Towards an Ontology of Goal-Oriented Requirements

Pedro Pignaton Negri[1], Vítor E. Silva Souza[1], André Luiz de Castro Leal[2], Ricardo de Almeida Falbo[1], and Giancarlo Guizzardi[1]

[1] Ontology and Conceptual Modeling Research Group (NEMO)
Department of Informatics, Federal University of Espírito Santo (UFES), Brazil
`pedropn@gmail.com`, {`vitorsouza,falbo,gguizzardi`}`@inf.ufes.br`,
[2] Department of Mathematics
Federal Rural University of Rio de Janeiro (UFFRJ), Brazil
`andrecastr@gmail.com`

Abstract. Goal-Oriented Requirements Engineering (GORE) has grown into an important area of research in the past decades. Still, some of its corners remain dark, since different GORE languages do not provide a well-founded conceptualization of the domain and are not consensual. This may lead to ambiguous or weak understanding of GORE concepts. In this paper, we introduce the Goal-Oriented Requirements Ontology (GORO), a domain ontology founded on the Unified Foundational Ontology (UFO) that intends to represent the nature and relations of concepts surrounding the GORE domain. We use GORO to explore and clarify the semantics used, sometimes implicitly, by well-known GORE languages.

Keywords: requirements, ontology, goal-oriented, GORE, UFO, GORO

1 Introduction

Goals have been assuming an important role in Requirements Engineering (RE). Increasing the domain understanding and clarifying stakeholders' intentions, Goal-Oriented Requirements Engineering (GORE) uses goals to elicit, elaborate, specify, analyze and negotiate requirements. GORE allows a better identification and understanding of requirements for stakeholders, compared to previous approaches [22].

Because of that, many goal modeling languages emerged to support the RE process, such as $i^\star$ [34], KAOS [6], Techne [2], among others. These languages intend to represent, essentially, the same concepts (goals and related notions). However, given that they have been built independently, we cannot ignore that their underlying constructs might not share the same semantics. Furthermore, these constructs are mostly described with metamodels—which are powerful structures to define a language's abstract syntax, but not to clarify its semantics [14]. We argue for the necessity of a well-founded conceptual model of the goal-oriented requirements domain to explore in depth the conceptualizations involved: a domain reference ontology for GORE.

As Guarino [1] and Guizzardi [10] argue, a domain ontology should be developed grounded in a foundational ontology—a domain-independent formal system of categories and their ties, that can be used to describe specific domains in reality. Therefore, we propose the Goal-Oriented Requirements Ontology (GORO), a domain ontology founded in UFO [10]. Most of GORO was built reusing existing ontologies, extracting, applying and connecting them, with focus on the GORE domain. In this work, we focus on the *goal* concept, the central term in all GORE approaches, and surrounding concepts. The consensus to build GORO was extracted from extensive research on goal modeling languages.

The purpose of GORO is to give better grounded elements to clarify and explore the nature of concepts involved. This impacts in a better communication between the stakeholders, since a more precise description of the entities of the domain is provided. Also, goal modeling languages can ground their concepts in a formal reference ontology that gives the possibility of interoperability between different models and, hence, translation between model concepts.

This paper is structured as follows. Section 2 gives us a background on GORE and some well-known GORE frameworks. Section 3 presents the Goal-Oriented Requirements Ontology (GORO), which is then evaluated in Section 4. Section 5 discusses related work and, finally, Section 6 concludes the paper.

2 Goal-Oriented Requirements Engineering

In the late 1990's, researchers have recognized limitations in traditional Requirements Engineering (RE) practice [27,25]. As response, new approaches were proposed around the concept of *goals*, giving birth to the research (sub)field of *Goal-Oriented Requirement Engineering* (GORE). Goals are declarative statements of intent to be achieved. They can express functional properties (a capability, capacity) or non-functional properties (quality) at different levels of abstraction: from high-level strategic goals (e.g., use of energy be optimized) to low-level tactical goals (e.g., lights be turned off at the end of the day). Satisfying goals can be considered the requirements for an intended system.

GORE brings several benefits to RE practice, such as: precise criteria for sufficient completeness of a requirement specification; the rationale of each requirement, justifying their existence; a natural mechanism for structuring complex requirements documents that increases readability; a clearer way to detect and resolve conflicts among requirements; etc. [22]. These benefits have stimulated the proposal of many different GORE approaches.

For instance, the $i^\star$ Framework [34], which proposes different models to different levels of abstraction. At the high level, Strategic Dependency models aim to describe the dependency relationships amongst actors: an actor depends on some other in order to accomplish some intention. An intentional element can be a goal, a task, a resource or a *softgoal*. At the lower level, Strategic Rationale models describe and refine the intentional elements into the boundary of each actor, exploring the rationale behind them. The $i^\star$ approach focuses on the early-requirements phase, identifying stakeholders and bringing their strategic goals into models. The purpose is to provide tools to model social aspects in RE based on goals. Tropos [3], an $i^\star$ variant, includes the goal decomposition concept to decrease the level of abstraction towards a late-requirements specification. A new version of $i^\star$ (now spelled *iStar*) includes goal refinement, which can be used to derive system requirements from high-level goals [5].

KAOS [6] is a systematic approach for discovering and structuring requirements. We are interested on their conceptual models and the associated language. In KAOS, a goal is a prescriptive statement of intent that a system should satisfy [24]. The goal model intends to describe the goals to be satisfied by (through) the system (software + environment), expressing also high-level goals, which are related to the system scope and stakeholders' intentions (more strategic), and low-level goals (more technical). The top-level goals can be consecutively refined/decomposed into lower-level, more operational ones, to express how they can be achieved.

Techne [2] is a requirements modeling language founded on the Core Ontology for RE (CORE) [20] which, in turn, is based on DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) [26], a foundational ontology that aims to capture the ontological categories underlying natural language and human common sense. Techne distinguishes among different mental states of stakeholders to represent concepts in the goal model. Agent desires are requirements that the system should satisfy and are captured through instances of the *goal* concept when they describe a verifiable functional condition, otherwise they are captured through other concepts like *softgoal* (which is extensively studied in [17]). *Beliefs* imply in *domain assumptions* which are a supposed state-of-affairs held about the system-to-be and/or its environment of interest. *Intentions* indicate commitments to act towards requirements satisfaction and, thus, they are captured via the *task* concept.

Aiming at providing a formal semantics to the concepts of these languages, also allowing interoperability between them, we propose the Goal-Oriented Requirements Ontology (GORO) and ground it on the Unified Foundational Ontology (UFO) [10,11]. We chose UFO because it has been constructed with the primary goal of developing foundations for conceptual modeling. Consequently, UFO addresses many essential aspects for conceptual modeling, which have not received a sufficiently detailed attention in other foundational ontologies [10]. Examples are the notions of material relations and relational properties. For instance, this issue did not receive up to now a treatment in DOLCE [26], which focuses solely on intrinsic properties (qualities). Moreover, UFO has been successfully employed in a number of semantic analyses, such as the one conducted here (see detailed discussion in [12]).

3 The Goal-Oriented Requirements Ontology

To build the Goal-Oriented Requirements Ontology (GORO) we used SABiO, a Systematic Approach for Building Ontologies [8]. With the purpose of building a reference ontology, we focused on SABiO's first two phases: (1) purpose identification and requirements elicitation; (2) ontology capture and formalization.

In the first phase, *Competency Questions* (CQs) were elicited as functional requirements for GORO. Due to space constraints, we present these CQs later in Section 4 (cf. Table 1), as they are used in ontology verification. As non-functional requirements, we specified that GORO should: be built using the characterization of the domain given by different GORE approaches in the literature (NFR1); be grounded on a well-know foundational ontology (NFR2); and reuse existing ontologies on related subjects (NFR3).

To satisfy NFR1, we explored the conceptualizations given by three different GORE approaches, presented in Section 2. KAOS and iStar were chosen due to their popularity: in a recent survey [18], KAOS and iStar(-based) approaches were used in over a third of the total of publications. Techne, on the other hand, was included due to the fact that it is already based on an ontology (CORE [20]). Regarding NFR2, we grounded GORO on the Unified Foundational Ontology (UFO) [10,11]. Finally, with respect to NFR3, we reused some well-founded conceptualizations of important concepts in our domain in the literature, namely: Guizzardi et al.'s interpretation of non-functional requirements (NFR) [17]; and Wang et al.'s contribution on assumptions (ASMP) [32]. We reuse these isolated conceptualizations, connecting their fragments to compose GORO.

GORO is presented in figures 1 and 2 in UML (Unified Modeling Language) class diagrams, used here primarily for visualization. Concepts from the reused ontologies are prefixed by their aforementioned acronyms. In the following subsections, we discuss different parts of GORO, highlighting concepts from the ontology using a Sans Serif font.

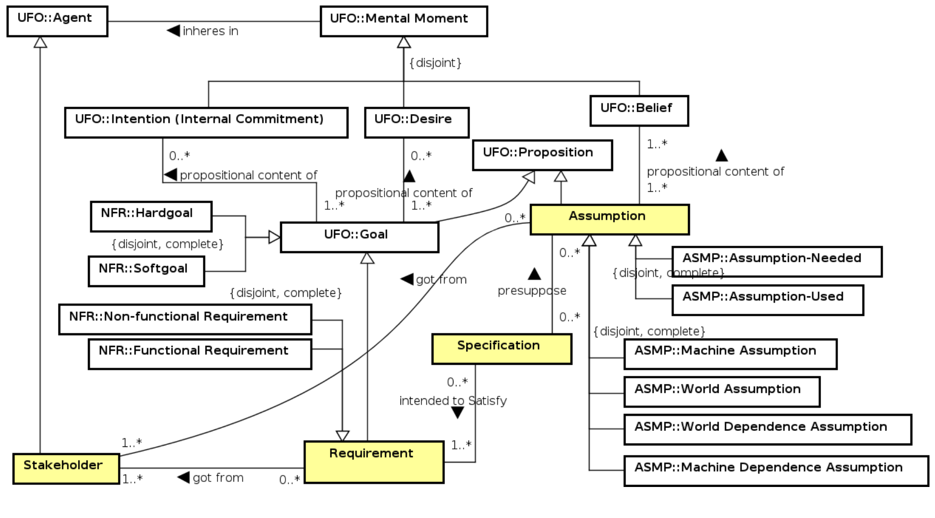

Fig. 1. GORO fragment focusing on the mental moment elements.

Intentions and Desires. In GORE, the domain characterization (and further identification of requirements and so on) is obtained initially by modeling the agents' goals. Figure 1 presents the GORO fragment focusing on agent's mental moment elements.

According to UFO, an Agent is a Concrete Individual or Particular (exists in reality and possesses an identity), that is Endurant (does not have temporal parts, but rather exists in time) and Substantial (existentially independent). Agents bear intentional properties such as Beliefs, Desires, Intentions, as opposed to Objects, which are non-agentive. These intentional properties are called Mental Moments, which are Particulars that are existentially dependent (a Moment only exists in other particulars), Intrinsic (dependent of one single individual) and Intentional (inheres in an Agent).

A Goal is a propositional content of two possible Mental Moments: the Agent's Intention or Desire. A Desire expresses the will of an Agent towards a particular state of affairs in reality (i.e., the Goal). An Intention represents not only a will but also an internal commitment of the Agent to act towards the Goal (implying, as shown in Figure 2 and discussed later, in a Plan to accomplish it). In that way, both Desire and Intention refer to an Agent's Goal, yet a Goal as an Intention is always associated with a Plan [15]. In a goal model, it is not possible to determine if a Goal is a propositional content of an Intention or a Desire before its completion. Considering that models may or may not be extensively detailed, we usually cannot ascertain about the Mental Moment type of a Goal. On the other hand, Goals with associated Plans to accomplish them, e.g. *tasks*, are notoriously propositional contents of Intentions.

Goals and Requirements. As described in Zave & Jackson's seminal paper [35] and refined by Wang et al. [31], a requirement is a desired behavior in the environment independently of the machine. It is important to emphasize that, even in a Software Engineering process, a requirement does not refer necessarily to software. Hence, a Requirement is a Goal, in the scope of a specific problem, that describes environmental conditions to be achieved through a desired solution resulting in satisfaction of the underlying strategic goals. The solution can involve software behavior, process models, organizational systems and so on (commonly referred to as a *socio-technical system*). A solution Specification is planned to be implemented in order to achieve the goal requirements. In fact, also the stakeholders from whom the goals and requirements are being elicited can define some specific solution specification to accomplish their problem. It is important to emphasize that a Specification is not an artifact, a document, in essence, but it could exist only in an agent's mind [31]. Thus, as Figure 1 shows, Goals become Requirements, and Agents become Stakeholders when they are part of a Requirements Engineering process, i.e., there is an effort to devise a Specification (software-based or otherwise) that satisfies the Requirements.

Hardgoals, Softgoals, Functional and Non-functional Requirements. GORO adopts the conceptualization given by Guizzardi et al. [17] that clarifies the discussion about Hardgoal, Softgoal, Functional and Non-Functional Requirement concepts. Hardgoals are defined as propositions that are objectively satisfied by a given set of situations. In constrast, Softgoals refer to a vague quality region whose exact boundaries are unknown. Not being possible to determine *a priori* the set of situations that satisfies them, their satisfiability is usually related to an agent's judgement. Thus, Softgoals, can be an initial and temporary vague expression of a goal not sufficiently defined yet.

On the other hand, Functional Requirements refer to functions (capabilities, capacities) that have the potential to manifest certain behavior in particular situations, whereas Non-Functional Requirements refer to qualities. The work clearly distinguish Hardgoals and Softgoals as orthogonal to the concepts of Functional and Non-Functional Requirements.

Assumptions. In GORE, Assumptions describe states-of-affairs in the environment of interest that the Stakeholders believe to be true, i.e., they are the propositional content of Stakeholders' Beliefs. They do not express the aim of an Agent towards a Situation in reality as Goals do, but a Belief that a specific Situation exists in the environment. Representing such Situations is sometimes necessary because they need to be considered in a given solution to a specific problem.

A recent work [32] explores the different kinds of Assumption, as shown in Figure 1. A World Assumption is about world phenomena, not visible by the machine. In a system that controls a room temperature, for example, a World Assumption could be *the air conditioner has cooling capacity to refrigerate the room.* A Machine Assumption is, on the other hand, about a machine's internal phenomena. Machine Dependence Assumption and World Dependence Assumption connect the world and machine states. The former represents an external world phenomenon that depends on some machine phenomenon whereas the latter specifies a machine phenomenon that depends on some world phenomenon. These imply that phenomena occurred in the machine and the world have to be simultaneously reflected in the world and the machine, to guarantee that states of the world are correctly represented at the machine and vice versa.

Representing all the related Assumptions is unusual in GORE since they are sometimes obvious. In the previous example, it is assumed that the system's data should reflect the real temperature of the room - in this case, the world dependence assumption is usually suppressed. However, representing some assumptions becomes necessary in situations when they are not too obvious or, even, when the modeler wants to emphasize them. In the same system examplified, we could have the goal of "*maintaining the supervisor informed via text message when the room temperature rises to more than a 310K*". The Machine Dependence Assumption could be "*when the message is sent by the system, the supervisor will receive it*". Revealing this assumption could be important since it exposes relevant nuances that could not have been thought of a priori.

An orthogonal distinction is made between Assumptions-Used and Assumptions-Needed. The former are propositions that are used *a priori* when constructing a new argument. The latter are propositions that are needed to support a previous conclusion, explaining the situation in which the designed solution works. Assumptions-used talks about properties that will naturally hold in the context that the solution exists and need to be strongly considered in the system creation i.e., the *physical laws.* Assumptions-needed is about properties that need to be held in someway - there is a direct action to perform it (making the assumption happen) - so that the system works succesfully, i.e., *the air conditioner has cooling capacity to refrigerate the room*. To express or not these kinds of elements is a modeler's decision but, again, expliciting these different assumptions' kinds in a GORE model may be helpful to the understanding.

Decomposition, Alternatives, Operationalization and Plans. Figure 2 shows the GORO fragment focuses on goal relations. At first, the Decomposition concept refines a Goal Requirement in its subgoals. Being Goal a Proposition, we can separate it in different parts that, together, compose the original ($G \iff G_1 \wedge G_2 \wedge G_3 \wedge ... \wedge G_n$). G is satisfied by exactly those Situations which satisfy $G_1...G_n$ conjunctively and, so, satisfying all subgoals $G_1, G_2, \ldots, G_n$ means satisfying the supergoal G [13]. Decomposes is a Formal Relation, i.e., it holds between two or more entities directly without any further intervening individual. Figure 2 also presents the concept of Goal *alternative*. In that relation, a Goal is not refined in its sub-parts but in alternative goals in which a Situation that satisfies each one of them (separately) also satisfies the main goal. So, considering S as satisfying: $S(G) \iff S(G_1) \vee S(G_2) \vee \ldots \vee S(G_n)$ [14]. Alternative is also a Formal Relation since it holds when the Situation that satisfies the alternative goals also satisfies the main goal. In that way, the relation exists because of intrinsic properties of the Goals.

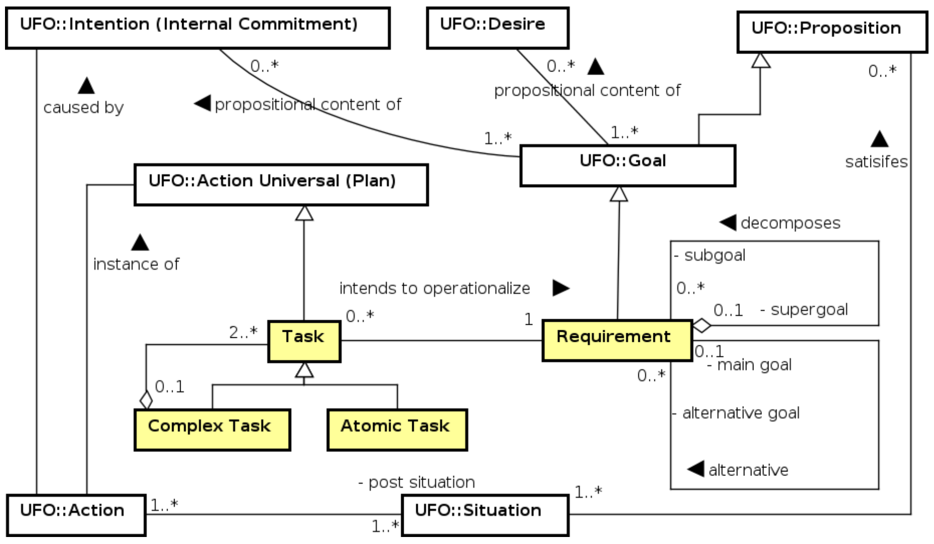

Fig. 2. GORO fragment focusing on relations.

Stakeholders may have the will to satisfy a Requirement acting in a specific way or, also, modelers want to describe in the model, giving a taste of the solution domain, *how* to satisfy some Requirement (i.e., following a Plan). The intends to operationalize link connects a Task and a Requirement expressing which way a desired state-of-affairs in reality should be reached or, in other words, how to satisfy the Requirement. It means that one or more execution of that Task (Action instances) produce a post-situation which satisfies the Goal (a Proposition) [13].

A Plan is simply a specific way to do something, but in a GORE context Tasks reveal the Intention of Stakeholders to pursue a Goal as Requirement in this specific way. So, every Task in a GORE model has, intrinsically, an associated Goal: the Goal to accomplish that Task. In that way, the Goal to accomplish a specific Task can be a Requirement. Goals as propositional contents of Intentions are always associated with Task as means to achieve the Goal since there is an internal commitment of the Agent to pursuing that Goal. Tasks can be Atomic Tasks or Complex Tasks, the latter being composed by two or more Tasks (Atomic or Complex ones).

Levels of Abstraction. GORE is primarily about early requirements, i.e., characterizing the problem domain, rather than detailing the solution. Many proposals, however, take GORE concepts further to late requirements (describing the solution) or even to architectural design. Dealing with Goals at different levels of abstraction is an important issue: they can be more strategic, dealing with issues about the problem domain that justify and motivate the solution; or more technical, closer to the solution domain, defining how the strategic goals will be satisfied. However, the threshold between strategic and technical goals can be fuzzy and relative and, so, comparisons of their levels of abstraction can be made.

The different kinds of relations from GORO change the goals' levels of abstraction through particular manners. The decomposes relation usually details a goal. The supergoal is decomposed specifying its subparts that might not have been initially identified. The nuances of the main goal are then described expanding the understanding of this goal and showing how it can be satisfied by satisfying it subparts and making it less strategic. Alternative relations lower the level of abstraction of a goal by providing different ways (*how*) to satisfy it. Intends to operationalize relation, in essence, is used to describe a way to satisfy a Goal and, therefore, represents a big change in the level of abstraction explaining explicitly and directly *how* to satisfy a Goal.

4 Evaluation

GORO has been evaluated using verification and validation techniques proposed by SABiO. For verification, we have checked that the ontology is able to answer the competency questions (CQs) that were raised as part of its requirements elicitation. Table 1 illustrates the result of this activity, which can also be used as a traceability tool, supporting ontology change management. The table shows that GORO answers all of its CQs.

For validation, Table 2 shows that GORO can represent the constructs included in well-known GORE languages. In what follows, we discuss these constructs and how they are represented in GORO.

KAOS and Techne do not model explicitly the Stakeholder in whom the mental moment inheres. Being Requirements Engineering mostly a social activity,

CQ	Description	Concepts and *Relations*
CQ1	What is a goal?	A Goal is a *subtype of* Proposition and the *propositional content of* and Intention or a Desire.
CQ2	In which way an agent 'has' a goal?	The aforementioned Intention and Desire are *subtype of* Mental Moment, which *inhere in* an Agent.
CQ3	What is a requirement?	A Requirement is a *subtype of* Goal and the *propositional content of* a Agent Mental Moment which is *got from* that Agent as Stakeholder.
CQ4	How does a goal interfere in a system-to-be?	A Specification for a system-to-be is *intended to satisfy* one or more Requirements, which is a *subtype of* Goal.
CQ5	How do goals relate with each other?	A decomposes relates a Goal as *supergoal* to other Goals as *subgoals*. An alternative relates a Goal as *main goal* to other Goals as *alternative goals*.
CQ6	How can goals be satisfied?	An intends to operationalize is a Formal Relation between Goal and Plan; an Action is an *instance of* a Plan whose *post-situation* is a Situation that *satisfies* the Goal (via its *supertype* Proposition). Goals can also be satisfied via decomposes (satisfaction of all *subgoals* satisfy the *supergoal*) or alternative (satisfaction of any *alternative goal* satisfies the *main goal*), as per CQ5.
CQ7	What is an assumption?	An Assumption is a *subtype of* Proposition and the *propositional content of* a Belief, which is a *subtype of* Mental Moment and is *got from* a Stakeholder.

Table 1. Verification: GORO's competency questions and how it answers them.

we observe the importance of representing these agents, as iStar does (via the Actor construct).

KAOS explicitly defines goals as softgoals/behavioral goal and functional/non-functional [24]. However, the term 'behavioral' goal expressing the opposite of softgoal is inaccurate since it can be associated to functional property. Also, in KAOS, softgoals are primarily used for comparing alternative goal refinements. Techne approaches goal, softgoal and quality constraint. However, a goal is defined as a verifiable functional condition, a quality constraint restricts values of non-binary measurable characteristics of the system-to-be and softgoal does so over qualities with ill-defined quality spaces. In its new version, iStar [5] distinguishes between goal and quality being the first equivalent to GORO's Hardgoal and the second similar to Non-functional requirement. Even in the oldest version of the language [33] [34], the softgoal is associated with non-functional requirements. All these different definitions from these different approaches reveal some confusion in using these concepts. Also, they are not formally defined in the approaches. GORO adopts a conceptualization given by an well-founded ontological interpretation [17] as exposed in Section 3, defining Hardgoal and Softgoal as orthogonal to Functional Requirement and Non-functional Requirement.

In Techne, a stakeholder's desire becomes an instance of Goal, whereas their intention to act in specific ways becomes an instance of Task [19]. A Techne Task is interpreted as a Task in GORO since it is, in fact, not an action or an event, but a plan defining a specific way to act. GORO goes beyond, since an intention has also an associated desire — the desired state-of-affairs intended — and, hence, a Requirement, along with a Task to satisfy it.

GORO	**iStar**	**KAOS**	**Techne**
Stakeholder	Actor	—	—
Requirement	Goal	Goal	Goal
Plan	Task	Operation	Task
—	—	Expectation	—
—	—	Requirement	—
Assumption	—	Domain Property	Domain Assumption
decomposes	AND-Refinement	AND-Refinement	Inference
alternative	OR-Refinement	OR-Refinement	Preference
intends to operationalize	Refinement	Operationalization	Inference

Table 2. Validation: how GORO represents concepts from different GORE languages.

As mentioned in Section 2, KAOS explores the refinement of strategic goals towards low-level ones until they originate requirements. We argue that requirements are not intrinsically just a finer-grained goal. A Requirement in KAOS is a Goal under the responsibility of a single agent of the software-to-be. According to our adopted definition, a requirement is about the whole environment and, so, a requirement will not be satisfied necessarily by a software. The nature of a Goal is independent of who has the responsibility to satisfy it. In that way, the responsibility assigned to agents (software, humans or other) towards a goal satisfaction does not change a goal concept/identity and, so, we do not use two constructs to represent it as KAOS suggests (Requirement vs. Expectation). Moreover, the delegation of responsibility of goal satisfaction is not traditionally performed during the Requirements Engineering phase, but rather during architectural design. This is why GORO adopts only the Goal concept. iStar does not make this distinction either, but an expectation idea can be observed when an Actor's Goal is delegated to another one. This does not modify the goal nature but expresses the dependence relation between these two agents and an expectation from the delegator to the delegatee. These expectations also express an assumption that they will be satisfied. In GORO, this relation is associated to the assumptions-needed concept. But the delegation relation is not treated in this version of the ontology and will be subject of future work.

KAOS adopts the terms AND/OR-Refinement to express associations between Goals, allowing one to decrease the level of abstraction of strategic goals looking for more realizable subgoals [23]. However, the term Refinement is generic and has different meanings when used with prefixes AND/OR. With the former, the parent goal will be satisfied by satisfying all subgoals in the refinement, whereas the latter allows multiple alternatives of subgoal sets (each set as an AND-Refinement). Relating to GORO, we conclude that KAOS's AND-Refinement is the same as decomposes and their OR-Refinement is related to alternative.

KAOS specifies how a goal can be satisfied through Operations, which can refer to a procedure in the environment (outside the boundaries of the system) or in the software itself. The Operation concept proposed by KAOS is, in fact, a Plan to be performed to achieve a specific Goal. Therefore, the link between a Goal and an Operation in KAOS is interpreted in GORO as an inteds to operationalize.

In Techne, the Inference construct is used to relate a set of instances of a Goal, for example, to another Goal instance in the model, indicating that satisfying the conjunction of the former means satisfying the latter. This is interpreted as a decomposes link in GORO. The inference node can also be used to relate a set of Tasks to a Goal, in this case representing an intends to operationalize relation. The alternative relation can be represented in Techne using the Preference node, which expresses not only alternative sets of ways to satisfy a Goal, but also the preference amongst all these alternatives. In this context, we argue that we should be careful not to represent too many concepts in a single construct.

In earlier $i^\star$ versions, such relations used to be modeled with different constructs: AND/OR-Decomposition (representing GORO decomposes and alternative, respectively) and Means-End (GORO intends to operationalize) [14]. Currently, iStar [5] has these relations merged into one construct called Refinement. Its semantic is given by the elements related to it in the model: when amongst goals, an AND-Refinement is the same as GORO decomposes whereas an OR-Refinement is the same as GORO alternative. If a Goal is refined into Tasks, the latter is a way (means) to fulfill the former (end). In $i^\star$, it was also possible to say that a Goal is a means to a Task. GORO does not recognize this idea since a desired state of affairs in reality (a Goal) cannot be a means to act through a Plan. This relation between Goal and Task is related to other GORE concepts not discussed here, such as *contribution* [16].

As explicitly explained in the iStar 2.0 documentation [5], the adoption of a generic term referring to different kinds of relation is made to facilitate and, so, promote the adoption of the language. This simplification can be useful but we argue towards a well-founded conceptualization and, so, GORO can be used as foundation, i.e., as a domain ontology, to ground all these approaches.

5 Related Work

Several initiatives have attempted to unify or interoperate different goal modeling languages, looking for a common conceptual goal-oriented model [4,9,21,28,29].

However, their analysis is based on meta-models being essentially syntactic. In order to have a deeper understanding of the meaning of these concepts, we advocate for an ontological, well-founded analysis of the GORE domain to improve our knowledge about it and move forward towards a better formal description of the domain. The Core Ontology for Requirements Engineering (CORE) [20], on the other hand, does not cover all the concepts necessary to understand deeply the GORE domain. Also, CORE is grounded in DOLCE [26] which Guizzardi et al. [17] claim as *"not appropriate for explaining all ontological phenomena required to effectively defining and dealing with NFRs"*, for example.

In [13,14], UFO is used as foundation ontology for an analysis of the GORE domain. The authors focus their analyses on the $i^\star$ language's fundamental concepts, such as *actor*, *agent*, *means-end*, *decomposition*, *contribution links*, etc. Some of these definitions were reused in GORO, which further takes into account KAOS and Techne and explore different concepts from the domain.

6 Conclusions

In this paper, we defined GORO, a domain reference ontology about Goal-Oriented Requirements Engineering (GORE). To build it, we followed the SABiO approach and used UFO as foundational ontology. During requirements elicitation, we analyzed the literature on GORE, focusing on well-known languages, namely $i^\star$/iStar, KAOS and Techne. After capturing and formalizing the ontology, GORO was evaluated positively through verification and validation techniques. We thus believe it fits well its purpose as a common vocabulary about the GORE domain and a precise description about its concepts, successfully representing the state-of-the-art on goal-oriented requirements modeling.

As future work, we intend to: (a) extend on the validation of this ontology by using formal validation techniques (e.g., Alloy); (b) provide a full formal characterization of GORO, including eventual domain-related formal contraints; and (c) use GORO in particular implementations under the domain of Adaptive Systems, which serves as an additional form of evaluation. To that end, we plan to combine the concepts of GORO with those of RRO (Runtime Requirements Ontology) [7], derive meta-models from the combined ontologies in order to develop a new version of the *Zanshin* framework [30] for adaptive systems, now properly grounded on an ontology about RRT and GORE.

Acknowledgments

NEMO (`http://nemo.inf.ufes.br`) is currently supported by Brazilian research agencies FAPES (# 0969/2015), CNPq (# 485368/2013-7, # 461777/2014-2), and by UFES' FAP (# 6166/2015). We would like to thank Nicola Guarino and Bruno Borlini for their participation during discussions about the research contained herein and Flávia Sperandio for her distinct support.

References

1. Proc. of the 1st International Conference on Formal Ontology and Information Systems. IOS Press (1998)
2. Borgida, A., Ernst, N., Jureta, I.J., Lapouchnian, A., Liaskos, S., Mylopoulos, J.: Techne: A (nother) requirements modeling language. Computer Systems Research Group. Toronto, Canada: University of Toronto (2009)
3. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems 8(3), 203–236 (2004)
4. Cares, C.: From the i* Diversity to a Common Interoperability Framework. Phd thesis, Universitat Politècnica de Catalunya Barcelona Tech, Barcelona (2012)
5. Dalpiaz, F., Franch, X., Horkoff, J.: istar 2.0 language guide. CoRR (2016)
6. Dardenne, A., van Lamsweerde, A., Fickas, S.: Goal-directed requirements acquisition. In: Selected Papers of the Sixth International Workshop on Software Specification and Design. Elsevier Science Publishers B. V. (1993)
7. Duarte, B.B., Souza, V.E.S., Leal, A.L.d.C., Falbo, R.A., Guizzardi, G., Guizzardi, R.S.S.: Towards an Ontology of Requirements at Runtime. In: Proc. of the 9th International Conference on Formal Ontology in Information Systems. vol. 283, pp. 255–268. IOS Press (2016)
8. Falbo, R.A.: SABiO: Systematic Approach for Building Ontologies. In: Guizzardi, G., Pastor, O., Wand, Y., de Cesare, S., Gailly, F., Lycett, M., Partridge, C. (eds.) Proc. of the Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering. CEUR (2014)
9. Fayouni, A., Kavakli, E., Loucopoulos, P.: Towards a unified meta-model for goal oriented modelling. In: Proceedings of the 2015 European, Mediterranean & Middle Eastern Conference on Information Systems (2015)
10. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. Phd thesis, University of Twente, The Netherlands (2005)
11. Guizzardi, G., de Almeida Falbo, R., Guizzardi, R.S.: Grounding software domain ontologies in the unified foundational ontology (ufo): The case of the ode software process ontology. In: Proceedings of the 11th Iberoamerican Conference on Software Engineering. pp. 127–140 (2008)
12. Guizzardi, G., Wagner, G., Almeida, J.P.A., Guizzardi, R.S.S.: Towards ontological foundation for conceptual modeling: The unified foundational ontology (ufo) story. Applied Ontology 10(3–4), 259–271 (2015)
13. Guizzardi, R., Franch, X., Guizzardi, G.: Applying a foundational ontology to analyze means-end links in the i* framework. In: Sixth International Conference on Research Challenges in Information Science. pp. 1–11 (2012)
14. Guizzardi, R., Franch, X., Guizzardi, G., Wieringa, R.: Using a foundational ontology to investigate the semantics behind the concepts of the i* language. In: Castro, J., Horkhoff, J., Maiden, N., Yu, E. (eds.) 6th International i* Workshop, iStar. vol. 978, pp. 13–18. CEUR-WS.org (2013)
15. Guizzardi, R., Guizzardi, G.: Ontology-based transformation framework from tropos to aorml. Social Modeling for Requirements Engineering, Cooperative Information Systems Series, MIT Press, Boston (2010)
16. Guizzardi, R.S.S., Franch, X., Guizzardi, G., Wieringa, R.: Ontological Distinctions between Means-End and Contribution Links in the i* Framework, pp. 463–470. Springer Berlin Heidelberg (2013)

17. Guizzardi, R.S.S., Li, F.L., Borgida, A., Guizzardi, G., Horkoff, J., Mylopoulos, J.: An ontological interpretation of non-functional requirements. In: FOIS (2014)
18. Horkoff, J., Aydemir, F.B., Cardoso, E., Li, T., Maté, A., Paja, E., Salnitri, M., Mylopoulos, J., Giorgini, P.: Goal-oriented requirements engineering: A systematic literature map. In: 24th IEEE International Requirements Engineering Conference. pp. 106–115 (2016)
19. Jureta, I.J., Borgida, A., Ernst, N.A., Mylopoulos, J.: Techne: Towards a new generation of requirements modeling languages with goals, preferences, and inconsistency handling. In: 18th IEEE International Requirements Engineering Conference. pp. 115–124 (2010)
20. Jureta, I.J., Mylopoulos, J., Faulkner, S.: A core ontology for requirements. Appl. Ontol. 4(3-4), 169–244 (2009)
21. Kavakli, E.: Goal-oriented requirements engineering: A unifying framework
22. van Lamsweerde, A.: Goal-oriented requirements engineering: A guided tour. In: Proceedings of the Fifth IEEE International Symposium on Requirements Engineering. pp. 249–. IEEE Computer Society (2001)
23. van Lamsweerde, A.: From System Goals to Software Architecture, pp. 25–43. Springer Berlin Heidelberg (2003)
24. van Lamsweerde, A.: Reasoning About Alternative Requirements Options, pp. 380–397. Springer Berlin Heidelberg (2009)
25. van Lamsweerde, A., Letier, E.: From object orientation to goal orientation: A paradigm shift for requirements engineering. In: Radical Innovations of Software and System Engineering. pp. 4–8. Springer-Verlag (2003)
26. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L.: Dolce: a descriptive ontology for linguistic and cognitive engineering. WonderWeb Project, Deliverable D17 v2 1 (2003)
27. Mylopoulos, J., Chung, L., Yu, E.: From object-oriented to goal-oriented requirements analysis. Commun. ACM pp. 31–37 (1999)
28. Nwokeji, J.C., Clark, T., Barn, B.S.: A proposal for consolidated intentional modeling language. Proceedings of the Second Workshop on Graphical Modeling Language Development pp. 12–22 (2013)
29. Patrício, P., Amaral, V., Araújo, J.a., Rui, M.: Towards a unified goal-oriented language. Proceedings - International Computer Software and Applications Conference pp. 596–601 (2011)
30. Souza, V.E.S.: Requirements-based Software System Adaptation. Ph.D. thesis, University of Trento, Italy (2012)
31. Wang, X., Guarino, N., Guizzardi, G., Mylopoulos, J.: Towards an ontology of software: a requirements engineering perspective. In: Formal Ontology in Information Systems - Proceedings of the Eighth International Conference. pp. 317–329 (2014)
32. Wang, X., Mylopoulos, J., Guizzardi, G., Guarino, N.: How software changes the world: The role of assumptions. In: Tenth IEEE International Conference on Research Challenges in Information Science. pp. 1–12 (2016)
33. Yu, E.S.: Conceptual modeling: Foundations and applications. Springer-Verlag (2009)
34. Yu, E.S.K.: Modelling strategic relationships for process reengineering. Ph.D. thesis, University of Toronto (1995)
35. Zave, P., Jackson, M.: Four Dark Corners of Requirements Engineering. ACM Transactions on Software Engineering and Methodology 6(1), 1–30 (1997)