. Here, that context are the voiced beliefs, desired, intentions, and evaluations of that person. If evaluations need to make sense to the software engineer, they will only do so within the said context, augmented by the beliefs, desires, intentions, and evaluations of the engineer, some of which are about what is being observed. To obtain an explicit account of this, a model could be conceived to relate one’s mental states to other’s mental states, and infinite regress would certainly appear, for it would seem relevant to ask what one believes about the other, what one believes that the other believes that one believes, and so on *ad infinitum*. Similarly, desires, intentions, and attitudes would become intertwined in the said chain, rendering the present discussion not only technically challenging but also removed from reality, if only because deliberation prior to choice operates on limited resources.

If consistency – in the sense of classical preference – is not assumed and the other’s mind remains inscrutable in a rigorous sense, then what remains is to rely on communicated evaluations as the proper comparison of alternative instances of concepts in CORE. “Real” attitudes, emotions, moods, of feelings that underlie evaluations may remain entirely hidden, but anything better seems hardly available without making significant concessions to questionable postulates. A very simple relation therefore arises from communicated evaluations that make explicit comparison.

Definition 16 (Evaluative comparison). There is an e-compare relationship between two non-empty sets X_i and X_j of instances of CORE concepts (X_i and X_j are disjoint) if and only if there is an instance

of Comparative evaluation that compares in terms of favor or disfavor all members of \mathbf{X}_i to all members of \mathbf{X}_j . We write $\mathbf{X}_i \mathcal{R}_{T(P, prop(\mathcal{R}))} \mathbf{X}_j$ the e-compare relationship between \mathbf{X}_i and \mathbf{X}_j , where:

- \mathcal{R} is the e-compare relationship, that obeys properties returned by the function $prop(\cdot)$;
- $T(P, prop(\mathcal{R}))$ is the dialectical tree that justifies the properties of \mathcal{R} given by $prop(\mathcal{R})$; and
- P in $T(P, prop(\mathcal{R}))$ is a premise to the conclusion that $prop(\mathcal{R})$ apply to \mathcal{R} .

It is important to observe that we have given up a unique concept of comparative evaluation in Definition 16. There can be as many comparative evaluation relationships as there are different combinations of properties thereof that we may encounter in a given development project. These properties are, for example, transitivity, symmetry, and/or completeness. In some cases, we may well have an evaluative comparison that obeys the standard axioms for economic preferences, but we do not confine evaluative comparison to that kind of comparison alone. Provided that there is a justification for some set of properties for a particular kind of evaluative comparison, it is admitted.

For illustration, consider the following instance of Communicated information:

\mathbf{i}_2 : If the user books a flight 30 or more days before the departure date, it is preferred that a paper ticket and an email booking confirmation be sent, than to send only an email booking confirmation. (Ex. 47)

We justifiably refine \mathbf{i}_2 with IF \mathbf{k}_3 THEN ((EITHER \mathbf{g}_3 OR \mathbf{g}_4) AND \mathbf{ec}_1), where:

\mathbf{k}_3 : User books a flight 30 or more days before the departure date. (Ex. 48)

\mathbf{g}_3 : Send a paper ticker and an email booking confirmation. (Ex. 49)

\mathbf{g}_4 : Send only an email booking confirmation. (Ex. 50)

\mathbf{ec}_1 : \mathbf{g}_3 STRICTLY-PREFERRED-TO \mathbf{g}_4 . (Ex. 51)

Observe that the justified refinement clearly indicates that two *alternative* goals are involved in the refinement (notice the binary modifier EITHER – OR), and that these two alternatives are evaluatively compared by \mathbf{ec}_1 . STRICTLY-PREFERRED-TO is a kind of e-compare relationship. Note finally that, according to Definition 16, if we claim that STRICTLY-PREFERRED-TO is transitive, we must justify that claim.

6. Requirements problem revisited

Zave and Jackson's formulation of the requirements problem states that, given \mathbf{R} and \mathbf{K} the aim of the engineer is to determine \mathbf{S} , such that $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$. Observe the following:

- The relation \vdash is a monotonic provability relation, so that:
 - \mathbf{K}, \mathbf{S} and \mathbf{R} must define a sound and complete theory;

- Any two distinct specifications \mathbf{S}_1 and \mathbf{S}_2 , such that $\mathbf{K}, \mathbf{S}_1 \vdash \mathbf{R}$ and $\mathbf{K}, \mathbf{S}_2 \vdash \mathbf{R}$ are equivalently desirable.
- Additional domain assumptions can never have an impact on the ability of the given plans to satisfy the given requirements, that is, for any $\mathbf{K}' \subseteq \mathbf{K}$, we always have $\mathbf{K}', \mathbf{S} \vdash \mathbf{R}$.
- The formulation $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$ is overly simplistic:
 - As we have argued throughout the paper, there are no concepts that capture the information captured in CORE by instances of Softgoal and Soft domain assumption. $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$ abstracts entirely from desires or beliefs over qualities with subjective quality spaces.
 - $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$ does not capture evaluations. There can be no ranking of desirability of alternative domain assumptions, requirements, or plans in $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$. As we have noted at the outset of the paper (Section 1), there is no notion of quality of a system when one adopts $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$: quality is either acceptable (when $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$) or unacceptable ($\mathbf{K}, \mathbf{S} \not\vdash \mathbf{R}$).

Before we proceed to reformulate the requirements problem below (Section 6.2), some preliminary remarks are highlighted (Section 6.1).

6.1. Methodological and technical preliminaries

6.1.1. Delimit a concrete subset of concepts

A subset of CORE concepts plays a role in the problem formulation. Examples mentioned in the preceding sections of the paper illustrated that Goal instances are more concrete than the Communicated information instances. We introduced the j-refine relationship to relate instances of the latter to those of the former. We did so in order to show how potentially complex combinations of speech acts given by stakeholders are broken down to arrive at a better understanding of the stakeholders' statements. The atomic parts that are thereby obtained are combined in justified refinement trees, so that the information on their combinations is not lost across levels of abstraction.

The first column of Table 2 lists typical instances of each CORE concept. The second column gives corresponding sets of instances, whereby each set carries *all* instances of the corresponding concepts that

Table 2

CORE concepts and corresponding symbols, sets thereof and subsets of the instances that play a role in the formulation of the requirements problem

Concepts in CORE (with symbols for their typical instances)	Set of all instances elicited in a given project	Subset of instances needed in the requirements problem formulation
i : Communicated information	I	
g : Goal	G	
fg : Functional goal	FG	$\mathbf{FG}^\uparrow \subseteq \mathbf{FG}$
qc : Quality constraint	QC	$\mathbf{QC}^\uparrow \subseteq \mathbf{QC}$
sg : Softgoal	SG	$\mathbf{SG}^\uparrow \subseteq \mathbf{SG}$
p : Plan	P	$\mathbf{P}^\uparrow \subseteq \mathbf{P}$
k : Domain assumption	K	
fk : Functional domain assumption	FK	$\mathbf{FK}^\uparrow \subseteq \mathbf{FK}$
qk : Quality domain assumption	QK	$\mathbf{QK}^\uparrow \subseteq \mathbf{QK}$
sk : Soft domain assumption	SK	$\mathbf{SK}^\uparrow \subseteq \mathbf{SK}$
ei : Individual evaluation	EI	$\mathbf{EI}^\uparrow \subseteq \mathbf{EI}$
ec : Comparative evaluation	EC	$\mathbf{EC}^\uparrow \subseteq \mathbf{EC}$

are elicited in a given development project. As we can have alternatives (e.g., alternative goals in \mathbf{G}), none of the sets needs to be consistent.

Now, we do not keep all of instances in all of these sets in the problem formulation. There is indeed no reason to keep those instances that are justifiably refined by other instances. We keep the latter, precisely because they are more concrete than the former. Moreover, we keep none of the instances of Communicated information, Goal, and Domain assumption. Any instance of Communicated information should be justifiably refined by instances of Goal, Plan, Domain assumption, and/or Evaluation; any instance of Goal should be justifiably refined by instances of Functional goal, Quality constraint, and/or Softgoal; and any instance of Domain assumption should be justifiably refined by instances of Functional domain assumption, Quality domain assumption, and/or Soft domain assumption.

The third column of Table 2 lists sets of instances that play a role in the formulation of the requirements problem. Following the preceding remarks, we denote \mathbf{FG}^\uparrow the set of functional goals that carries no instances \mathbf{fg} that are justifiably refined; evidently, we have that $\mathbf{FG}^\uparrow \subseteq \mathbf{FG}$. This same observation applies for \mathbf{QC}^\uparrow , \mathbf{FG}^\uparrow , \mathbf{P}^\uparrow , \mathbf{FK}^\uparrow , \mathbf{QK}^\uparrow , \mathbf{SK}^\uparrow , \mathbf{EI}^\uparrow and \mathbf{EC}^\uparrow .

6.1.2. Concretize instances

Combining justified refinement trees is critical, because it relates the information elicited over the course of RE. Without doing so, the software engineer would work with seemingly unrelated instances of various concepts, some of them associated to justified refinement trees, while others (namely, softgoals and soft domain constraints) with justified approximation trees. For instance, say that we have two distinct domain assumptions \mathbf{k}_i and \mathbf{k}_j , but no domain assumption that is more abstract, less detailed, and/or less precise than \mathbf{k}_i and \mathbf{k}_j . Then, how \mathbf{k}_i and \mathbf{k}_j relate? Are they in conjunction? Are they alternatives? Are they conditionally related? To know this, we require a more abstract \mathbf{k} , such that \mathbf{k}_i and \mathbf{k}_j are in the justified refinement tree of \mathbf{k} (i.e., \mathbf{k}_i and \mathbf{k}_j are leaves in $\mathcal{T}_R(\mathbf{k})$). The relationship between \mathbf{k}_i and \mathbf{k}_j is explicit in $root(\mathcal{T}_R(\mathbf{k}))$.

If we generalize the said issue, we observe that it is necessary to build and combine justified refinement and justified abstraction trees up to a point, at which the following are known:

- the relationship between all members of \mathbf{FG}^\uparrow ;
- the relationship between all members of \mathbf{QC}^\uparrow ;
- and so on for each set in the third column of Table 2.

It is straightforward to see that the relationship between all members of \mathbf{FG}^\uparrow is given by the root of the tree that justifiably refines the most abstract member of \mathbf{FG} . This most abstract functional goal – denote it \mathbf{fg}^\bullet – can be seen as the source of all other functional goals. In yet other terms, any more concrete functional goal can be obtained by successive justified refinement of the most abstract functional goal \mathbf{fg}^\bullet . Now, there is no guarantee that we will always be able to elicit \mathbf{fg}^\bullet from the stakeholders and then refine it to obtain more concrete ones. It is more likely that we will have variously concrete functional goals, which need to be related via justified refinement trees of more abstract functional goals, until \mathbf{fg}^\bullet is constructed. Regardless of how we get to the most abstract functional goal \mathbf{fg}^\bullet , we need it because we must determine the root of its justified refinement tree.

Not only do we require the root of the justified refinement tree for \mathbf{fg}^\bullet , but we need it to be written over all members of \mathbf{FG}^\uparrow . Consider hypothetical justified refinement trees shown in Fig. 10. It is clear that $root(\mathcal{T}_R(\mathbf{fg}_1^\bullet)) = (\text{EITHER } \mathbf{fg}_2 \text{ OR } \mathbf{fg}_3)$ does not give us the relationship between \mathbf{FG}^\uparrow . To obtain this, we need to apply the following procedure. Below, \mathbf{X}^\uparrow is any set from the third column in Table 2; \mathbf{X} is the corresponding set from the second column in Table 2.

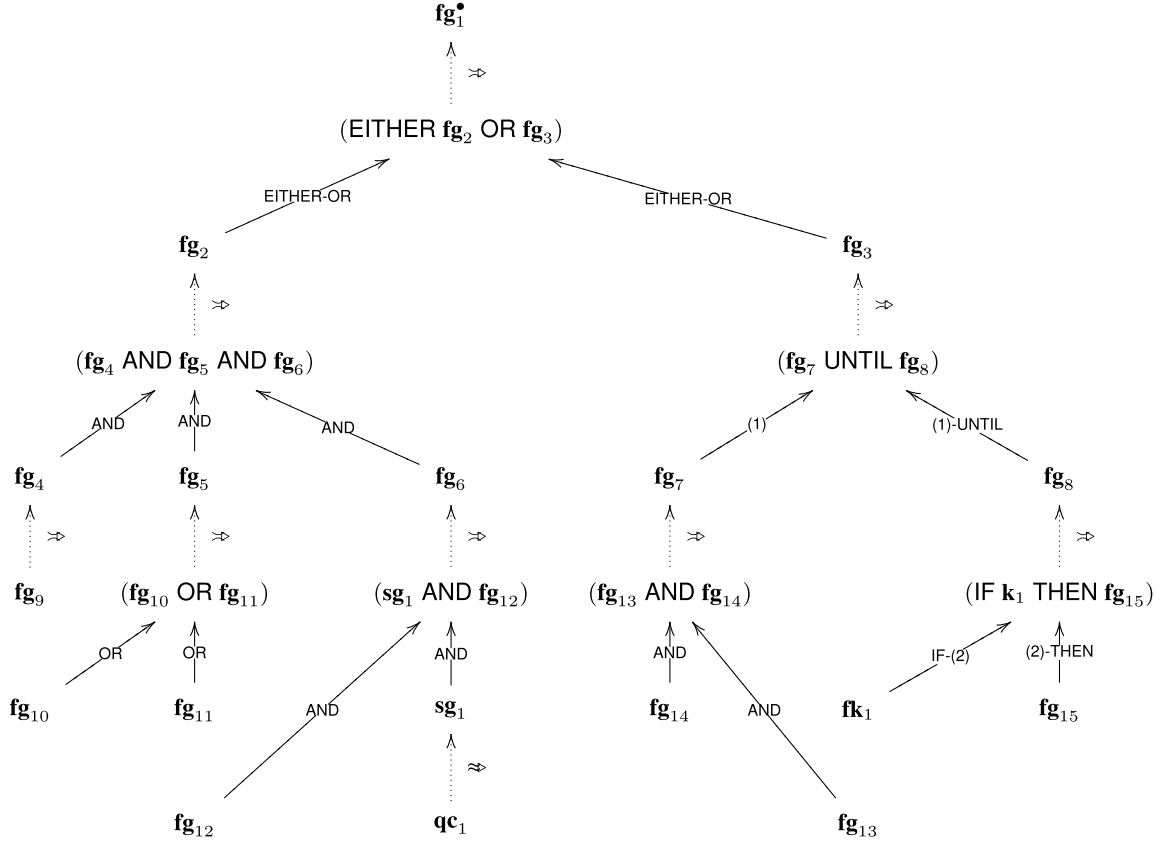


Fig. 10. Hypothetical example of connected justified refinement trees for the most abstract functional goal fg_1^* . As in Fig. 6, we use numbers in between parenthesis to relate parts of asymmetrical n-ary connectives: above, “(1)” is used for UNTIL and “(2)” for IF-THEN.

1. Justifiably refine instances in \mathbf{X} until there is a \mathbf{x}^* .
2. For each instance \mathbf{x} in \mathbf{X}^\dagger , such that there is $\mathcal{T}_R(\mathbf{x})$, replace \mathbf{x} with $root(\mathcal{T}_R(\mathbf{x}))$.

The output of this procedure is the rewritten $root(\mathcal{T}_R(\mathbf{x}^*))$, in which there are no members of $\mathbf{X} \setminus \mathbf{X}^\dagger$. In our hypothetical example, we have rewritten EITHER fg_2 OR fg_3 into:

$$\text{EITHER } (fg_9 \text{ AND } (fg_{10} \text{ OR } fg_{11})) \text{ AND } (sg_1 \text{ AND } fg_{12}) \\ \text{OR } ((fg_{13} \text{ AND } fg_{14}) \text{ UNTIL } (\text{IF } fk_1 \text{ THEN } fg_{15})).$$

Since softgoals and soft domain assumptions may appear in the rewritten $root(\mathcal{T}_R(\mathbf{x}^*))$, we further perform the following two steps:

1. Replace each sg in \mathbf{SG}^\dagger with $root(\mathcal{T}_A(sg))$.
2. Replace each sk in \mathbf{SK}^\dagger with $root(\mathcal{T}_A(sk))$.

We have thereby replaced sg_1 with qc_1 , and obtained the following:

$$\text{EITHER } (fg_9 \text{ AND } (fg_{10} \text{ OR } fg_{11})) \text{ AND } (qc_1 \text{ AND } fg_{12}) \\ \text{OR } ((fg_{13} \text{ AND } fg_{14}) \text{ UNTIL } (\text{IF } fk_1 \text{ THEN } fg_{15})).$$

Our formulation of the requirements problem uses $root(\mathcal{T}_R(\mathbf{x}^\bullet))$ that is rewritten according to the procedure above. As we are concerned only with the “concrete” concepts (Section 6.1.1), we will apply that procedure to the most abstract instances of the following concepts: Functional goal, Quality constraint, Softgoal, Plan, Functional domain assumption, Quality domain assumption, Soft domain assumption, Individual evaluation, and Comparative evaluation. These are the concepts for which we retain the most concrete instances in the third column in Table 2. We summarize the given operation as follows.

Definition 17 (Concretize). Let \mathbf{X} be the set of all elicited instances of a CORE concept. Assuming that:

- instances in \mathbf{X} are justifiably refined and combined until there is a $\mathbf{x}^\bullet \in \mathbf{X}$ to which all members in \mathbf{X} are related via combined justified refinement trees; and
- all instances in \mathbf{SG}^\uparrow are justifiably approximated; and
- all instances in \mathbf{SK}^\uparrow are justifiably approximated; and
- each \mathbf{sg} in \mathbf{SG}^\uparrow is replaced with $root(\mathcal{T}_A(\mathbf{sg}))$; and
- each \mathbf{sk} in \mathbf{SK}^\uparrow is replaced with $root(\mathcal{T}_A(\mathbf{sk}))$;

then for each instance \mathbf{x} in \mathbf{X} , such that there is $\mathcal{T}_R(\mathbf{x})$, replace \mathbf{x} with $root(\mathcal{T}_R(\mathbf{x}))$. We denote the output obtained via the concretize operation with $\mathcal{C}(\mathbf{X})$.

In our hypothetical example in Fig. 10, if we concretize \mathbf{FG} , we obtain:

$$\mathcal{C}(\mathbf{FG}) = \text{EITHER } (\mathbf{fg}_9 \text{ AND } (\mathbf{fg}_{10} \text{ OR } \mathbf{fg}_{11}) \text{ AND } (\mathbf{qc}_1 \text{ AND } \mathbf{fg}_{12})) \\ \text{OR } ((\mathbf{fg}_{13} \text{ AND } \mathbf{fg}_{14}) \text{ UNTIL } (\text{IF } \mathbf{fk}_1 \text{ THEN } \mathbf{fg}_{15}))$$

Of course, we assume in the example above that:

- $\mathbf{FG}^\uparrow = \{\mathbf{fg}_9, \mathbf{fg}_{10}, \mathbf{fg}_{11}, \mathbf{fg}_{12}, \mathbf{fg}_{13}, \mathbf{fg}_{14}, \mathbf{fg}_{15}\}$; and
- $\mathbf{qc}_1 \in \mathbf{QC}^\uparrow$; and
- $\mathbf{fk}_1 \in \mathbf{FK}^\uparrow$.

For another illustration of the concretize operation (Definition 17), consider again Figs 6, 7 and 9. We can straightforwardly combine these into Fig. 11. For the sake of example, we assume that we have identified some functional domain assumptions \mathbf{fk}_1 and \mathbf{fk}_2 that justifiably refine, respectively \mathbf{k}_1 and \mathbf{k}_2 . Further assume that \mathbf{fk}_1 , \mathbf{fk}_2 , \mathbf{fg}_1 , \mathbf{sg}_2 , and \mathbf{sg}_2 are not further refined. To concretize \mathbf{i}_1^\bullet , we perform the following steps on the tree in Fig. 11:

1. Replace:
 - (a) \mathbf{k}_1 with \mathbf{fk}_1 ; and
 - (b) \mathbf{k}_2 with \mathbf{fk}_2 ; and
 - (c) \mathbf{g}_1 with $(\mathbf{fg}_1 \text{ AND } \mathbf{sg}_1)$; and
 - (d) \mathbf{g}_2 with $(\mathbf{fg}_1 \text{ AND } \mathbf{sg}_2)$.
2. Replace:
 - (a) $(\mathbf{k}_1 \text{ AND } \mathbf{k}_2)$ with $(\mathbf{fk}_1 \text{ AND } \mathbf{fk}_2)$; and
 - (b) $(\mathbf{g}_1 \text{ AND } \mathbf{g}_2)$ with $((\mathbf{fg}_1 \text{ AND } \mathbf{sg}_1) \text{ AND } (\mathbf{fg}_1 \text{ AND } \mathbf{sg}_2))$.
3. Replace:

$$\text{IF } (\mathbf{k}_1 \text{ AND } \mathbf{k}_2) \text{ THEN } (\mathbf{g}_1 \text{ AND } \mathbf{g}_2)$$

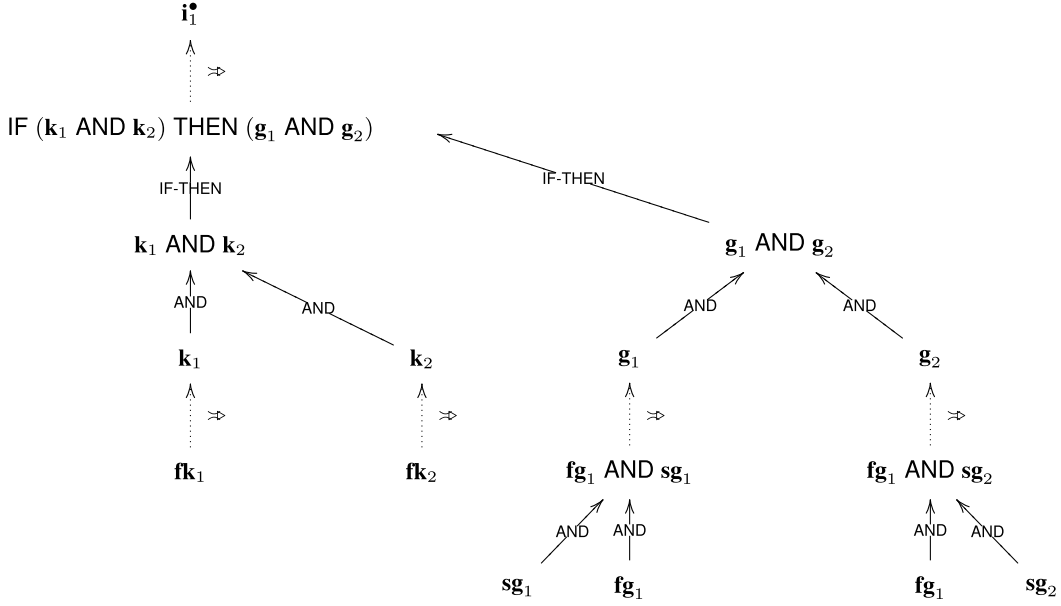


Fig. 11. Combination of justified refinements from Figs 6, 7 and 9.

with:

$$\text{IF } (\mathbf{fk}_1 \text{ AND } \mathbf{fk}_2) \text{ THEN } ((\mathbf{fg}_1 \text{ AND } \mathbf{sg}_1) \text{ AND } (\mathbf{fg}_1 \text{ AND } \mathbf{sg}_2)).$$

As a result, we obtain:

$$\mathcal{C}(\mathbf{I}) = \text{IF } (\mathbf{fk}_1 \text{ AND } \mathbf{fk}_2) \text{ THEN } ((\mathbf{fg}_1 \text{ AND } \mathbf{sg}_1) \text{ AND } (\mathbf{fg}_1 \text{ AND } \mathbf{sg}_2)).$$

In this simplified example, we end up replacing initially communicated information \mathbf{i}_1^* by $\text{root}(\mathcal{T}_R(\mathbf{i}_1^*))$. More importantly, any instance that appears in $\text{root}(\mathcal{T}_R(\mathbf{i}_1^*))$ is replaced by the most concrete instances obtained by successive justified refinement over the course of the RE project. Instances elicited over the course of RE are justifiably refined until the most concrete instances reflect the content of atomic speech acts and are sufficiently clear precise for the purposes of the software engineer. In that case, $\text{root}(\mathcal{T}_R(\mathbf{i}_1^*))$ is rewritten with only the members of \mathbf{FG}^\uparrow , \mathbf{QC}^\uparrow , \mathbf{SG}^\uparrow , \mathbf{P}^\uparrow , \mathbf{FK}^\uparrow , \mathbf{QK}^\uparrow , \mathbf{SK}^\uparrow , \mathbf{EI}^\uparrow and \mathbf{EC}^\uparrow . These sets indeed contain instances that are the most concrete ones, that is, that have not been justifiably refined any further (Section 6.1.1).

6.2. New formulation of the requirements problem

There are two important intuitive ideas behind our problem formulation:

1. Over the course of the RE phase of a development project, the software engineer should identify a set of plans \mathbf{P}^* . If the system-to-be and the stakeholders act in accordance to the concretization $\mathcal{C}(\mathbf{P}^*)$ of \mathbf{P}^* , then the conditions described by $\mathcal{C}(\mathbf{FG}^C)$ and $\mathcal{C}(\mathbf{QC}^C)$ should be brought about, and this without violating the conditions described by $\mathcal{C}(\mathbf{FK}^C)$ and $\mathcal{C}(\mathbf{QK}^C)$. Intuitively, acting along the plans in the setting described by the functional and quality domain assumptions, will lead to the satisfaction of the functional goals at the desired levels of quality constraints.

2. If the software engineer takes the first set of plans that meets the conditions in point 1 above, the engineer is *satisficing*. To see why, observe that $\mathcal{C}(\mathbf{FG}^C)$ describes alternative desired and compulsory conditions, and that stakeholders are not equally satisfied with all of the alternatives. Indeed, they communicated individual and comparative evaluations. Intuitively then, we need to rank alternative sets of plans that satisfy the conditions in point 1 above. Our problem formulation ranks the potential plans on the basis of evaluations. The aim is to move from satisficing to the selection of the best plans among those that are feasible and have been identified over the course of RE.

Consider hypothetical examples to clarify the second point above. Say that there are two alternative functional goals, say \mathbf{fg}_1 and \mathbf{fg}_2 , and that some instance \mathbf{ec} of Comparative evaluation states that it is strictly more desirable to satisfy \mathbf{fg}_1 than \mathbf{fg}_2 . Let A be a sentence, such that \mathbf{fg}_1 and \mathbf{fg}_2 are its subsentences. Intuitively, what \mathbf{ec} does is that it orders models of A with respect to the degree of favor that stakeholders communicated via expressives. In the case of \mathbf{fg}_1 and \mathbf{fg}_2 , any model \mathcal{M} of A , in which \mathbf{fg}_1 is true and \mathbf{fg}_2 is false, is strictly more preferred according to \mathbf{ec} than any other model \mathcal{N} of A , in which \mathbf{fg}_1 is false and \mathbf{fg}_2 is true.

Consider a different case. Say that \mathbf{fg}_3 and \mathbf{fg}_4 are not alternative functional goals. Instead, there is an instance \mathbf{ei} of Individual evaluation, which indicates that \mathbf{fg}_3 is compulsory and \mathbf{fg}_4 is optional. Again, let A be a sentence, such that \mathbf{fg}_3 and \mathbf{fg}_4 are its subsentences. What \mathbf{ei} does is that it orders models of A as follows. Any model \mathcal{M} of A , in which \mathbf{fg}_3 and \mathbf{fg}_4 are both true is strictly more preferred according to \mathbf{ei} than any other model \mathcal{N} of A , in which \mathbf{fg}_3 is true and \mathbf{fg}_4 is false.

In more rigorous terms, these ideas work out as follows.

Definition 18. Let \mathcal{M} and \mathcal{N} be models of the sentence A . Let ω be an order over alternative complete truth assignments to sentences. We say that \mathcal{M} ω -dominates \mathcal{N} , writing $\mathcal{N} \leq_{\omega} \mathcal{M}$, if:

1. \mathcal{M} and \mathcal{N} have the same domain, and
2. the truth assignment of P in \mathcal{M} ranks at least as high on ω as the truth assignment of P in \mathcal{N} .

Definition 19. Say that we explicitly consider only a subset of possible models of A . All other models of A remain unknown to us. A model \mathcal{M} of A is called ω -suitable if any other explicitly considered model \mathcal{M}' is such that:

$$\mathcal{M} \leq_{\omega} \mathcal{M}' \text{ only if } \mathcal{M}' = \mathcal{M}.$$

Definition 20. We say that A suitably entails q with respect to ω , written $A \models_{\omega} q$, provided that q is true in all models of A that are ω -suitable.

Definition 21 (Requirements problem). Given $\mathcal{C}(\mathbf{FG})$, $\mathcal{C}(\mathbf{QC})$, $\mathcal{C}(\mathbf{SG})$, $\mathcal{C}(\mathbf{FK})$, $\mathcal{C}(\mathbf{QK})$, $\mathcal{C}(\mathbf{SK})$, $\mathcal{C}(\mathbf{EI})$ and $\mathcal{C}(\mathbf{EC})$, find a set of plans \mathbf{P}^* such that:

$$\mathcal{C}(\mathbf{FK}), \mathcal{C}(\mathbf{QK}), \mathcal{C}(\mathbf{P}^*) \models_{\mathcal{C}(\mathbf{EI})} \mathcal{C}(\mathbf{FG}), \mathcal{C}(\mathbf{QC}) \quad \text{and}$$

$$\mathcal{C}(\mathbf{FK}), \mathcal{C}(\mathbf{QK}), \mathcal{C}(\mathbf{P}^*) \models_{\mathcal{C}(\mathbf{EC})} \mathcal{C}(\mathbf{FG}), \mathcal{C}(\mathbf{QC}).$$

Above, “suitable” is intended to highlight that we seek the optimum not among all possible alternative solutions, but only among those that are explicitly considered. The aim is to point out the fact that the software engineer operates under limited resources, and cannot consider all possible alternatives.

Informally, \mathbf{P}^* is the set of plans consistent with concretizations of functional and quality domain assumptions (i.e., $\mathcal{C}(\mathbf{FK})$ and $\mathcal{C}(\mathbf{QK})$), whereby models of $\mathcal{C}(\mathbf{FK})$, $\mathcal{C}(\mathbf{QK})$, $\mathcal{C}(\mathbf{P}^*)$ are models of functional goals and quality constraints (i.e., $\mathcal{C}(\mathbf{FG})$ and $\mathcal{C}(\mathbf{QC})$). Models of $\mathcal{C}(\mathbf{FK})$, $\mathcal{C}(\mathbf{QK})$, $\mathcal{C}(\mathbf{P}^*)$ are suitable with respect to orders established by concretizations of individual evaluations (i.e., $\mathcal{C}(\mathbf{EI})$) and comparative evaluations (i.e., $\mathcal{C}(\mathbf{EC})$). In other words, we select a set of plans that (i) does not violate functional and quality domain assumptions, (ii) ensures that functional goals and quality constraints verify, and (iii) ensure that there is no other set of plans that does both (i) and (ii) and ranks higher in both the order given by individual evaluations and the order given by comparative evaluation. Intuitively, the order given by individual evaluations indicates that more of the optional functional goals and/or quality constraints a set of plans that satisfies, the more is that set of plans desirable. The order given by comparative evaluations states that a set of plans that satisfies a combination of functional goals and quality constraints that rank *higher* over the concretization of comparative evaluations is more desirable than a set of plans that satisfies a combination of functional goals and quality constraints that ranks *lower* over the concretization of comparative evaluations.

7. Illustrated walkthrough

Up to this point, we used various examples to illustrate the concepts in CORE. The aim at present is to consider a case study and from there on provide a CORE walkthrough that is not interrupted by definitions and accompanying discussions. The reader will note that we adopt no specific notation herein, but remain in line with the way the examples were presented above. Notations that may be appropriate for the presentation of instances of CORE concepts are not the topic at present; neither are the methodological issues that arise in realizing case studies such as the one below. We return to these considerations briefly in the conclusion of the paper (Section 9).

The case study comes from the development of a complex web-based software system that opened to the public in mid-July 2008. Our knowledge of the example comes from the first author's extensive involvement in the entire development process, with particular focus on the early steps of the requirements engineering phase, when the scope of the application was delimited, the main functionalities defined, and the user interface prototypes built.

The application in question is a social network organized around audio content, called *b92.fm* (henceforth, FM).⁸ The impetus for the development of FM came from Radio-Television B92, an influential broadcasting corporation in South East Europe. The company has extensive experience in the creation and distribution of news and entertainment content online; its main website drew above 300,000 unique visitors per month during 2008 according to Google Analytics.⁹ The following is an excerpt pieced together from internal documents and emails sent between main actors involved in the very first discussions of what FM might be and how it would relate to the current offerings of the company.

Radio B92, the first component of the B92 [broadcasting] Network, started up in the early nineties as an important catalyst of democratic change and values in a then isolated, war-torn and politically oppressed society. In 1998, Radio B92 received due recognition for its societal role when the MTV honored it with the "Free Your Mind" award for promoting tolerance and human rights. The aim with FM is to pursue these values by strengthening its presence online in a completely new format that

⁸The application is accessible at the following Internet address: <http://www.b92.fm/>.

⁹<http://www.google.com/analytics/>.

will facilitate and deepen the interaction between the Radio and its audience. The core idea of FM is to enable users to create their own playlists consisting of a variety of B92 radio shows and music content. They can then listen to these playlists anytime and exchange them with other users of FM. In addition to the currently broadcast shows, we would post online the audio archives of the most popular shows from the '90s when the Radio acquired its cult status. In addition to in-house produced audio content, users will be able to search through and play the content from a large audio database consisting of all music that B92 broadcasts. We are already aware of the royalties and constraints that record labels and the artists' representatives impose, and the business model still remains relevant. Users will be able to create their own playlists. Each playlist can be seen as a radio station, because any number of other users can listen to it. The creation of playlists is as simple as adding items at the Amazon.com to your shopping cart. The user simply needs to browse the website and add favorite songs, music genre tags, or shows. Such selections can be saved as playlists. All users of b92.fm will be able to exchange playlists. Audio will be streamed to each individual user; it cannot be downloaded.

Any user will be able to upload their audio creations. The system will rank the similarity of users' tastes, so as to suggest "neighbors" and thereby facilitate the creation of groups with similar interests. The users will be able to aggregate their other multimedia content which they already published on various online platforms like Youtube.com, Flickr.com and others.

We instantiate Communicated information to encompass the excerpt above in its entirety:

i₁: Radio B92, the first component of the B92 [broadcasting] Network... (Ex. 52)

We then proceed to justifiably refine it, which gives us the structure shown in Fig. 12. Instances shown in Figure 12 are the following:

i₂: Users should have the freedom to express themselves on FM, if their expression is not harmful to the values of tolerance and mutual respect. (Ex. 53)

g₁: FM will deepen the interaction between Radio B92 and its audience. (Ex. 54)

g₂: Users can create their own playlists consisting of a variety of B92 radio shows and music content. (Ex. 55)

g₃: Users can listen to playlists anytime and exchange them with other users of FM. (Ex. 56)

g₄: All current shows broadcast on Radio B92 should be posted on FM. (Ex. 57)

g₅: Users can search through and play audio content from B92's music database. (Ex. 58)

k₁: FM must not violate the constraints that record labels and the artists' representatives impose. (Ex. 59)

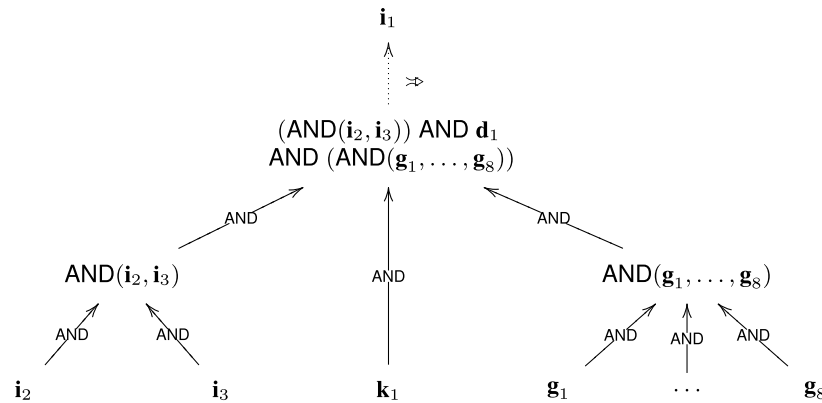


Fig. 12. Justified refinement of instance i_1 of Communicated information – see text for details.

- i_3 : The creation of playlists should be as simple as adding items at Amazon.com to the shopping cart. The user simply needs to browse the website and add favorite songs, music genre tags, or shows. (Ex. 60)
- g_6 : Any user will be able to upload their audio creations. (Ex. 61)
- g_7 : The system will rank the similarity of users' tastes, so as to suggest "neighbors" and thereby facilitate the creation of groups with similar interests. (Ex. 62)
- g_8 : The users will be able to aggregate their other multimedia content which they already published on various online platforms like Youtube.com, Flickr.com and others. (Ex. 63)

The justified refinement of i_1 is a simple separation of the various parts of i_1 , ending in a refinement tree with n-ary AND modifiers. We omit the justification for this refinement, for the move from i_1 to the instances in Fig. 12 is straightforward. We turn to i_2 , and refine it as shown in Fig. 13.

- d_2 : User-generated content on FM is in accordance with usual standard of appropriate behavior. (Ex. 64)
- g_9 : Users will have the freedom to express themselves on FM. (Ex. 65)

Note that i_4 is actually less precise on the criteria of appropriate behavior than i_2 . The counterarguments in Fig. 14 that let us keep i_4 as it is.

We can then continue to increase the detail of i_2 , by refining k_2 and g_9 . The justified refinement for k_2 is shown in Fig. 15, and the justification is shown in Fig. 16.

- sk_1 : Users' discussions on FM are in accordance with usual standard of appropriate behavior. (Ex. 66)
- sk_2 : Users' comments on FM are in accordance with usual standard of appropriate behavior. (Ex. 67)

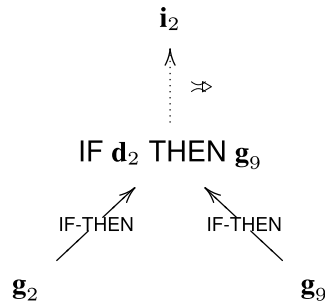
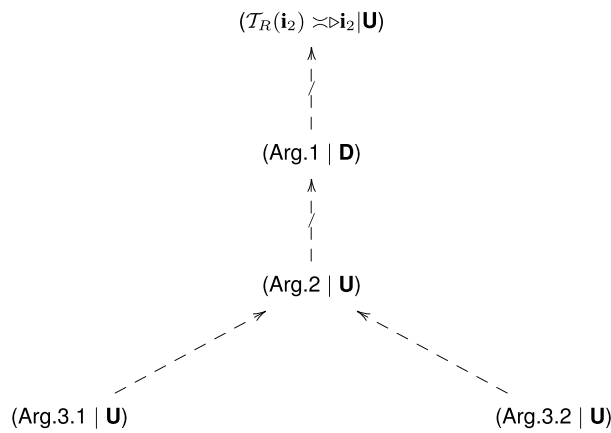


Fig. 13. Justified refinement of instance i_2 of Communicated information – see text for details.



Arguments used in the dialectical tree above:

- (Arg.1) k_2 is less precise on the criteria of appropriate behavior than i_2 .
- (Arg.2) Given that moderation practices for FM will be the same as for B92's news website, and that these practices proved adequate in the last decade, we can leave the details on moderation practices imprecise here.
- (Arg.3.1) The criteria indicated in i_2 for appropriate behavior are not precisely defined in documentation. They may not be the only criteria to use in evaluating appropriate behavior.
- (Arg.3.2) Informal rules used in moderating content on B92's news website developed over the last decade. While they are not formalized, very few user complaints have been recorded, and in the relevant cases, moderation practices were adjusted in response.

Fig. 14. Dialectical tree for the justified refinement in Fig. 13.

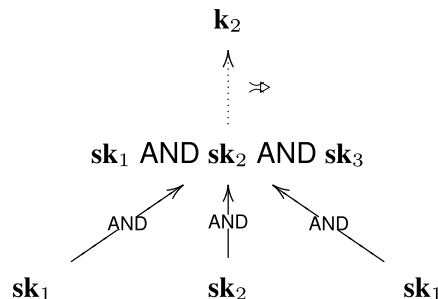
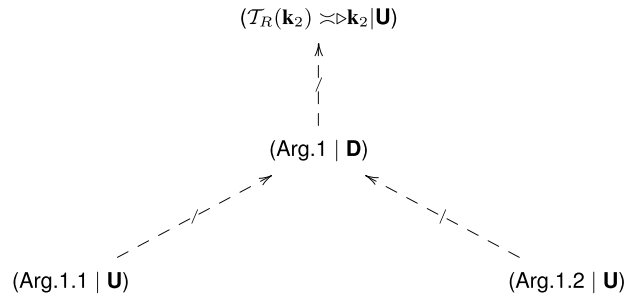


Fig. 15. Justified refinement of instance k_2 of Communicated information – see text for details.



Arguments used in the dialectical tree above:

- (Arg.1) Photos and videos are user-generated content that is also displayed on FM, and therefore needs to be moderated.
 - (Arg.1.1) Photos are not hosted on FM and cannot be uploaded on FM by users. All photos are drawn from external websites (e.g., Flickr, Picasaweb), where the photos are already moderated.
 - (Arg.1.2) Videos are not hosted on FM and cannot be uploaded on FM by users. All videos are drawn from external websites (e.g., Youtube, Vimeo), where the videos are already moderated.
-

Fig. 16. Dialectical tree for the justified refinement in Fig. 15.

sk₃: The audio content uploaded by users on FM is in accordance with usual standard of appropriate behavior. (Ex. 68)

We now justifiably approximate **sk₁**. From the arguments in the justification of $\mathcal{T}_R(\mathbf{i}_1) \asymp \triangleright \mathbf{i}_1$, we need not approximate **sk₁** by looking into the intended meaning(s) of the quality “appropriate behavior”. Instead, we looked at the effects that users’ inappropriate behavior has. Experience from the moderation of discussion boards at the news website indicates that one or both of the following tend to occur when inappropriate behavior occurs and is not contained by the moderators: (i) there is a formal protest letter that arrives to the company, and/or (ii) the debate heats up over one message, so that the number of replies to that message increases significantly in proportion to the total number of messages in the given discussion. There are no more than $X_{\mathbf{qk}_1}$ formal complaints per year on user-generated content on FM.

qk₁: There are no more than $Y_{\mathbf{qk}_2}$ formal complaints per year on user-generated content on FM. (Ex. 69)

qk₂: $X_{\mathbf{qk}_1} > Y_{\mathbf{qk}_2}$. (Ex. 70)

fk₂: $X_{\mathbf{qk}_1} > Y_{\mathbf{qk}_2}$. (Ex. 71)

ec₁: **qk₂** STRICTLY-PREFERRED-TO **qk₁**. (Ex. 72)

qk₂: At any time, at most 15% of new discussions are not verified by a moderator at B92. (Ex. 73)

qk₂: At least 15% of discussions are moderated by users by the end of the first 3 months. (Ex. 74)

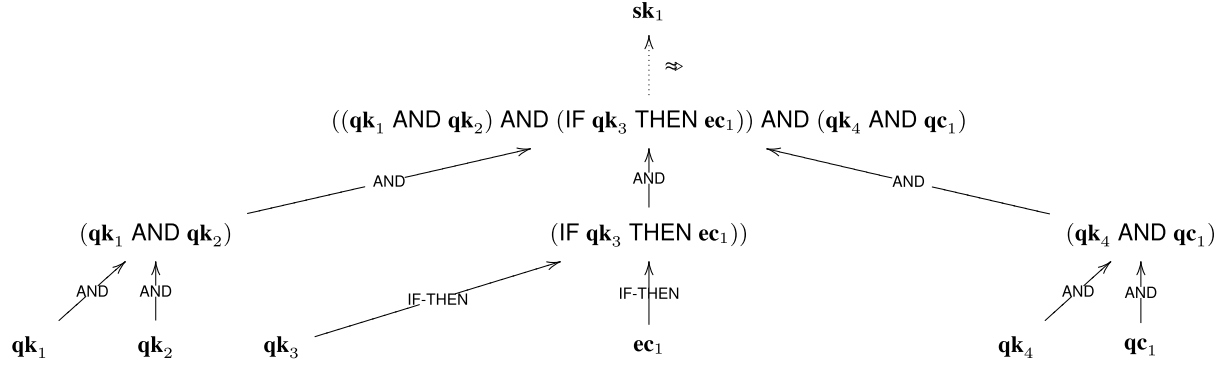
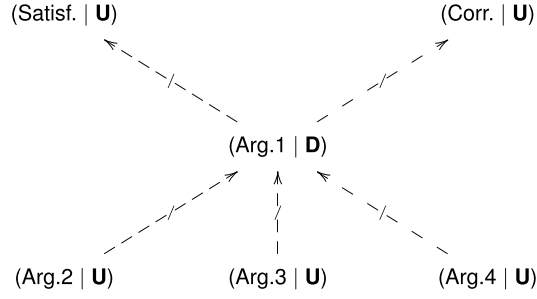


Fig. 17. Justified refinement of instance sk_1 of Communicated information – see text for details.



- (Satisf.) Satisfying $root(\mathcal{T}_A(sk_1))$ satisfies sk_1 .
- (Corr.) Values in the quality space of each quality referred to in $\mathcal{T}_A(sk_1)$ correlate sufficiently with values in the quality space of the quality referred to in sk_1 .
- (Arg.1) The meaning intended for the quality “appropriate behavior” is not made precise in the justified refinement tree $\mathcal{T}_A(sk_1)$.
- (Arg.2) Experience from the B92 news website indicates that the frequency of formal protests for inappropriate messages that escaped moderation and the proportion of replies to a single message are good indicators of questionable content in discussions.
- (Arg.3) The less formal complaints on discussion content, the more likely that the content is appropriate.
- (Arg.4) The more users take a moderating role, the less questionable content will escape moderation.

Fig. 18. Dialectical tree for the justified refinement in Fig. 15.

Recall that we need to provide two justifications when justifiably approximating softgoals and soft domain assumptions. Both are shown in Fig. 18.

The leaves of the justified refinement and approximation trees identified to this point need to be further systematically subjected to refinement and approximation. This is performed up to the point where (i) the software engineer is satisfied with the level of detail, and (ii) the stakeholders agree on the concretization of individual and comparative evaluations. Both issues fall outside of the scope of the present paper, and are to be dealt with via methodologies for RE. We can nevertheless make the following observations:

- The appropriateness of a level of detail naturally brings us to the question of mathematical formalization of requirements, that is, the use of formal methods in RE. It is usually said in RE that *early*

phases can be distinguished from the *late* RE phases (e.g., Castro et al., 2002). The distinction is methodological, and is rooted in that the information used at the early phase is expressed informally. In contrast, late RE involves reasoning on information formulated in a mathematical logic. Formal methods thus play an important role at late RE. Our approach fits this methodological perspective. Systematic construction of justified refinement and approximation trees will bring the software engineer to the point where it will be straightforward to formalize individual leaves of these trees. What needs to be done at that point is to map the informal unary and binary modifiers used in these trees to those formally defined in the mathematical logic of choice. For example, the modifier AND would map to logical conjunction \wedge , OR to logical disjunction \vee , the conditional IF-THEN might map to material implication \rightarrow , and so on. Concretization (Definition 17) would then amount to combine logical formulas found at the leaves of the refinement and approximation trees.

- It is evident that stakeholders do not agree by default with each other about the individual and comparative evaluations. More precisely, their individual and/or comparative evaluations may be conflicting. This is a methodological problem beyond the scope of the present discussion. Means to facilitate negotiation (e.g., Boehm et al., 1998 or Keeney & Raiffa, 1976 and subsequent) are thereby necessary for the successful resolution of the requirements problem.

8. Related work

In the preceding sections of this paper we have contrasted the details of our proposal to the principal lines of related work. We have thus discussed in detail the deficiencies of Zave and Jackson's ontology and problem formulation, and our responses thereto. We have contrasted our j-approximate and j-refine relationships to, respectively classical contribution and refinement relationships used in RE. The aim at present is to take a more broader perspective on related work. First, we recall some key definitions of the *requirement* concept (Section 8.1), primarily in order to illustrate their variety and draw broad parallels and departures from the corresponding concepts in CORE. We then do the same for the *goal* concept in RE. We close the discussion of related work by contrasting Zave and Jackson's ontology to CORE (Section 8.3) on considerations that have not already received attention in the sections above.

8.1. Requirement concept in requirements engineering research

It does not require considerable effort to see that the RE field richly and variously defines the *requirement* concept. For instance, Ross & Schoman's (1977) early definition of RE points out that requirements describe the purpose of a system:

Requirements definition must say why a system is needed, based on current or foreseen conditions, which may be internal operations or an external market. It must say what system features will serve and satisfy this context. And it must say how the system is to be constructed.

In a survey of research efforts in RE, Zave (1997) elaborates:

Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families.

Letier & van Lamsweerde (2004) concur while placing emphasis on the critical role of system goals:

Requirements engineering is concerned with the identification of the goals to be achieved by the system-to-be, the operationalization of such goals into specifications of services and constraints, and the assignment of responsibilities for such services and constraints among human, physical, and software components forming the system.

A different terminology is used in the IEEE Guide to the Software Engineering Body of Knowledge (2004) and Kotonya and Sommerville's textbook on RE (Kotonya & Sommerville, 2000):

[Requirements engineering] is concerned with the elicitation, analysis, specification, and validation of software requirements. [...] Software requirements express the needs and constraints placed on a software product that contribute to the solution of some real-world problem.

The above agrees with Goguen & Linde (1993) that requirements express needs:

A basic question in Requirements Engineering is how to find out what users really need.

To go beyond needs, one might consider the IEEE 610 standard on Software Engineering Terms (1991) and read:

[A requirement is]: (1) a condition or capability needed by a user to solve a problem or achieve an objective; (2) a condition or capability that must be met or possessed by a system component to satisfy a contract, standard, specification or other formally imposed documents; (3) a documented representation of a condition or capability as in (1) or (2).

The *requirement* concept has evidently been variously defined as a purpose, a need, a goal, a function(ality), a constraint, a behavior, a service, a condition, or a capability. Surveying more definitions would give same or other interpretations, but the point remains the same: the *requirement* concept is richly interpreted and variously used. The problem here lies not in the need for rich conceptual foundations for RE, but in the fact that there actually is no agreement on conceptual foundations on which the various efforts can be compared and benefits of their accumulation and combination reaped. Definitions cited above are hardly synonymous.

It is with Zave and Jackson's proposal of the "requirements problem" formulation that the field moved towards a clearer understanding of the *requirement* concept. Their following observation is key with regards to the problem of arriving at a definition of the *requirement* concept (Zave & Jackson, 1997, p. 7):

The primary distinction necessary for requirements engineering is captured by two grammatical moods. Statements in the 'indicative' mood describe the environment as it is in the absence of the machine or regardless of the actions of the machine; these statements are often called 'assumptions' or 'domain knowledge'. Statements in the 'optative' mood describe the environment as we would like it to be and as we hope it will be when the machine is connected to the environment. Optative statements are commonly called 'requirements'.

This observation is important because it states that information relevant for the software engineer during RE is about the beliefs about the problematic situation in which the development project is initiated, and which will hold after the system is deployed, *and* about what stakeholders desire be the case after system development and deployment. The critical idea is that a fundamental distinction between domain assumptions and requirements is traced to grammatical moods and thereby to propositional attitudes (i.e., beliefs, desires, and intentions). It is not that their explicit distinction between requirements

and domain assumptions was truly new, but that the separation between these two can be grounded in grammatical moods observed in communication. While one could argue that hints of these ideas appear already in Ross & Schoman's (1977) definition (i.e., requirements say "why a system is needed"), Zave and Jackson add more.

It is precisely Zave and Jackson's idea to trace requirements back to grammatical moods that led us to look into the relationship between grammatical moods and RE concepts. But therein lies the problem. The issue in question results from two facts: (i) optative and indicative moods are two among a dozen or so grammatical moods that are commonly identified in, for instance, natural language processing research (e.g., McShane et al., 2004); and (ii) it is in general challenging to arrive at an exhaustive list of grammatical moods, as grammatical moods are specific to each natural language (e.g., Jespersen, 1963 and later). Our solution is to turn instead to speech acts. This brought two key benefits. First, to turn to speech acts instead of grammatical mood is to classify communicated information on bases that are independent of the natural language in which RE is performed. This is not to say that Searle's classification of speech acts is not debated, as discussed recently by Zaefferer (2006). It is, however, widely accepted and a serious contender for its spot is absent at the time of writing. Second, speech acts led us to delimit the scope of CORE to propositional attitudes and evaluations. Not only did this give us a solid criterion to delimit the scope of CORE, but also led us to highlight the important role played by emotions, attitudes, moods, and feelings.

8.2. Goal concept in requirements engineering research

Since the 1970s, system development frameworks include some form of analysis involving *goals* (van Lamsweerde, 2001). The *goal* concept has become an essential part of any system's documentation through standards (e.g., IEEE-Std-830/1993). Just as for the *requirement* concept (Section 8.1), there is no shared definition of the *goal* concept.

Table 3 lists informal definitions of the *goal* concept appearing in various RE frameworks where the *goal* concept plays a central role (i.e., goal-oriented RE frameworks). KAOS highlights the nonoperational nature of *goals*, pointing to the need for taking action to make goals precise by refinement. Broadly speaking, the KAOS definition is in line with those of Tropos, *i*^{*}, GDC and Lightswitch: a goal designates desirable conditions on the system and/or its environment. Such conditions restrict the set of alternative system and environment states. A different conceptualization appears in the NFR framework, where goals are employed for representing nonfunctional requirements, in addition to design decisions, and arguments for or against other goals. We can interpret "design decisions" as restricting potential desired system and environment states. Concepts of argument and justification appear in NFR and GBRAM.

The definition of the *goal* concept in CORE departs from the conceptualizations cited above primarily in that (i) we place our Goal concept within a wider ontology, (ii) we ground our definition in the communication between the software engineer and the stakeholders, (iii) we precisely relate our Goal concept to derived concepts, that is, Functional goal, Quality constraint, and Softgoal. Both Goal and Functional goal, Quality constraint, and Softgoal follow the very general idea that is present in all of the frameworks cited in Table 3: goals capture what is desired.

8.3. From Zave and Jackson's ontology to core ontology for requirements engineering

Below, we first cover some final remarks on the more specific differences between CORE and Zave and Jackson's ontology (Section 8.3.1). We then argue that the core ontology pays particular attention

Table 3

Informal definitions of the *goal* concept in RE frameworks where the *goal* concept plays a central role (i.e., goal-oriented RE frameworks)

Framework	Informal definition of the <i>goal</i> (and derived) concepts
KAOS	<p>“A goal is a nonoperational objective to be achieved by the composite system. Nonoperational means that the objective is not formulated in terms of objects and actions available to some agent in the system; in other words, a goal as it is formulated cannot be established through appropriate state transitions under control of one of the agents”. (Dardenne et al., 1993)</p> <p>“A goal is a desired property about quantities in the environment”. (Letier, 2001)</p>
Tropos and i^*	<p>“A goal is a condition or state of affairs in the world that the stakeholders would like to achieve”. (Castro et al., 2002; Yu, 1997)</p>
NFR	<p>“Goals [refer to] non-functional requirements, design decisions and arguments in support or against other goals.” (Chung et al., 1999; Mylopoulos et al., 1992)</p>
REF	<p>“According to the nature of a goal, a distinction is made between hard goals and soft goals. A goal is classified as hard when its achievement criterion is sharply defined [...]. For a soft goal, instead, it is up to the goal originator, or to an agreement between the involved agents, to decide when the goal is considered to have been achieved [...]. In comparison to hard goals, soft goals can be highly subjective and strictly related to a particular context; they enable the analysts to highlight quality issues [...] from the outset [...]”. (Donzelli, 2004)</p>
GDC	<p>“An enterprise goal is a desired state of affairs that needs to be attained”. (Kavakli, 1999)</p>
GBRAM	<p>“Goals are high level objectives of the business, organization or system. They capture the reasons why a system is needed and guide decisions at various levels within the enterprise”. (Anton, 1996)</p>
Lightswitch	<p>“[A] maintenance goal is said to represent a condition that remains constant. [...] [An] achievement goal has definite pre and post-conditions. The pre-condition represents the interpretation that the state of affairs has drifted (or will drift) outside of the threshold associated with the norm [i.e., a variable of the system whose state the system attempts to maintain unchanged as defined by an observer]. The post condition is an interpretation that is within this threshold”. (Regev & Wegmann, 2005)</p>

to “quality” (in the broad sense of the word; i.e., as in service quality, or product quality) in contrast to Zave and Jackson’s and comparable endeavors (Section 8.3.2). Finally, we study the consequences of the introduction of evaluations in the core ontology for requirements, and advance the idea that satisficing is not appropriate to characterize that approach to the resolution of the requirements problem (Section 8.3.3).

8.3.1. Final detailed remarks

The concepts in CORE arose from the realistic assumption that RE involves the use of information that the software engineer acquires via communication with the stakeholders of the system-to-be. Concepts were suggested in order to cover the basic mental attitudes (beliefs, desires, and intentions) and evaluations on propositional attitudes (arising from attitudes, emotions, feelings or moods) that are conveyed via speech acts. If what should be covered by conceptualizations for RE is that which can be communicated and relates to mental states of the stakeholders, then CORE is acceptable as a core for conceptualizations in RE.

Comparison of CORE and Zave and Jackson’s ontology reveals superficial correspondences. The Goal concept in CORE appears to correspond to Zave and Jackson’s *requirement*; our Domain assumption parallels their *domain assumption*; our Plan captures what they call *specification*. Correspondences are revealed, however, as superficial once the concepts derived from Goal and Domain assumption are more thoroughly studied.

Concepts that might correspond to Softgoal and Soft domain assumption are entirely absent from Zave and Jackson's ontology. This absence is due to Zave and Jackson's formulation of the requirements problem. As we observed in the introduction of this paper (Section 1), satisfaction in $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$ is binary: \mathbf{K} and \mathbf{S} cannot satisfy \mathbf{R} to some extent only. Softgoal and Soft domain assumption reflect precisely the very real possibility that there are finer than binary degrees of satisfaction. Indeed, when facing two different specifications that both satisfy the same requirements, the software engineer can choose either according to Zave and Jackson – both are equally good. In contrast, these quality-related concepts in CORE allow more nuance in the comparison of alternative specifications.

Another nuance should be highlighted. Namely, while Softgoal and Soft domain assumption are entirely absent from Zave and Jackson's ontology, the same cannot be categorically claimed for Quality constraint and Quality domain assumption. That is, Quality constraint and Quality domain assumption are absent, but simpler variants thereof *could* exist in Zave and Jackson's ontology as specializations of their *requirement* and *domain assumption* concepts. To simplify this discussion, consider Quality constraint only, whereby the same applies *mutatis mutandis* to Quality domain assumption. If we define a quality constraint $q_i = 500$ ms then the formulation $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$ remains relevant: the answer on binary satisfaction can be given for any specification with regards to $q_i = 500$ ms – either $q_i = 500$ ms and \mathbf{S} is acceptable, or $q_i \neq 500$ ms and \mathbf{S} is not acceptable. But say that we define $q_i \leq 500$ ms. While we can still answer the binary satisfaction question in that case with $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$, we lose the nuance that lower times are better than higher times among the values below the threshold $q_i = 500$ ms. This is essentially due to the implicit evaluations that hide behind instances of Quality constraint and Softgoal, and Quality domain constraint and Soft domain assumption. We discussed these above (Section 5.1).

8.3.2. Quality-orientation

CORE allowed us to highlight the importance of partial satisfaction of softgoals, namely through their specific relationship to quality constraints, to deal with idealistic and pessimistic expectations, and to account for preferences. The proposed definitions for goal, quality constraint, and softgoal concepts have several advantages. Together, these concepts cover the key taxonomy of functional and nonfunctional requirements. In contrast to previous work, it is clear which of the nonfunctional requirements are verifiable. All three concepts are grounded in an intuitive foundational ontology. It has not been particularly clear how measures (or qualities in general) fit within the goal and softgoal separation in RE. This issue is resolved herein. We acknowledge through quality constraints the basic tenet that what cannot be measured cannot be managed. However, we also make it clear that not all can be directly defined or measured. Convenience, security, safety, along with considerations such as fun amount to softgoals, which involve subjective and local quality spaces that are difficult to precisely describe and share. Our definitions are not specific to particular kinds of functional and nonfunctional requirements, but span much of ongoing software engineering research on these topics. We define the softgoal concept in a way that states precisely what a softgoal is within the communication between the engineers and stakeholders, and this without breaking off from the tradition of how softgoals are used in RE (established in Mylopoulos et al., 1992 and later). Recently, Glinz suggested that a “nonfunctional requirement is an attribute of or a constraint on a system” (Glinz, 2007), though it remains unclear what concepts the terms ‘attribute’ and ‘constraint’ denote. Our conceptualization has the advantages of being grounded in the actual context of RE (i.e., in the communication between the engineer and the stakeholders) and we are explicit on the meaning of our concepts.

A major departure of CORE from Zave and Jackson's ontology is our attention to the notion of “quality”, as it is usually understood in management science, software engineering, and related literature. In

software engineering, quality modeling research focuses on how to appropriately refer to the relevant information on, and reason about the quality of software. Early on, Boehm et al. (1976) defined quality of software in terms of a collection of distinct, interrelated, and interdependent quality attributes, including general utility, decomposed onto as-is utility, maintainability, and portability, which are subsequently decomposed onto, e.g., reliability, efficiency, and so on. Effort on obtaining precise and widely applicable definitions of software quality ensued. Quality models grounded in early research have been standardized (see, e.g., ISO 9126 International Organization for Standardization, 2001). Following observations that the choice of quality attributes and of their hierarchy may appear arbitrary, and that there are limited means for measuring characteristics at the hierarchies' top (Kitchenham & Pfleeger, 1996), such "top-down" approaches to quality modeling currently remain of interest mainly as catalogs of quality attributes (e.g., OMG, 2005). The catalogs are now applied usually in combination with a "bottom-up" approach (Dromey, 1995), whereby measurable software properties are linked to quality attributes these properties affect. In parallel with developments in quality models focused on quantitative evaluation of quality, qualitative models have also been proposed: the Nonfunctional Requirements approach (Mylopoulos et al., 1992) allows both top-down and bottom-up quality modeling, while evaluating quality through a qualitative reasoning procedure. Variations in quality definitions are not confined to software engineering. Confusion appears inherited from quality management research in management science that has been carried over to software engineering over the last three decades (see, e.g., Fox & Frakes, 1997 for a discussion).¹⁰ An overview of the various definitions (Reeves & Bednar, 1994) concludes that an agreed upon definition is elusive.

The principal aim of representing and reasoning about quality is to facilitate the construction and running of systems capable of meeting and exceeding user's expectations. The quantitative, metric-based approach to quality enables a characterization of the system in terms of its measurable properties and behaviors. Within such a perspective, the software engineer can argue that the system achieves some levels of quality according to the engineer's quality conceptualization or one adopted from international standards. Such a perspective is acknowledged and clearly incorporated within our core ontology through the notion of quality constraints. This, however, gives no guarantee that the system's users will share the same perception on the system's quality. Indeed, it is acknowledged in management (e.g., Oliver, 1977; Parasuraman et al., 1988; Parasuraman & Zeithaml, 2006; Reeves & Bednar, 1994) and software engineering research (e.g., Kitchenham & Pfleeger, 1996) that quality is a subjective experience: evaluations of quality will vary among users and will not necessarily correspond to evaluations based on metrics over measurable properties and behaviors of the system. It is through softgoals and evaluations that we accommodate subjective perception and evaluation of quality. Both the measurement perspective on quality, and the subjective one are absent from Zave and Jackson's ontology. No concepts are available in their ontology to cover the information that the engineer and/or the stakeholders may express about quality in either of these perspectives.

8.3.3. *From satisficing to suitability*

We have noted earlier that Simon (1955) introduced the notion of satisficing in opposition to optimization. We also mentioned that Mylopoulos et al. (1992) subsequently adopted that term in relation to their softgoal concept.

¹⁰One notable example of the transfer from management science to software engineering is the Capability Maturity Model (CMM) from the Software Engineering Institute at Carnegie Mellon University (1995). The CMM is grounded in Deming's (1982) and Juran's (1951) contributions.

We can readily discard optimization in RE, since it is unrealistic. The engineer cannot optimize because this would assume, among others, the knowledge of all possible alternatives. That is clearly impossible. Simon's satisficing stands in stark contrast to optimization, and it recognizes its inadequacy to convey the behavior of a rational decision maker. Given resource boundedness, the decision maker – here, the engineer – will adopt the first alternative that is above the threshold (i.e., acceptance) levels. Now, recall that the prescribed aim of the engineer is to build a system that the stakeholders will evaluate as being of high quality. Although this is clearly not optimization, still it is not Simon's satisficing. Indeed, once we know that it is important to rank above-threshold alternatives according to evaluations, we fall outside of the behavior that Simon described. While no considered alternative is (or rather, can be proven) optimal, it is not desirable to choose the first considered above-threshold alternative. Instead, it is hoped that several above-threshold alternatives will be defined over the course of RE, and that the one ranking the highest on the decision criteria (given by the softgoals and evaluations) should be adopted.

9. Conclusions and future work

What is the problem that the software engineer is trying to solve during the RE phase of software development?

The answer seems simple enough. *If* the engineer knows what the stakeholders want, and *if* she knows the feasible potential ways of satisfying the former, the problem which remains is purely one of selection and description of the chosen solution. Zave and Jackson's major contribution has been to work out the precise conditions that any selected way of satisfying stakeholders' needs must fulfill. Given what the stakeholders need (**R**), and the operating conditions of the system-to-be (**K**), the solution is **S**, such that $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$.

Obviously, none of the ifs above verifies in reality. Research and practice of RE clearly shows that it requires considerable skill, effort, and guidance to arrive at a point where most of the important requirements are known, and only a few alternative ways of satisfying them are identified. Even worse, this primarily methodological problem is not the only one that both research and practice are facing.

The software engineer elicits information relevant to the RE effort from what is conveyed by the stakeholders, or via other means, such as documentation of legacy systems. If we consider what can be conveyed, we see that there is more both in scope and in depth than Zave and Jackson's formulation captures. In scope, because desires and beliefs go hand in hand with expressions of emotions, attitudes, feelings, and moods. In depth, because there are fundamentally different kinds of needs, each requiring very different methodological treatment.

There is nothing revolutionary in arguing that $\mathbf{K}, \mathbf{S} \vdash \mathbf{R}$ is simplistic and idealistic. But there have been no alternatives that even remotely attempt to accommodate what is in fact systematically observed – namely, the more extensive scope and depth of information relevant to any RE effort.

Our first key premise in the present paper has been that it is more useful for the RE field to avoid conceptual idealism and elegance, in favor of a more realistic view of the RE effort. Our second critical premise is that the scope of the core ontology for requirements should be determined by the basic kinds of communicative actions that stakeholders can use in order to advance their desires, beliefs, intentions, and emotions, attitudes, feelings, and moods. Our final key premise is that the core ontology should be as deep as is necessary to cover all concepts that have fundamental conceptual differences, and thereby command very different methodological treatments.

The result is that we have shown Zave and Jackson's ontology to be incomplete. It does not cover all types of basic concerns – namely, the beliefs, desires, intentions, and evaluations – that the stakeholders

communicate. This makes Zave and Jackson's formulation of the requirements problem removed from the reality of research and practice. In response, we provide CORE, an ontology that covers all types of possible basic concerns and rests on sound conceptual foundations defined by an explicit foundational ontology. The new core ontology for requirements leads to the new formulation of the requirements problem that avoids the pitfalls of Zave and Jackson's formulation. The result are new standards for what minimum information should be represented in RE languages and new criteria for determining whether RE has been successfully completed.

We have, however, gone further than merely extending the scope and depth of Zave and Jackson's proposal. Firstly, in intimately tying a core ontology for requirements to the dictum and modus in communication, we have effectively moved to assumptions that differ from those found in related work. When identifying core concepts and relationships for RE, related work relies mainly on the experience of RE and various lower-level software engineering conceptualizations. CORE is in contrast grounded in communication, be it verbal or otherwise, through which requirements elicitation takes place. CORE thereby directly relates the bases of the requirements elicitation process to the concepts and relationships instantiated to capture what is communicated and how it is communicated. We argue that this is a very strong departure. What related work does is that it defines concepts and relationships rather independently of how the instances of these same concepts and relationships are obtained in reality. The engineer is given the elicited content on one hand, and some concepts and relationships on the other; the engineer then needs to see what bits and pieces of elicited content instantiate which concepts and relationships. It is effectively the engineer's choice if something is a goal, a plan, or otherwise. This is very different in CORE, since it is the very act of communicating something during elicitation that fully determines the classification of the communicated content. With CORE, it is not the engineer who is choosing the classification of the communicated content. It is instead the stakeholder who determines via the modus and the dictum if the communicated content is a goal, softgoal, or otherwise. Secondly, we argued that the satisfaction of requirements is defeasible. This is again very different from Zave and Jackson's proposal, where proof is sought that given domain assumptions and a specification together satisfy requirements. We have shown that defeasible reasoning permeates CORE. Not only is satisfaction defeasible, but the approximation and refinement relationships are also defeasible. It is important to understand that our reformulation of the requirements problem is not merely the result of a different ontology. The reformulation is very different from Zave and Jackson's formulation. Moving from monotonic to nonmonotonic reasoning is not merely a technical change. It is much more significant, for it changes the understanding of the RE problem. Satisfaction as with "+" does not reflect appropriately the problem that the engineer is to resolve. We argue that it is instead feasible optimization " $\models_{\mathcal{C}(\cdot)}$ " that is sought. In other words, there is no definite satisfaction and it is hardly possible to arrive at the best set of plans, or even to select the "best" among them. We deal with approximations and few alternative feasible solutions, and we can at best properly justify the choice of one solution over others. In summary, RE and our problem formulation change rather significantly the understanding of the requirements problem and its resolution.

CORE and the new formulation will hopefully raise important new questions in requirements and software engineering. Four categories of such questions immediately come to mind:

1. *Ontological foundations.* While we have provided definitions of the concepts in CORE, a step further can be taken. Namely, formal mathematical definitions and associated axioms for CORE are needed in relation to the current formalization of DOLCE. The overall aim is to obtain a characterization of CORE that is as precise as feasible. In doing so, we hope to move towards what Herre &

Heller (2006) call a complete system of local ontological mappings, which would cover all components of CORE.

2. *Conceptual extensions.* CORE carries no concepts that can capture the social aspects of the environment of the system-to-be. Concepts of organization, actor, role, and so on, along with relationships between these should be explored, so that an appropriate “social” layer can be added on top of CORE.
3. *Knowledge representation.* Usable representation of knowledge that is acquired over the course of the RE effort plays a critical role in successfully completing the software development project. Knowledge representation formalisms and specification languages that can capture the instances of CORE should be explored.
4. *Automated reasoning.* A key question concerns the kind of automated reasoning that needs to be performed in order to verify the so-called suitability of a set of plans. Advances in AI will certainly prove helpful in this respect. Questions of scalability of automated reasoning and its use within wider RE methodologies will also need to be addressed. These are not new questions in RE. However, the added expressivity of CORE requires us not only to draw from past research, but poses new challenges.
5. *Methodological support.* Established frameworks for RE lack capabilities needed for the organization of the RE effort when CORE is used. It is for instance unclear at present how to deal efficiently with the justification of approximation and refinement in methodological terms, or of how to elicit and analyze evaluations.

A final important remark to close the present discussion is that CORE is not an ontology specific to software. While this work did arise from the theoretical and practical problems in requirements engineering, which is currently strongly tied to software, there is in fact nothing in CORE or the requirements problem formulation that makes these contributions restricted to software. This is due to the fact that CORE is grounded in communication, and that we imposed no restrictions on the topic of communication. We expect specialization to happen as CORE becomes used together with various software engineering frameworks. It is therefore also relevant to consider if and how CORE fits in other fields, in which decisions are taken only after opinions and demands of potentially many stakeholders are elicited and analyzed. Encouraging results in this respect could lead to the conclusion that many topics in current and predominantly *software* requirements engineering are in fact issues to be addressed within a broader context of a software-independent field of requirements engineering or requirements science.

Acknowledgments

This is a significantly revised and expanded presentation of the ideas that appeared separately and in preliminary form in the *Proceedings of the 16th International Requirements Engineering Conference (RE'08)* (Jureta et al., 2008), the *Proceedings of the 1st International Workshop on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM'07)* (Jureta et al., 2007), and the *Proceedings of the 25th International Conference on Conceptual Modeling (ER'06)* (Jureta et al., 2006). The RE'08 paper (Jureta et al., 2008) was awarded the best paper award at the conference. John presented some of the ideas in his keynote at the *1st International Workshop on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM'07)* (Mylopoulos et al., 2007). We thank the participants of these events for their remarks and observations. We are particularly grateful to the editors and reviewers at Applied Ontology, who provided detailed and relevant indications on how to further clarify the relationship be-

tween the concepts in CORE and DOLCE. Some of this work was done while Ivan was visiting Carnegie Mellon University and the University of Trento. We gratefully acknowledge support from the universities of Trento, Toronto, Namur and Carnegie Mellon, along with Collège Interuniversitaire pour les Sciences du Management (CIM), and Fonds de la Recherche Scientifique – FNRS, both in Belgium. We thank Dejan Restak, Stevan Koprivica, and Vladimir Marić for allowing to use select information from the B92.fm project.

Appendix

A. Glossary of notation

K	the domain assumptions (in Zave and Jackson’s ontology).
S	the specification (in Zave and Jackson’s ontology).
R	the requirements (in Zave and Jackson’s ontology).
ϕ, ψ, \dots	some content (dictum) of a speech act.
i ϕ or i	some instance of Communicated information.
g ϕ or g	some instance of Goal.
fg ϕ or fg	some instance of Functional goal.
qc ϕ or qc	some instance of Quality constraint.
sg ϕ or sg	some instance of Softgoal.
p ϕ or p	some instance of Plan
k ϕ or k	some instance of Domain assumption.
fk ϕ or fk	some instance of Functional domain assumption.
qk ϕ or qk	some instance of Quality domain assumption.
sk ϕ or sk	some instance of Soft domain assumption.
e ϕ or e	some instance of Evaluation.
ei ϕ or ei	some instance of Individual evaluation.
ec ϕ or ec	some instance of Comparative evaluation.
I, I^C, I^O	Set of all instances of Communicated information, and its compulsory and optional partitions.
G, G^C, G^O	set of all instances of Goal, and its compulsory and optional partitions.
FG, FG^C, FG^O	set of all instances of Functional goal, and its compulsory and optional partitions.
QC, QC^C, QC^O	set of all instances of Quality constraint, and its compulsory and optional partitions.
SG, SG^C, SG^O	set of all instances of Softgoal, and its compulsory and optional partitions.
P, P^C, P^O	set of all instances of Plan, and its compulsory and optional partitions.
K, K^C, K^O	set of all instances of Domain assumption, and its compulsory and optional partitions.
FK, FK^C, FK^O	set of all instances of Functional domain assumption, and its compulsory and optional partitions.
QK, QK^C, QK^O	set of all instances of Quality domain assumption, and its compulsory and optional partitions.
SK, SK^C, SK^O	set of all instances of Soft domain assumption, and its compulsory and optional partitions.

E, E^C, E^O	set of all instances of Evaluation, and its compulsory and optional partitions.
EI, EI^C, EI^O	set of all instances of Individual evaluation, and its compulsory and optional partitions.
EC, EC^C, EC^O	set of all instances of Comparative evaluation, and its compulsory and optional partitions.
\mathcal{K}	indefeasible knowledge (in argumentation).
Δ	defeasible rules (in argumentation).
Δ^\downarrow	ground instances of Δ (in argumentation).
$p \dashrightarrow q$	some defeasible rule, read “ p is a reason for q ” (in argumentation).
$\approx \triangleright$	j-approximate relationship, also called the “justified approximation” relationship.
$\mathcal{T}_A(\cdot)$	justified approximation tree.
$root(\cdot)$	function, when applied to a tree it returns the root of the tree.
\cup	some unary modifier.
\mathbb{N}	some n-ary modifier.
$\succ \triangleright$	j-refine relationship, also called the “justified refinement” relationship.
$\mathcal{T}_R(\cdot)$	justified refinement tree.
\mathbf{FG}^\uparrow	set of the most concrete instances of Functional goal.
\mathbf{QC}^\uparrow	set of the most concrete instances of Quality constraint.
\mathbf{SG}^\uparrow	set of the most concrete instances of Softgoal.
\mathbf{P}^\uparrow	set of the most concrete instances of Plan.
\mathbf{FK}^\uparrow	set of the most concrete instances of Functional domain assumption.
\mathbf{QK}^\uparrow	set of the most concrete instances of Quality domain assumption.
\mathbf{SK}^\uparrow	set of the most concrete instances of Soft domain assumption.
\mathbf{EI}^\uparrow	set of the most concrete instances of Individual evaluation.
\mathbf{EC}^\uparrow	set of the most concrete instances of Comparative evaluation.
$\mathbf{fg}^\bullet, \mathbf{qc}^\bullet, \dots$	the most abstract instance of, respectively, Functional goal, Quality constraint, etc.
$\mathfrak{C}(\cdot)$	the concretize function applicable to a set of instances (e.g., \mathbf{FG}) of a single concept in CORE.
$\mathcal{M}, \mathcal{N}, \dots$	Models.
\models_ω	ω -suitable entailment.

B. Additional axioms for the intentional selection of Non-Agentive Social Objects

As the Artifact is created by the association of a Quality to a Non-Agentive Social Object, the latter must not change during the lifetime of the Artifact:

$$\begin{aligned}
 & (\text{IntentionalSels}(e, a, x, o, q) \wedge \text{NASO}(o) \wedge \text{PRE}(x, t)) \\
 & \rightarrow (K(o, x, t) \wedge \forall z(\neg z = o \rightarrow \neg K(z, x, t))). \tag{SART-A1}
 \end{aligned}$$

Above, $PRE(x, t)$ stands for x being present at time t . As long as o exists, it constitutes the Artifact x so that the artifact continues to exist:

$$(SART(x) \wedge K(o, x, t) \wedge PRE(o, t') \wedge t < t') \rightarrow K(o, x, t'). \quad (SART-A2)$$

The attributed capacity must be unique (SART-A3) and its quale cannot change in time (SART-A4):

$$(IntentionalSelS(e, a, x, o, q) \wedge IntentionalSelS(e', a', x, o', q')) \rightarrow q = q'. \quad (SART-A3)$$

$$(IntentionalSelS(e, a, x, o, q) \wedge ql(v, q, t) \wedge ql(v', q, t')) \rightarrow v = v'. \quad (SART-A4)$$

For a given intentional selection event, the selector agent(s) and the artifact must be unique (SART-A5); and the attributed capacities are only qualities of artifacts (SART-A6):

$$(IntentionalSelS(e, a, x, o, q) \wedge IntentionalSelS(e, a', x', o', q')) \\ \rightarrow (x = x' \wedge a = a'). \quad (SART-A5)$$

$$(AttributedCap(q) \wedge qt(q, x)) \rightarrow SART(x). \quad (SART-A6)$$

Following Borgo & Vieu's (2009) axiomatization of physical artifacts, it is necessary to ensure that capacities and attributed capacities map to qualia in the same space of capacities. The unary predicate CR is introduced, such that $CR(v)$ states that the quale v is a region in the capacity space.

$$(Capacity(q) \wedge ql(v, q, t)) \rightarrow CR(v). \quad (SART-A7)$$

It is further needed to allow the attributed capacity to have a set of capacity regions for a quale. To this aim, Borgo and Vieu use the predicate IN , whereby $IN(z, v)$ states that z is a member of v .

$$(AttributedCap(q) \wedge ql(v, q, t) \wedge IN(z, v)) \rightarrow CR(z). \quad (SART-A8)$$

There must be a creation event for each social artifact:

$$SART(x) \rightarrow \exists e \text{ CreationEv}(e, x), \quad (SART-A9)$$

where $CreationEv$ is defined as follows:

$$CreationEv(e, x) =_{def} \exists a, o, q \text{ IntentionalSelS}(e, a, x, o, q) \\ \wedge \exists t (qt_T(t, e) \wedge \forall t' (t' < t \rightarrow \neg(x, t'))). \quad (SART-D1)$$

C. Basics of argumentation

We recall here some basics of an approach to defeasible reasoning based on the notion of *argument*, and some notational conventions used throughout the paper. The material presented in this section is in line with and strongly draws from established ideas in philosophy, artificial intelligence, and defeasible logics.

As Koons summarizes, “reasoning is defeasible when the corresponding argument is rationally compelling but not deductively valid” (Koons, 2005). The reason we are interested in defeasible reasoning here is that it is nonmonotonic. If a set of premises logically entails a consequence, any superset of those premises also entails the same consequence – that is, deductive consequence is monotonic. In contrast, nonmonotonic consequence allows conclusions to be “undone” by new information. While a set of premises may defeasibly entail a conclusion, additional information may not allow the same conclusion to be drawn. We are thus involved in hypothetical reasoning, which fits the kind of reasoning of a software engineer looking to specify an appropriate system-to-be.

Among the various forms of defeasible reasoning (e.g., analogical reasoning, scientific induction) we are interested in logic-based approaches, and in particular on those that rely on the notion of *argument*. Arguments are *prima facie* proofs, in which a piece of information is (a defeasible) reason for another; in other words, believing that the former verifies is reason enough to believe that the later verifies as well. Support for the conclusion is thus given but is not established once and for all – it is important to consider counterarguments when drawing defeasible conclusions. The interest in *argument*-based defeasible reasoning herein arises from the intuitiveness of the notion of argument, and the possibility for the informal yet systematic argument-based defeasible reasoning in addition to a formal one. This caters to the clear need for systematic reasoning during the early phases of RE, when information of various degree of precision and formality is available and should all be accounted for in decision-making.

An *argument structure* is defined recursively as follows:

1. Any information of the form $P \text{ therefore } c$ is an argument structure, where c is called “conclusion” and is a speech act of any type and P is a set of premises, whereby there is commitment to the truth of the premises.¹¹
2. For A and B such that $A = P_A \text{ therefore } c_A$ and $B = P_B \text{ therefore } c_B$, if $P_A \subseteq P_B$ then A is a subargument structure of B .
3. The conclusion cannot be used to support its premise.
4. Premises must be consistent.
5. Nothing is an argument structure unless it obeys the rules above.

The suggested definition is common in philosophy (see, e.g., Hitchcock, 2006 for a discussion) and artificial intelligence (for an overview, see Chesñevar et al., 2000). It allows complex argument structures (in which a premise can be a conclusion of another argument structure – point 2 above), bans cyclical argument structures (point 3), and avoids inconsistent premises (point 4). Starting from the above in-

¹¹It is accepted that a premise is an assertive speech act, while the conclusion can be any speech act (van Eemeren & Grootendorst, 2004). It is also accepted that the conclusion of one argument can act as a premise for the conclusion of another argument (Hitchcock, 2006). What is needed in a premise is commitment to the truth of the content of the premise, consequently requiring that the premise be expressed by way of an assertive. Therefore, when the conclusion of an argument supports the conclusion of another, there must be commitment to the truth of the supporting conclusion. We may thus rewrite the supporting conclusion in the form more common to assertives if we are to use that conclusion to support some other conclusion.

formal conceptualization of an argument structure, we can go on to define an informal defeasible logic, in which we are interested in confronting argument structures and reaching defeasible conclusions in a structured manner. A more rigorous approach is to reuse the results from the argumentation modeling literature (see Chesñevar et al., 2000 for an overview) in the artificial intelligence field, which focuses on formalizing commonsense reasoning in the aim of automation. Therein, formal approaches to argumentation, consistent with informal ones, are suggested. These approaches rely on an argumentation model, that is, a static representation of an argumentation process, which can be seen as a search for arguments, where an argument consists of a set of rules chained to reach a conclusion. Each rule can be rebutted by another rule based on new information. To formalize such defeasible reasoning, elaborate syntax and semantics have been developed (e.g., Besnard & Hunter, 2001; Chesñevar et al., 2000; Simari & Loui, 1992) commonly involving a mathematical logic to formally represent the argumentation process and reason about interactions between argument structures. Below, we borrow Simari and Loui's argumentation system (Simari & Loui, 1992) primarily because of its integration of the accepted results in argumentation and its simplicity. Simari and Loui are concerned mainly with characterizing – by accounting only for syntactic considerations – the conditions under which an argument structure is more appropriate than another one. This is an advantage herein, as we can pursue our discussion without requiring that the content of the stakeholders' communications be formalized and rely on a formal model. Having formalized content is a rather unrealistic expectation given the cost of using formalization during RE. Still, if formalized content is available, the argumentation system described below still applies.

Assume a language \mathcal{L} . The knowledge of an agent – herein, all that a participant in RE can communicate – is referred to by a pair (\mathcal{K}, Δ) , where \mathcal{K} refers to *indefeasible knowledge*, which is assumed consistent (i.e., $\mathcal{K} \not\vdash \perp$), and Δ is a finite set of defeasible rules. A defeasible rule is of the form $p \dashrightarrow q$ and reads “ p is a reason for q ”. The defeasible consequence $\vdash\sim$ refers to a derivation from \mathcal{K} to a formula h using ground instances of Δ (we denote Δ^\downarrow the ground instances of Δ). The defeasible consequence $\vdash\sim$ is defined as follows. Let $\Gamma = \{p_1, p_2, \dots, p_n\}$, where each p_i is a member of \mathcal{K} or a member of Δ^\downarrow . A sentence p is a defeasible consequence of the set Γ , i.e., $\Gamma \vdash\sim p$ if and only if there exists a sequence q_1, \dots, q_m such that $p = q_m$ and, for each i , either q_i is an axiom of \mathcal{L} or q_i is in Γ , or q_i is a direct consequence of the preceding members of the sequence using modus ponens or instantiation of a universally quantified sentence. A subset P of Δ^\downarrow is an *argument* for c if and only if:

1. $\mathcal{K} \cup P \vdash\sim c$ (P defeasibly derives c).
2. $\mathcal{K} \cup P \not\vdash\sim \perp$ (P is consistent with regards to \mathcal{K}).
3. $\nexists P' \subset P$ such that $\mathcal{K} \cup P' \vdash\sim c$ (P is minimal).

The argument P and the conclusion c together form an *argument structure*, denoted $\langle P, c \rangle$ (terms “an argument” and “and argument structure” are often used interchangeably). An argument structure $\langle P, c \rangle$ contains subarguments: e.g., an argument structure $\langle S, q \rangle$ is a subargument of $\langle P, c \rangle$ if and only if $S \subseteq P$. Arguments can stand in the following basic relationships (Simari & Loui, 1992; Chesñevar et al., 2000):

- Two arguments $\langle P_1, c_1 \rangle$ and $\langle P_2, c_2 \rangle$ *disagree* if and only if $\mathcal{K} \cup \{c_1, c_2\} \vdash\sim \perp$.
- Given two arguments $\langle P_1, c_1 \rangle$ and $\langle P_2, c_2 \rangle$, $\langle P_1, c_1 \rangle$ *counterargues* $\langle P_2, c_2 \rangle$ at literal c if and only if there exists a subargument $\langle P, c \rangle$ of $\langle P_2, c_2 \rangle$ such that $\langle P_1, c_1 \rangle$ and $\langle P, c \rangle$ disagree.
- Given two arguments $\langle P_1, c_1 \rangle$ and $\langle P_2, c_2 \rangle$, $\langle P_1, c_1 \rangle$ *defeats* $\langle P_2, c_2 \rangle$, in the weak or strong sense, if and only if there exists a subargument $\langle P, c \rangle$ of $\langle P_2, c_2 \rangle$ such that: $\langle P_1, c_1 \rangle$ counterargues $\langle P_2, c_2 \rangle$

at the literal c and (1) $\langle P_1, c_1 \rangle$ is *strictly more specific* than $\langle P, c \rangle$, or (2) $\langle P_1, c_1 \rangle$ is unrelated by specificity to $\langle P, c \rangle$.¹²

As an example, consider the following argument structure:

```

This car is recommendable.
←-- This car is reliable.
    ←-- This car is European.
        ↯-- This car is not European.
            ←-- This car has not been assembled in Europe.
                ←-- This car has spare parts.

```

Above, we support the conclusion that this car is recommendable by stating that this car is reliable. The reliability is supported by the premises that the car has spare parts and that it is European. We have an argument that disagrees with the statement that the car is European. The premise of this argument is that the car has not been assembled in Europe. The defeasible rules we have in this example arise from, e.g., common knowledge of the reasoner, who considers that reliable cars are in general recommendable cars, that European cars tend to be reliable, and so on.

In the example above, “This car is recommendable” is a tentative conclusion. Given such a conclusion, we are interested in providing supporting or contradicting evidence in order to determine whether it is reasonable to accept that conclusion. A structured process that allows us to do so is the *justification process*, which consists of recursively defining and labeling a *dialectical tree* $T\langle P, c \rangle$ as follows (Simari & Loui, 1992):

1. A single node containing $\langle P, c \rangle$ with no defeaters is by itself a dialectical tree for $\langle P, c \rangle$. This node is also the root of the tree.
2. Suppose that $\langle P_1, c_1 \rangle, \dots, \langle P_n, c_n \rangle$ each defeats $\langle P, c \rangle$. Then the dialectical tree $T\langle P, c \rangle$ for $\langle P, c \rangle$ is built by placing $\langle P, c \rangle$ at the root of the tree and by making this node the parent node of roots of dialectical trees rooted respectively in $\langle P_1, c_1 \rangle, \dots, \langle P_n, c_n \rangle$. One way of finding arguments $\langle P_1, c_1 \rangle, \dots, \langle P_n, c_n \rangle$ that defeat $\langle P, c \rangle$ is to look for arguments that support the negation of a premise in P or the negation of the conclusion c .
3. When the tree has been constructed to a satisfactory extent by recursive application of steps 1 and 2 above, label the leaves of the tree *undefeated* (**U**). For any inner node, label it undefeated if and only if every child of that node is a *defeated* (**K**) node. An inner node will be a defeated node if and only if it has at least one **U** node as a child. Do step 4 below after the entire dialectical tree is labeled.
4. $\langle P, c \rangle$ is a *justification* (or, P justifies c) if and only if the node $\langle P, c \rangle$ is labeled **U**.

We provide an example of justification in the discussion of the justified approximation relationship (Section 5.1).

References

- Abram, A. & Moore, J.W., eds. (2004). *Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society.
- Anonymous (2006). Agreement between the European Union and the United States of America on the processing and transfer of passenger name record (PNR) data by air carriers to the United States Department of Homeland Security. *Official Journal of the European Union*, 298, 29–31.

¹²The specificity relation establishes a partial order over arguments, whereby arguments that are “more informed” are rank higher (see, e.g., Simari & Loui, 1992 for details). Other criteria for ordering arguments may be valid.

- Anton, A.I. (1996). Goal-based requirements analysis. In *Proceedings of the International Conference on Requirements Engineering*. IEEE (pp. 136–144).
- Austin, J.L. (1962). *How To Do Things with Words*. Harvard University Press.
- Bellini, P., Mattolini, R. & Nesi, P. (2000). Temporal logics for real-time system specification. *ACM Comput. Surv.*, 32(1), 12–42.
- Bennett, J. (2003). *A Philosophical Guide to Conditionals*. Oxford: Clarendon Press.
- Besnard, P. & Hunter, A. (2001). A logic-based theory of deductive arguments. *Artif. Intell.*, 128(1,2), 203–235.
- Bizer, G.Y., Barden, J.C. & Petty, R.E. (2003). Attitudes. In *Encyclopedia of Cognitive Science*. Hampshire, England: MacMillan.
- Bizer, G.Y., Barden, J.C. & Petty, R.E. (2003). Attitudes. In *Encyclopedia of Cognitive Science*. Hampshire, England: MacMillan.
- Boehm, B.W., Brown, J.R. & Lipow, M. (1976). Quantitative evaluation of software quality. In *ICSE*. IEEE (pp. 592–605).
- Boehm, B.W., Egyed, A., Kwan, J., Port, D., Shah, A. & Madachy, R.J. (1998). Using the winwin spiral model: A case study. *IEEE Computer*, 31(7), 33–44.
- Borgo, S. & Vieu, L. (2009). Artefacts in formal ontology. In A. Meijers (ed.), *Handbook of Philosophy of Technology and Engineering Sciences*. Elsevier.
- Camerer, C., Loewenstein, G. & Prelec, D. (2005). Neuroeconomics: How neuroscience can inform economics. *Journal of Economic Literature*, 43(1), 9–64.
- Castro, J., Kolp, M. & Mylopoulos, J. (2002). Towards requirements-driven information systems engineering: the tropos project. *Inf. Syst.*, 27(6), 365–389.
- Chesñevar, C.I., Maguitman, A.G. & Loui, R.P. (2000). Logical models of argument. *ACM Comput. Surv.*, 32(4), 337–383.
- Chung, L., Nixon, B.A., Yu, E. & Mylopoulos, J. (1999). *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers.
- Cimiano, P., Eberhart, A., Hitzler, P., Oberle, D., Staab, S. & Studer, R. (2004). The smartweb foundational ontology. Technical report, SmartWeb Project Report, SEP.
- Cohen, P.R. & Levesque, H.J. (1990). Intention is choice with commitment. *Artif. Intell.*, 42(2,3), 213–261.
- Customs and Border Protection, Department of Homeland Security (2008). Changes to the Visa waiver program to implement the Electronic System for Travel Authorization (ESTA). Program DOCID:fr09jn08-3. *Federal Register*, 73(111), 32440–32453.
- Damasio, A.R. (1994). *Descarte's Error: Emotion, Reason and the Human Brain*. New York: Avon.
- Damasio, A.R., Tranel, D. & Damasio, H. (1991). Somatic markers and the guidance of behavior: theory and preliminary testing. In H.S. Levin, H.M. Eisenberg & H.M. Benton (eds), *Frontal Lobe Function and Dysfunction* (pp. 217–229). New York: Oxford University Press.
- Dardenne, A., van Lamsweerde, A. & Fickas, S. (1993). Goal-directed requirements acquisition. *Sci. Comput. Program.*, 20(1,2), 3–50.
- Darimont, R. & van Lamsweerde A. (1996). Formal refinement patterns for goal-driven requirements elaboration. In *SIGSOFT FSE* (pp. 179–190).
- Deming, W.E. (1982). Quality, productivity, and competitive position. Massachusetts Institute of Technology, Center for Advanced Engineering Study.
- Dijkstra, E.W. (1972). Notes on structured programming. In *Structured Programming*. Academic Press (pp. 1–82, Chapter i).
- Donzelli, P. (2004). A goal-driven and agent-based requirements engineering framework. *Requir. Eng.*, 9(1), 16–39.
- Dromey, R.G. (1995). A model for software product quality. *IEEE Trans. Softw. Eng.*, 21, 146–162.
- Dunn, B.D., Dalgleish, T. & Lawrence, A.D. (2006). The somatic marker hypothesis: A critical evaluation. *Neuroscience and Biobehavioral Reviews*, 30, 239–271.
- Eagly, A. & Chaiken, S. (1998). Attitude structure and function. In *The Handbook of Social Psychology* (4th edn). New York: McGraw-Hill.
- EC Article 29 Data Protection Working Party (2007). Opinion 2/2007 on information to passengers about transfer of PNR data to US authorities. Technical report, European Commission.
- Elster, J. (1998). Emotions and economic theory. *Journal of Economic Literature*, 36(1), 47–74.
- Ferrario, R. & Oltramari, A. (2004). Towards a computational ontology of mind. In *Proceedings of Formal Ontologies in Information Systems FOIS-2004*. Amsterdam: IOS Press (pp. 1–9).
- International Organization for Standardization (2001). ISO 9126-1:2001, Software engineering – Product quality, Part 1: Quality model. International Organization for Standardization.
- Fox, C. & Frakes, W. (1997). The quality approach: is it delivering? *Commun. ACM*, 40(6), 24–29.
- Frege, G. (1879). *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Halle a/S.: L. Nebert.

- Genesereth, M.R. & Nilsson, N.J. (1987). *Logical Foundations of Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann.
- Giorgini, P., Mylopoulos, J., Nicchiarelli, E. & Sebastiani, R. (2003). Formal reasoning techniques for goal models. *J. Data Semantics, 1*, 1–20.
- Glinz, M. (2007). On non-functional requirements. In *Proceedings of the 15th IEEE International Requirements Engineering Conference*, 2007. IEEE (pp. 21–26).
- Goguen, J.A. & Linde, C. (1993). Techniques for requirements elicitation. In *Proc. Int. Symp. Req. Eng.* IEEE (pp. 152–164).
- Gruber, T.R. (1993). Toward principles for the design of ontologies used for knowledge sharing. In N. Guarino & R. Poli (eds), *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers (pp. 907–928).
- Herre, H. & Heller, B. (2006). Semantic foundations of medical information systems based on top-level ontologies. *Knowledge-Based Systems, 19*(2), 107–115. (Intelligent Software Design.)
- Hitchcock, D. (2006). The concept of argument, and informal logic. In J. Woods, J. Gabbay & P. Thagard (eds), *Philosophy of Logic*, Handbook of the Philosophy of Science 5. Elsevier.
- Hoare, C.A.R. (1972). Proof of correctness of data representations. *Acta Inf., 1*, 271–281.
- Hsee, C.K. (1996). The evaluability hypothesis: An explanation for preference reversals between joint and separate evaluations of alternatives. *Organizational Behavior and Human Decision Processes, 67*(3), 247–257.
- IEEE (1991). IEEE 610 – standard glossary of software engineering terminology. Technical report, IEEE.
- IEEE (2004). Guide to the software engineering body of knowledge (www.swebok.org). Technical report, IEEE.
- Carnegie Mellon University Software Engineering Institute (1995). *Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley.
- Jespersen, O. (1963). *The Philosophy of Grammar*. London: George Allen & Unwin Ltd.
- Jiang, L., Topaloglou, T., Borgida, A. & Mylopoulos, J. (2006). Incorporating goal analysis in database design: A case study from biological data management. In *14th IEEE International Conference on Requirements Engineering (RE 2006)*, Minneapolis/St.Paul, MN, USA, 11–15 September 2006 (pp. 196–204).
- Juran, J.M. (1951). *Quality Control Handbook*. McGraw-Hill.
- Jureta, I.J., Faulkner, S. & Schobbens, P.-Y. (2006). A more expressive softgoal conceptualization for quality requirements analysis. In *Proceedings of the 25th International Conference on Conceptual Modelling (ER'06)*. Springer (pp. 281–295).
- Jureta, I.J., Faulkner, S. & Schobbens, P.-Y. (2007). Achieving, satisficing, excelling. In *Proceedings of the 1st International Workshop on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM'07)*. Springer (pp. 286–295).
- Jureta, I.J., Mylopoulos, J. & Faulkner, S. (2008). Revisiting the core ontology and problem in requirements engineering. In *16th IEEE Int. Requirements Engineering Conference*. IEEE (pp. 71–80).
- Kahneman, D., Ritov, I. & Schkade, D. (1999). Economic preferences or attitude expressions?: An analysis of dollar responses to public issues. *J. Risk and Uncertainty, 19*, 203–235.
- Kavakli, E. (1999). Goal-driven requirements engineering: Modeling and guidance. PhD thesis, University of Manchester.
- Keeney, R.L. & Raiffa, H. (1976). *Decisions with Multiple Objectives*. New York: Wiley.
- Kitchenham, B. & Pfleeger, S.L. (1996). Software quality: The elusive target. *IEEE Softw., 13*(1), 12–21.
- Koons, R. (2005). Defeasible reasoning. In E.N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*. Spring.
- Kotonya, G. & Sommerville, I. (2000). *Requirements Engineering: Processes and Techniques*. Wiley.
- Lenat, D.B., Guha, R.V., Pittman, K., Pratt, D. & Shepherd, M. (1990). Cyc: Toward programs with common sense. *Commun. ACM, 33*(8), 30–49.
- Letier, E. (2001). Reasoning about agents in goal-oriented requirements engineering. PhD thesis, Dept. d'Ingénierie Informatique, Université catholique de Louvain.
- Letier, E. & van Lamsweerde, A. (2004). Reasoning about partial goal satisfaction for requirements and design engineering. In *Proceedings of the 12th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, Newport Beach, CA, USA, 31 October–6 November 2004 (pp. 53–62).
- Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltamari, A. & Schneider, L. (2003). DOLCE: A Descriptive Ontology for Linguistic and Cognitive Engineering. Technical report, Institute of Cognitive Science and Technology, Italian National Research Council.
- Masolo, C., Borgo, S., Gangemi, A., Guarino, N. & Oltamari, A. (2003). Ontology library (final). Technical report, WonderWeb Deliverable D18, December 2003. Available at: <http://wonderweb.semanticweb.org>.
- McShane, M., Nirenburg, S. & Zacharski, R. (2004). Mood and modality: out of the theory and into the fray. *Natural Language Engineering, 10*(1), 57–89.
- Menzel, C. (2007). Actualism. In E.N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*. Spring.
- Mylopoulos, J., Chung, L. & Nixon, B. (1992). Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Trans. Softw. Eng., 18*(6), 483–497.
- Mylopoulos, J., Jureta, I. & Faulkner, S. (2007). An ontology for requirements. In *Advances in Conceptual Modeling – Foundations and Applications, ER 2007 Workshops CMLSA, FP-UML, ONISW, QoIS, RIGiM, SeCoGIS*. Springer (p. 224).

- Nielsen, J., Farrell, S., Snyder, C. & Molich, R. (2001). E-commerce user experience: Checkout and registration. Nielsen Norman Group. Available at: <http://www.nngroup.com/reports/ecommerce/checkout.html>.
- Niles, I. & Pease, A. (2001). Towards a standard upper ontology. In *FOIS* (pp. 2–9).
- Oberle, D. (2006). *Semantic Management of Middleware, Vol. 1 of the Semantic Web and Beyond*. Springer.
- Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Sintek, M., Kiesel, M., Mougouie, B., Baumann, S., Vembu, S. & Romanelli, M. (2007). DOLCE ergo SUMO: On foundational and domain models in the SmartWeb Integrated Ontology (SWIntO). *J. Web Sem.*, 5(3), 156–174.
- Oliver R. (1977). Effect of expectation and disconfirmation on post-exposure product evaluation: An alternative interpretation. *Journal of Applied Psychology*, 62, 480–486.
- OMG (2005). UML profile for modeling Qos and fault tolerance characteristics and mechanisms specification, v1.0. Object Management Group.
- Oppy, G. (1998). Propositional attitudes. In *Routledge Encyclopedia of Philosophy*. London: Routledge.
- Parasuraman, A., Berry, L.L. & Zeithaml, V.A. (1988). SERVQUAL: A multiple-item scale for measuring consumer perceptions of service quality. *Journal of Retailing*, 64, 12–40.
- Parasuraman, A. & Zeithaml, V.A. (2006). Understanding and improving service quality: A literature review and research agenda. In B. Weitz & R. Wensley (eds), *Handbook of Marketing*. Sage Publications (pp. 339–370).
- Peirce, C.S. (1935). *Collected Papers of Charles Sanders Peirce*. Cambridge, MA: Harvard University Press.
- Rao, A.S. & Georgeff, M.P. (1991). Modeling rational agents within a bdi-architecture. In *KR* (pp. 473–484).
- Reeves, C.A. & Bednar, D.A. (1994). Defining quality: Alternatives and implications. *The Academy of Management Review, Special Issue: Total Quality*, 19(3), 419–445.
- Regev, G. & Wegmann, A. (2005). Where do goals come from: the underlying principles of goal-oriented requirements engineering. In *13th IEEE International Conference on Requirements Engineering (RE 2005)*, Paris, France, 29 August–2 September 2005 (pp. 353–362).
- Robbins, L. (1981). Economics and political economy. *The American Economic Review*, 71(2), 1–10.
- Ross, D.T. & Schoman, K.E. Jr. (1977). Structured analysis for requirements definition. *IEEE Trans. Software Eng.*, 3(1), 6–15.
- Sanford, D.H. (1989). *If P, then Q: Conditionals and the Foundations of Reasoning*. London and New York: Routledge.
- Schneider, L. (2003). Designing foundational ontologies: The object-centered high-level reference ontology ochre as a case study. In *Proceedings of the International Conference on Conceptual Modeling (ER)* (pp. 91–104).
- Searle, J.R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge: Cambridge University Press.
- Searle, J.R. (1975). A taxonomy of illocutionary acts. In *Language, Mind, and Knowledge* (Studies in the Philosophy of Science, Vol. 7). Minneapolis: University of Minnesota Press.
- Searle, J.R. (1980). The background of meaning. In J.R. Searle, F. Kiefer & M. Bierwisch (eds), *Speech Act Theory and Pragmatics* (pp. 221–232). Dordrecht, The Netherlands: D. Reidel Publishing Co.
- Sen, A. (1973). Behaviour and the concept of preference. *Economica*, 40(159), 241–259.
- Sen, A. (1993). Internal consistency of choice. *Econometrica*, 61(3), 495–521.
- Simari, G.R. & Loui, R.P. (1992). A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, 53(2,3), 125–157.
- Simon, H.A. (1955). A behavioral model of rational choice. *The Quarterly Journal of Economics*, 69(1), 99–118.
- Spear, A.D. (2006). Ontology for the twenty first century: An introduction with recommendations. Technical report, Institute for Formal Ontology and Medical Information Science (IFOMIS), Saarbrücken, Germany.
- Stenius, E. (1967). Mood and language game. *Synthese*, 17, 254–274.
- Uschold, M. & Gruninger, M. (1996). Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11(2).
- van Eemeren, F.H. & Grootendorst, R. (2004). *A Systematic Theory of Argumentation: The Pragma-Dialectical Approach*. Cambridge: Cambridge University Press.
- van Lamswerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *5th IEEE International Symposium on Requirements Engineering (RE)* (p. 249).
- Varian, H. (1984). *Microeconomic Analysis*. New York: Norton.
- Wirth, N. (1971). Program development by stepwise refinement. *Commun. ACM*, 14(4), 221–227.
- Wittgenstein, L. (1953). *Philosophical Investigations*. Oxford: Blackwell.
- Wittgenstein, L. (2001). *Tractatus Logico Philosophicus* (2 edn). Routledge.
- Yu, E. (1997). Towards modeling and reasoning support for early requirements engineering. In *Proceedings of the IEEE International Symposium on Requirements Engineering*. IEEE (pp. 226–235).
- Zaefferer, D. (2006). Deskewing the searlean picture – a new speech act ontology for linguistics. In *Proceedings of the 32nd Annual Meeting of the Berkeley Linguistic Society (BLS-32)*. Berkeley Linguistic Society.
- Zave, P. (1997). Classification of research efforts in requirements engineering. *ACM Comput. Surv.*, 29(4), 315–321.
- Zave, P. & Jackson, M. (1997). Four dark corners of requirements engineering. *ACM T. Softw. Eng. Methodol.*, 6(1), 1–30.
- Zave, P. (1997). Classification of research efforts in requirements engineering. *ACM Comput. Surv.*, 29(4), 315–321.