Towards Goal-oriented Requirements Engineering for Ontology Specification

Can goals replace competency questions?

Tiago Prince Sales, Ernani Gaspar, John Oliveira Guerson

Ontology and Conceptual Modeling Research Group (NEMO)
Informatics Department, Federal University of Espírito Santo (UFES), Brazil

{tpsales, johnguerson, egsantos}@inf.ufes.br

Abstract. Competency questions have been accepted by many as the key artifact in ontology specification. Nonetheless, many methods that used them lack support to the process of discovering such questions. In a previous paper Goal-oriented requirements specification in proposed as the solution to this problem. Now we elaborate on that work and discuss whether competency questions are necessary. In this paper we present a proof-of-concept that show that a GORE method can fulfill the role played by competency questions in ontology specifications and that in fact, it is a more appropriate solution.

Keywords: Ontology-driven conceptual modeling, Goal-oriented requirements engineering, competency questions

1 Introduction

Proposed throughout the emergence of ontology in the computer science, competency questions are still considered by many as a key artifact in ontology specification. The importance given to CQs comes from what they were designed to provide: the ontology scope. The scope defines which concepts and properties one would have to model and thus, is one of the main requirements to start developing an ontology.

A shortcoming in the elicitation of competency questions was pointed out in [9]. In this paper, the authors argue that many CQ-centered ontology specification methods, such as NeOn [16] and TOVE [11], fall short in providing methods and tools for this elicitation activity. This limitation leaves modelers to define CQs in an ad-hoc manner.

The lack of guidance in competency questions elicitation compromises the overall result of ontology development, since requirements are not clearly identified. A systematic method for ontology specification is of great importance for making ontology development indeed an engineering activity. By providing well defined methods and tools, modelers can expect a certain level of quality in their final product, i.e. the ontology, and the activity ceases to be a "black art".

To provide this necessary systematic approach, in [9] the authors propose the derivation of competency questions from goal models. The proposal was made using Tropos [5], a framework for representing goal-oriented requirements models, which provides a method, and a modeling language, for goal modeling. With the support of such technique, modelers are given a starting point and a method to obtain the competency questions.

In a previous work [2] we performed an empirical study to assess if the method proposed in [9] indeed extinguished the semantic gap between requirements and competency questions. The analysis showed that, although reduced, the gap was still there and the competency questions were not as systematically generated as expected.

In this paper we show that this gap is motivated by the lack of goal refinement proposed in [9] and that when refined thoroughly, goals and competency questions are redundant. We argue that the role of "scope definers" played by competency questions can naturally be played by goals if the ontology specification is performed as Goal-oriented Requirements Engineering (GORE).

By making this analysis we show that there is no necessary need to combine goals and competency questions, and that ontologies can be generated directly from goal models. In fact, we argue that most GORE methods are more suitable for ontology specification than CQ-centered ones.

The remainder of this paper is organized as follow: Section 2 provides the background in GORE, competency questions and the previous initiatives of combining GORE and CQs. Section 3 discusses the relation of goals and competency questions. Section 4 presents the proof-of-concept, describing the method, the models used to compare both techniques and the results obtained. Section 5 finalizes with some overall conclusions.

2 Background

In this section we discuss conceptual and methodological references to both, Requirements Engineering (RE) as an activity of Software Engineering (SE) and to Ontology Requirements as an activity of Ontology Engineering.

2.1 Requirements Engineering

In SE, requirements are descriptions of services and constraints of a system. The activity of identifying, defining, analyzing, documenting and validating them, is named RE [18]. RE concerns are the objectives of the real world [22], that is, observing environment factors, where the software will operate [15, 22]. RE activity expose designers to real world concepts in the software operating environment, bringing value to its users. Requirements are statements about the stakeholders' needs and about the environment, expressed by specific views of each person involved, which can be conflicting. These needs can also be restricted, by economic, technical or technological constraints. All this make RE complex. Delays in deliveries, overspending, and dissatisfaction of users and customers due to a product difficult to use and expensive to maintain, are problems that have a common underlying reasons. These reasons are related to poorly specified requirements [13]. Furthermore as they are used in subsequent stages of development, problems with requirements may jeopardize the entire project.

RE has been around since the end of 70's, when it was officially understood as a subarea of SE. Nevertheless, it started to effectively develop in later decades. In the mid-80s, Greenspan [10] presents a language-based knowledge representation for requirements, RML (Requirements Modeling Language), being the forerunner of object-oriented programming languages. Since then, we have been witnessing the appearance of a number of evolutionary lines, such as, goal orientation, aspects orientation, and use cases driven approaches. This research addresses specifically goal-oriented approaches to RE, such as, KAOS [7, 8], *i** [20, 21], Tropos [5, 19] and URN [1].

2.2 Goal-oriented RE

Goal-oriented RE refers to using goals as an abstraction for accessing requirements life-cycle. A goal can be explained as a prescriptive statement of intent, satisfied through cooperation of agents [14]. Another definition for goals is that they are stakeholders' intended states of the world [5]. These definitions complement one another. Domain descriptions, on the other hand, are descriptive statements that hold in the world, independently of the software system. An agent is an active part of a domain playing a specific role in goal satisfaction. Goal satisfaction is actually achieved by the collaboration between agents.

We can express goal in different levels of granularity. They can be stated in a high level of abstraction, e.g. enterprise level, such as "improve customer satisfaction". Also, goals can be state as a technical concern, like: "the valve [of a machine] must shut if the temperature goes higher than 600 degrees". These different levels suggest some structuring like classification-decomposing characteristic of goals [3, 14, 17].

A type of relation between goals is decomposition. It is addressed in two ways: "AND" decomposition and "OR" decomposition. The former takes a high level of abstraction goal and take it to lower levels, implying a chain of states that shall hold. The latter shows possible alternative intended states (not mandatory "XOR").

There are different approaches to contribution among goals, or more generally, intentional elements. Tropos and *i** take into account this kind on analysis, although KAOS passes, but provides a conflict analysis to deal with goal obstacles and conflicts. Put in another way, in KAOS contributions are mereological, and contributions are treated only in the negative sense, as a conflict (as an impossibility for goal satisfaction). The others treat contributions as positive and negative, not presupposing impossibility of a state to hold; at least from the start, since certain contribution links in Tropos do imply unsatisfiability [19].

Goal-oriented methods and frameworks have been successfully applied in a number of projects in SE. They have formal implementations, that is, an engineering approach to requirements. We decided, in this part of the research, to use Tropos, but using the others seems to be not detrimental. In fact, we plan to exercise KAOS framework, since it is a different line of evolution, in order to investigate whether its other models, that compose the requirement specification, may contribute to ontology specification.

2.3 Ontology Specification Methods

In this section we present methods and tools for ontology specification proposed in the TOVE [11] and NeOn [6] methodologies. We chose the former because it is the first effort for ontology engineering and because it is where competency questions were first conceived. The latter was chosen because it is currently one of the most well-known and accepted methods and can be understood as an evolution of the TOVE's proposal.

It is important to emphasize that both methods were developed with the purpose of developing ontologies in the sense of logical models that should support automated reasoning. We on the other hand, are concerned with ontologies in the sense of ontology-driven conceptual models, which has as a main concern expressivity. In this latter sense, ontologies are supposed to support communication between human agents.

2.3.1 TOVE's ontology specification

This section presents part of the TOVE methodology for ontology development that is concerned with ontology specification. It is heavily based on [REF] and thus quotation marks will be omitted.

In TOVE's method, ontology specification consists of two main activities: defining the purpose and the scope of the ontology. On one hand, the defining the purpose regards understanding why the ontology is being built, how is it going to be used and who is going to use it. On the other hand, to determine the scope of the ontology means defining which concepts are going to be modeled, along with which relationships and properties of such concepts.

To identify the purpose of the ontology, the authors propose the specification of motivating scenarios, which are defined as follows:

Definition 1 - Motivating Scenario: "*The motivating scenarios are story problems or examples which are not adequately addressed by existing ontologies. A motivation scenario also provides a set of intuitively solutions to the scenario problems.*

From this definition, we see that these scenarios are the drivers for the ontology development, i.e. the starting point of the whole method. By describing the problem the organization faces and for which the ontology is part of the solution, the authors argue that one can identify the uses, users and needs of the organization. Note that, although the intention of the artifact is clearly stated, the authors don't propose a specific format, neither a method for writing these scenarios.

After capturing the purpose of the ontology, through the motivating scenarios, the method proposes the identification of the scope. The artifact to capture it is a set of competency questions, whose definition follows:

Definition 2 - Competency Question: "*We can consider these queries [competency questions] to be expressiveness requirements that are in the form of questions. An ontology must be able to represent these questions using its terminology, and be able to characterize the answer to these questions using the axioms and definitions*".

Given such definition, we can thus understand competency questions as informational demands (or services) the ontology-to-be is supposed to provide. Also, it states that the constraints and definitions embedded in the ontology will provide the right answer to such questions.

Although the authors do not propose any systematic method for specify competency questions, they do argue that they should be written in a stratified manner, where complex questions are decomposed until a look-up level is obtained. Also, note that the only restriction regarding the sort of questions that should be specified is that they must arise from the motivating scenario. With both the motivating scenarios and the competency questions at hand, the method presupposes that the modelers have sufficient resources to start the capturing and design of the ontology.

2.3.2 NeOn's ontology specification

The NeOn method defines ontology specification as the activity of gathering the requirements the ontology-to-be should fulfill. They use the term requirement in a rather loose way, which encompasses the reason the ontology is being built, its users and uses and competency questions. Nonetheless, it defines as the output of the ontology specification activity a document named Ontology Requirements Specification Document (ORSD), whose contents will be presented in the following.

The method proposes that the ORSD should contain: purpose, scope, level of formality, intended users, intended uses, groups of competency questions (with the same definition of TOVE's) and a pre-glossary of terms. Although the ORSD has many components, the CQs still play the central role in the ontology specification. Although the NeOn holds a great similarity with TOVE, the method is a lot more structured, by providing methods for requirements identification and ORSD overall development. Taking a closer look, one can consider NeOn as an evolution of TOVE to which regard ontology specification. The ORSD proposed in NeOn can be seen as structured motivating scenarios and competency questions, since it mainly proposes the identification of the same concerns.

2.4 Previous initiative

Our approach takes into account a number of considerations made in Fernandes et al. [9]. Their paper presents an approach to represent CQs in goal models and derive the ontology as a well-founded ontology conceptual model and use it in the development of an information system [12]. The authors explore Tropos approach to RE in both phases, unlike ours, which uses only early requirements. Moreover, our objective here is not information systems development, but contribute to the Ontology Requirements Specification Document (ORSD), as Del Carmen Suarez-Figueroa et al. [6] name the product of Ontology Specification activity.

3 Goals and Competency Questions

The objective of this section is to theoretically argue, based on the definitions of the previous section, why the CQ-centered ontology specification methods (e.g. TOVE [11] and NeOn [6]) can be replace by state of the art goal modeling (e.g. Tropos [5], KAOS [5], *i** [21]).

Our analysis begins in the starting point of ontology specification methods: the understanding of the context in which the ontology is being built. In TOVE this is captured through the motivating scenarios and in NeOn, through the users, uses, purpose and basically everything that is not a CQ. Note that both approaches propose that CQs must be derived from the organization needs, which mean that, prior to developing the ontology, one must to a satisfactory extent, understand the problem at hand, which is related and their goals in that given context. Only after understanding all this, one is eligibly capable to specify any solution at all; in this case, define the requirements of the ontology to be designed.

Thinking about this initial activity of identifying the purpose of why the ontology is being built, we can see that the NeOn method mostly structures the TOVE's proposal of motivating scenarios, by describing which information must be specified. But taking a closer look, we can see that GORE frameworks can capture all this information. First, the users of the ontology are actors (or stakeholders, or agents) in GORE frameworks. The reason the ontology is being built is to satisfy users' necessities, which can be captured through the goal concept. Uses for the ontology can be capture by refinement using the task (or plan) concept.

By making this assessment of the concepts that need to be captured in the initial phase of ontology specification, theoretically GORE can be an alternative to any of the other ontology specification approaches. But we go further, it is not only an alternative, it is much more adequate, since the frameworks provide modeling languages and methods to perform this activity. This methodological support guides modelers throughout the process and improve the quality of the specification. This argument is in line with [9], and aids in the consolidation of their proposal.

The main focus of our analysis is, nonetheless, whether GORE frameworks make the derivation of CQs unnecessary. To answer that we must come back to the CQ definition: *questions the ontology must answer*. If the ontology must answer it, it's because somebody may need/want to ask it eventually. It is safe to assume that there is at least one activity to be performed by some actor, which requires retrieving or storing that piece of information. If we go back to goal modeling, we see that the frameworks allow the representation of such needs/activities, both, information related or not. In addition, the GORE frameworks allow the description of who uses that information and what it is used for. If CQs are to be used to determine the scope, i.e. the concepts and their properties to be modeled, we can obtain the same input from goal modeling, especially if they are to be refined to the leaf level. So, if CQs represent these activities (or needs) to store or retrieve information, goals are well suited for the purpose as well.

We can also see this similarity between goals and CQs if we think of how people propose to capture them: through decomposition or refinement. Both goals and CQs are initially represented in a more general view and later are refined until an atomic level is achieved. From this perspective, goals provide more resource to guide this decomposition, since one can use decomposition, and contributions and means-end relationships with improved semantics.

Overall, GORE provides more structured constructs and methods for requirement specifications. Our point is that ontology specification methods are in fact rudimentary RE methods, which leave to ontology engineers the responsibility to make a lot of assumptions and decisions, without appropriate methods and tools.

4 Proof of Concept

4.1 Method

The presented analysis is conducted during the execution of the tasks of a real industrial project to develop domain ontology to support interoperability in a public organization. For this reason, we set up a prototype process for ontology specification based in literature review and interviews, which is shown in Figure 1. We conduct a legal, normative (organizational) and general literature review to grasp the organization intents, needs, collaboration and dependency issues, as well as its structure and delegation of goals, in order to shape the first goal models. Interviews adjusted these first designs, taking into account what the organization's units actually practice in their day-to-day activities.

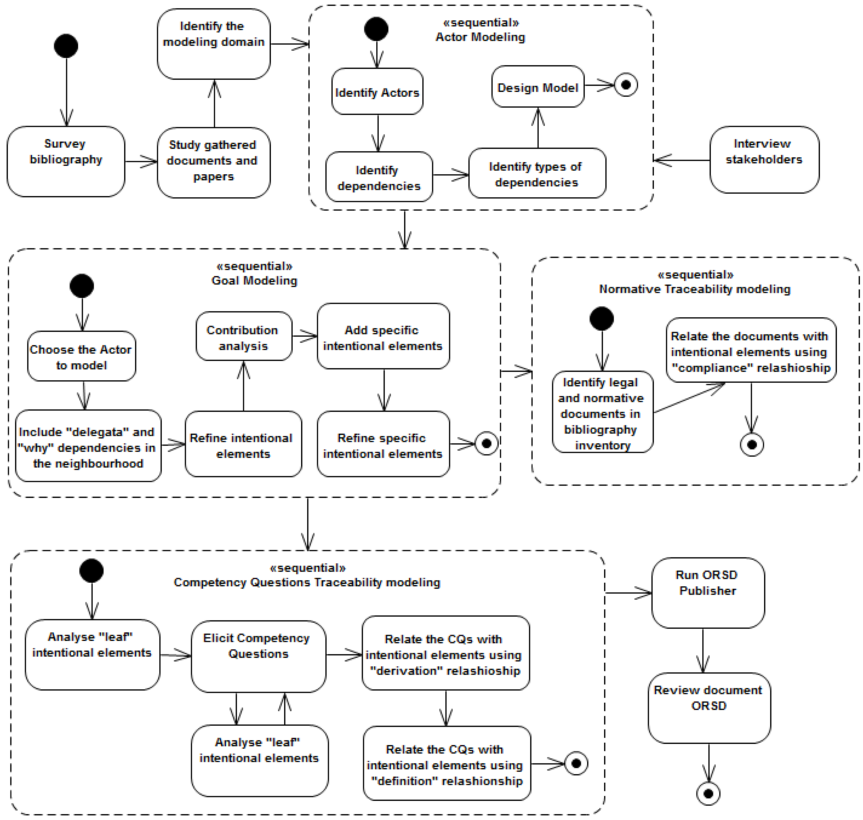

Figure 1 - Ontology specification prototype process.

The goal models, captured in this part of the overall design activity, give us a way to map legal and normative reference to goals, using a diagram that we propose to extend Tropos set of tools, namely, Normative Traceability diagram (NT). This diagram keeps track of the motives certain goals and their relationships exist in the model using the *compliance* association. Further, using those models as raw material, we capture the CQs analyzing the designed goals and trying to set up questions that would make possible to satisfy them. We propose to commit these findings visually with a diagram: the Competency Questions Traceability Diagram (CQT). CQs *derives* from goals and are *defined* in *generic* or *normative* (including legal) *documents*. Examples of both diagrams, using the subdomain illustrating this paper, are shown in the next subsection. Preparing and reviewing the specification document is the last part of the prototype process of Figure 1.

4.2 Example

We present in this paper an example of a Brazilian regulatory agency in the ground transportation sector. It is a fragment of the domain of passengers' charter road transportation. This domain is about passengers' ground bus transportation in charter regime, that is, by trip or by a specific path or trajectory, periodically, from one end-point to another. We call the first category eventual charter. It may be a charter trip to a basketball game in another town, or a concert. However, if the trip has touristic ends, it is called touristic charter. We call the second category, continuous charter. Examples of this kind of transportation are that of students from home to school every day, or of workers going to their place of work. The fragment we present is about both categories but, within this context, there are no differences, except for the fact that the databases differentiate the services provisions.

We use, as discussed in subsection 2.2, Tropos to perform the early requirements analysis we present in the paper; it means using *i**/Tropos visual representation language, and the artifacts it prescribes to this phase. Figure 2 shows an intermediary abstraction level goal diagram fragment for the charter transport regulation business. This fragment is about only one goal of the operational unit delegated for the charter category; it is about monitoring the provision of passengers' transport services in charter regime. There are others, such as, the preparation of habilitation processes and their granting, and the preparation of authorization contracts and their granting. Further, the fragment deals mostly with database maintenance, or capturing and maintaining all the data needed to successfully execute the operational unit duty (GEFAE – Charter Authorization Department).

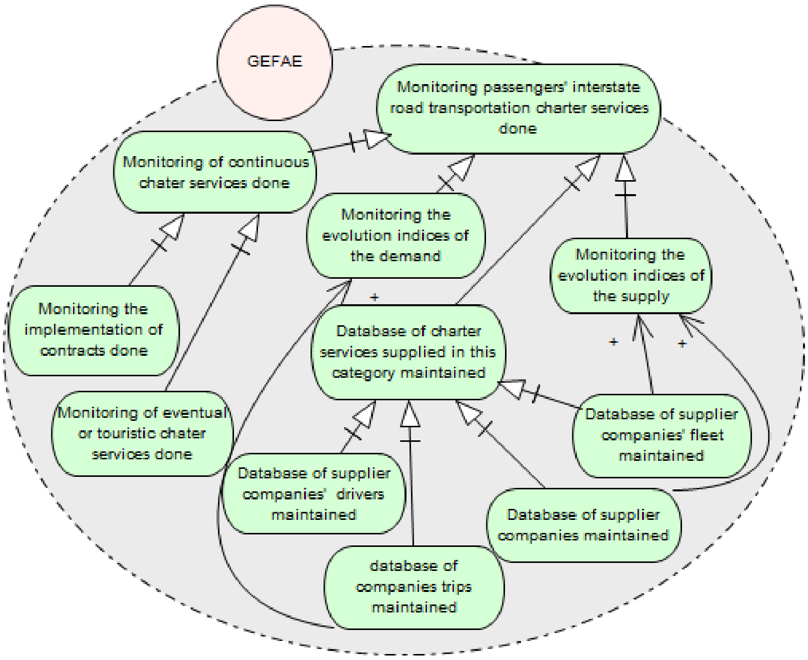

Figure 2 - Goal diagram for monitoring charter passenger's road transportation.

In the diagram of Figure 2 we have the representations of hard goals *decompositions* and positive (*plus*) *contributions*. Monitoring continuous charter services have "AND" decompositions, meaning that to monitor these services as a whole they must satisfy each of the monitoring goals. We make the decision to leave out the direct contributions from database related goals to some goals, since these entailments are already holding due to decomposition. The semantics of contributions are important to indices computations, as shown in the diagram.

Going downwards in this diagram we can read what must satisfied, that is refinements mean what. However, if we go upwards we read the reasons why the goals satisfaction is there for. Although the reading of this diagram gives us a lot of knowledge about stakeholders' intentions, it does not tell us much about the concepts the represented goals deal with. Actually, we can derive CQs from this diagram; we illustrate this in the CQT diagram shown in Figure 3. However, we can see some problems with this decision. Although the questions do make sense, in a number of them we perceive a gap, or that we are jumping over something, or assuming things that the goals, from which they are derived, do not convey. We are assuming that a "fleet" is of buses, that there is a concept which tell us that a "driver" works at a company, at a given date, or period. This, we name the semantic gap from CQs to goals from which they are derived.

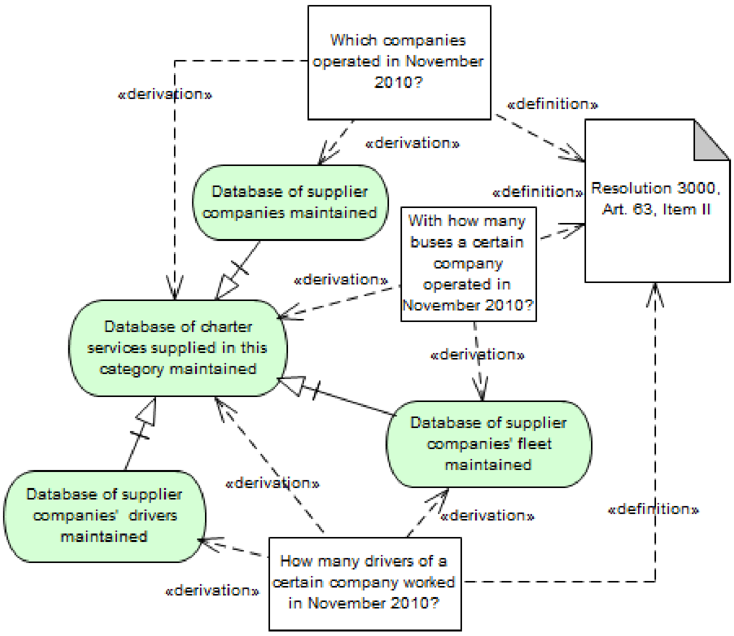

Figure 3 - Competency Questions Traceability diagram for goals related to database issues in passenger's charter road transportation domain.

To answer to this questions, or why this semantic gap occurs, we recurred to the very reason that goals are refined in RE, that is, to state intentions at certain level that allow us to model software information entities, besides capturing the behavior elements and environmental characteristics. Once we figured that out, we could draw the diagram presented in Figure 4, illustrating a more detailed refinement of two database related goals of Figure 2. In this diagram we can see the concepts in a clearer form, sometimes directly stated. We can now redraw the diagram of Figure 3 using these new found goal refinements and take the gap to minimum or none, as presented in Figure 5.

We deliberately left in the diagram of Figure 5 goals without detailed refinements to explicitly address the problem in one illustration. For example, the problem with the concepts "fleet" and "bus" is still present, because we have not sufficiently refined this (right-most question). Otherwise, the diagram presents that the instance counting of the top-most question is more complicate than it seemed to be. Now it shows that "operate" presupposes an authorization for determined operated services, and this presupposes a date when they actually happened. We also show a mixed case, about "drivers". We intentionally left out the refinement of goal stating about "drivers", "buses" and "fleet", but refined the goal that explicitly tell us that transportation companies have employees.

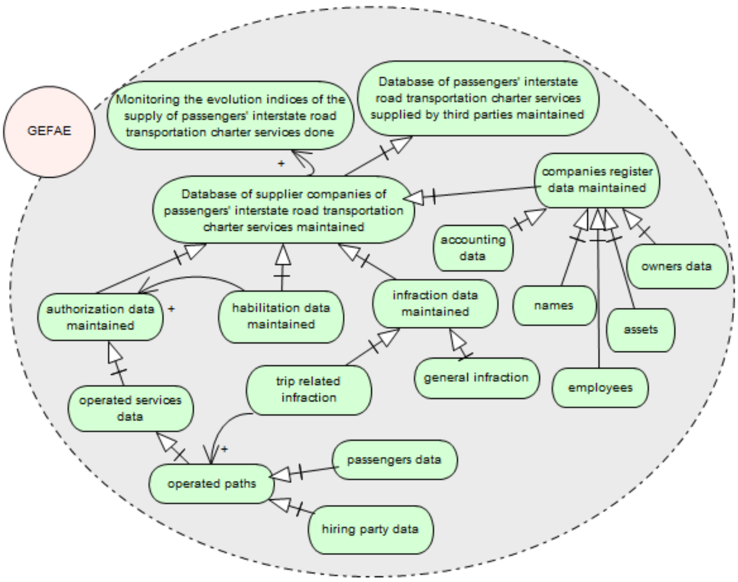

Figure 4 - Goal diagram detailed refinement for database (data registers) related goals for monitoring chater passenger's road transportation.

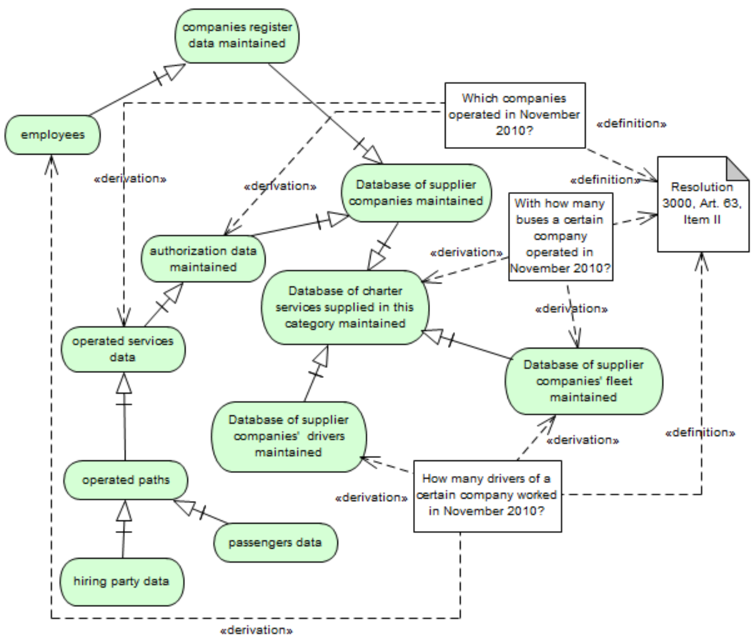

Figure 5 – CQT showing derivation from detailed goals and other the old ones, which maintain the sematic gap.

Although, in this case, we are still presupposing things, we are able to tell one more thing: that drives are a company's employee. We have diminished a bit the gap, which will be solved whether we refine those specific goals.

4.3 Discussion

No matter what is the genesis of competency questions, what we understand about the exercise we have done is that when goals were not enough refined, the questions we make have a semantic gap, which translates in assumptions we must make to drive to concepts in the ontology. These assumptions vary in accuracy in direct proportion to the ontology engineer experience and proficiency (this is a reasonable conclusion, once she takes the responsibility to derive questions that explicitly involve concepts that in most cases were not even mentioned in the literature or in interviews). Whether we refine goal to the leaf state, we can explicitly have the concepts we need.

These facts make us think about a parallel that, in principle has absolutely nothing to do with our investigation, which is Goal, Question, Metric (GQM) paradigm [4]. We do not want to dive too deep here about this, but, in this approach, goals, questions and metrics do present a semantic gap, fulfilled by a set of rules and templates to fill in the blanks, and (not or) by software engineers' expertise and proficiency. The instrumentation to "fill in the blanks" seems to be met with goal refinements, in our point of view.

We can conclude from the exercises of our proof-of-concept conducted during the ontology specification task of a Brazilian ground transportation regulatory agency, using our prototype process is that refining goal to leaf state reveals ontology concepts (informational content), as it is expected (one of the reasons of RE: elicit entities, besides behavior and environment characteristics).

We can also conclude that, although we argue in favor of goals in comparison with CQs, they presented common limitations. Both approaches do not help modelers to reveal domain rules and ontological distinctions. These are additional necessities to scope definition for ontology design, since they will contribute for the ontology to correctly answer the informational questions..

5 Conclusion

We have been digging more, in depth and breadth, with hypotheses involving the uses of goal-oriented approaches to ontology specification starting with [9]. We noticed that there was a gap about the concepts captured from the derivation of goals to CQs. We have named this gap, the semantic gap. Starting to pursue the reasons for this, we realize that lack of enough refinement accounts for the problem, but, almost as a consequence, we figured out that doing so, goals overlap with CQs themselves. In other words, goals refined to the leaf level convey the same raw material as CQs: concepts, properties and relations for modeling the ontology.

This work is not finished, but at this time we can relate some important contributions to ontology engineering, giving ways to the formalization of a sound ontology specification process, namely:

- Presented sound, although preliminary, results of using a well-known and successful RE process that drives to CQs with minimum or non-existent semantic gap;
- Presented an alternative way to produce ontology specification, that is, performing ontology conceptual modeling direct from leaf goals captured during the early requirements; and
- Contributed to a better understanding of cross-fertilization issues between SE and OE.

We are aware that some other issues wait for us. One of them is to consider the proposal made in [9] about performing late requirements analysis in Tropos, taking the ontology itself as an agent as the method prescribes. This hypothesis in contradicted by other that states that epistemological needs, or entities, properties and relationships, inheres to business needs. They are actually embedded in goal modeling, with no need for a machine or software to be explicitly represented.

We have not solved two problems in this research: eliciting domain rules and ontological distinctions. The first is possibly solved using KAOS' notion of domain properties and domain hypothesis. Nothing yet is devised for ontological distinctions. Perhaps going in the same direction as for domain rules, may be a preliminary choice.

References

[1] Amyot, D. 2003. Introduction to the user requirements notation: learning by example. *Comput. Netw.* 42, 3 (Jun. 2003), 285–301.

[2] Azevedo, C. et al. 2013. *Towards a Goal-oriented Approach to Ontology Specification*.

[3] Baader, F. and Nutt, W. 2003. The description logic handbook. F. Baader et al., eds. Cambridge University Press. 43–95.

[4] Basili, V.R. 1992. *Software modeling and measurement: the Goal/Question/Metric paradigm*. University of Maryland at College Park.

[5] Bresciani, P. et al. 2004. TROPOS: An Agent-Oriented Software Development Methodology. *Autonomous Agents and Multi-Agent Systems*. 8, 3 (May. 2004), 203–236.

[6] Del Carmen Suárez-Figueroa, M. et al. 2008. D5.4.1 NeOn Methodology for Building Contextualized Ontology Networks.
[7] Dardenne, A. et al. 1993. Goal-Directed Requirements Acquisition. *Sci. Comput. Program.* 20, 1-2 (1993), 3–50.
[8] Darimont, R. et al. 1997. GRAIL/KAOS: an environment for goal-driven requirements engineering. *Proceedings of the 19th international conference on Software engineering* (New York, NY, USA, 1997), 612–613.
[9] Fernandes, P.C.B. et al. 2011. Using Goal Modeling to Capture Competency Questions in Ontology-based Systems. *JIDM*. 2, 3 (2011), 527–540.
[10] Greenspan, S.J. et al. 1994. On Formal Requirements Modeling Languages: RML Revisited. *ICSE* (1994), 135–147.
[11] Grüninger, M. and Fox, M.S. 1995. Methodology for the Design and Evaluation of Ontologies. (1995).
[12] Guarino, N. 1998. *Formal Ontology in Information Systems: Proceedings of the 1st International Conference June 6-8, 1998, Trento, Italy*. IOS Press.
[13] Kotonya, G. and Sommerville, I. 1998. *Requirements Engineering - Processes and Techniques*. John Wiley & Sons.
[14] Lamsweerde, A. Van 2007. *Requirements Engineering*. John Wiley & Sons.
[15] Nuseibeh, B. and Easterbrook, S.M. 2000. Requirements engineering: a roadmap. *ICSE - Future of SE Track* (2000), 35–46.
[16] Pérez, A.G. et al. 2008. NeOn Methodology for Building Ontology Networks : Ontology Specification. February (2008), 1–18.
[17] Polya, G. 1945. *How To Solve It*. Princeton University Press.
[18] Sommerville, I. 1995. *Software engineering (5th ed.)*. Addison Wesley Longman Publishing Co., Inc.
[19] Susi, A. et al. 2005. The Tropos Metamodel and its Use. *Informatica (Slovenia)*. 29, 4 (2005), 401–408.
[20] Yu, E. 2011. Modelling strategic relationships for process reengineering. *Social Modeling for Requirements Engineering*. 11, (2011).
[21] Yu, E. 1997. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering. *RE "97: Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE"97)* (Washington, DC, USA, 1997), 226–235.
[22] Zave, P. and Jackson, M. 1997. Four dark corners of requirements engineering. *ACM Trans. Softw. Eng. Methodol.* 6, 1 (Jan. 1997), 1–30.