A process for building a domain ontology: an experience in developing a government budgetary ontology

3 authors:

Graciela Brusa
National University of Technology, Santa Fe, Argentina
6 PUBLICATIONS 133 CITATIONS

María Laura Caliusco
National University of Technology
64 PUBLICATIONS 509 CITATIONS

Omar Chiotti
National Scientific and Technical Research Council
123 PUBLICATIONS 1,015 CITATIONS

A Process for Building a Domain Ontology: an Experience in Developing a Government Budgetary Ontology

Graciela Brusa

Dirección Provincial de Informática
San Martín 2466, Santa Fe (Santa Fe)
Argentina

gracielabrusa@santafe.gov.ar

Ma. Laura Caliusco

CIDISI, CONICET-UTN-FRSF,
Lavaise 610, Santa Fe (Santa Fe)
Argentina

mcaliusc@frsf.utn.edu.ar

Omar Chiotti

INGAR, CONICET-UTN-FRSF,
Avellaneda 3657, Santa Fe (Santa Fe)
Argentina

chiotti@ceride.gov.ar

Abstract

During the last years, there has been a growing concern on ontology due to its ability to explicitly describe data semantics in a common way, independently of data source characteristics, providing a schema that allows data interchanging among heterogeneous information systems and users. Several works have been aimed to improve ontology technological aspects, like representation languages and inference mechanisms, and less attention has been paid to practical results of development method application. This paper presents a discussion on the process and product of an experience in developing ontology for the Public Sector whose organization requires a strong knowledge managment. Particularly, this process was applied to develop ontology for Budget Domain.

Keywords: ontology engineering, development methodology.

1 Introduction

Since an ontology has gained recognition from academy and industry, there are several definitions about what an ontology is. These definitions came from different disciplines and have been used for different purposes. In information science, an ontology can be seen as a dictionary of terms formulated in a canonical syntax and with commonly accepted definitions designed to yield a lexical or taxonomical framework for knowledge-representation which can be shared by different information systems communities (Smith, 2003). In order to define a complete commonly accepted definition, an agreement must be reached. This agreement has to follow a comprehensive ontology engineering process.

There are several mature methodologies that have been proposed to structure this process and thus to facilitate it. Moreover, the success of these methodologies has been demonstrated in a number of applications (Corcho et. al, 2005). Nevertheless, the ontology development in some areas has not been as expected. One example is the public sector area, which is characterized by a wide range of task and work arrangements. Some process can be fully automated but its scope is limited to simple processes of registering, accounting and calculating Processes in which stakeholders participate because legal rules and knowledge play an important role are much more important (Klichewski, 2002).

Besides, decision-making in public administration occurs at organizational or policy level but it is also characteristic of its operative work. Thereby, public agents must be able to access information and knowledge to help their tasks.

The objective of this paper is to share with the ontology community the process followed to develop an ontology for Budgetary Domain. To this aim, this work is organized as follows: Section 2 discusses the main ontology development methodologies. Section 3 shows how we have adapted different methologies to define the budgetary domain ontology. Section 4 discusses the implementation of the ontology. Section 5 presents a summary of results. Finally, Section 6 is devoted to the conclusions of this work.

2 Ontology Development Methodologies

Before starting to define the ontology, different development methodologies were studied (Wache et al, 2001). From this study, two main groups can be identified. On the one hand, there are experience-based methodologies, such as the methodology proposed for Grüninger and Fox (1995), based on TOVE Project or the other exposed by Uschold and King (1996) (Uschold & Gruninger, 1996) from Enterprise Model. Both were issued in 1995 and belong to the enterprise modeler domain. On the other hand, there are methodologies that propose evolutive prototypes models, such us METHONTOLOGY (Gómez-Pérez et al, 2004) that proposes a set of activities to develop ontologies based on its life cycle and the prototype refinement; and 101 Method (Noy & McGuinness, 2001) that proposes an iterative approach to ontology development.

On the one hand, there is not just one correct way or methodology for developing ontologies. Usually, the first ones are applied when the requirements are clearly known at the begining; the second ones when the objectives are not clear from the begining. Moreover, it is common to merge different methodologies since each of them provides design ideas that distinguish it from the others . This merging depends on the ontology users and ontology goals.

On the other hand, like any other conceptual modeling activity, ontology construction must be supported by software engineering techniques (Falbo, 2004). Thus, we used methods and tools from software engineering to support ontology engineering activities.

In general terms, the ontology development can be divided into two main phases: specification and conceptualization. The goal of the specification phase is to acquire informal knowledge about the domain. The goal of the conceptualization phase is to organize and structure this knowledge using external representations that are independent of the implementation languages and environments. The objective of the next section is to show how we have adapted different ontology development methodologies to define the specification and conceptualization phases. Furthermore, it shows how different software engineering techniques were used to define different representations during these phases.

3 Building Government Ontology for Budgetary Domain

This section describes the process of an experience in developing a Government Ontology for Budgetary Domain.

3.1 Specification: The Ontology Goal and Scope

The definition of ontology goal and scope was considered the first step in this study case as it is proposed in 101 Method and in the first activity in METHONTOLOGY.

The scope limits the ontology, specifying what must be included and what must not. It is an important step for minimizing the amount of data and concepts to be analyzed, especially for the extent and complexity of the budgetary semantics. In successive iterations for verification process, it will be adjusted if necessary.

This ontology only considers the needs for creating an analytic budget with concepts related to expenses. It does not consider the concepts related to other stages as budgetary executing, accounting, payments, purchases or fiscal year closure. Therefore, it includes general concepts for the budget life cycle and specific concepts for the formulation.

3.2 Specification: Domain Description

Taking into account that this work was made from scratch and that 101 METHOD proposes the enumeration of important terms to continue as well as METHONTOLOGY plans to use intermediate representations for organizing knowledge domain in the conceptualization phase (Gómez-Pérez et al, 2004), it was necessary to make a previous domain analysis.

In this analysis, the application to formulating the provincial budget and its related documentations were studied and revised. Furthermore, meetings with a group of experts were carried out. This group was conformed by public officials responsible for the whole budget formulation process in the Executive Power, expert professionals of Budget Committee in Legislative Power, public agents of the administrative area in charge of creating their own budget, and software engineers who bring informatics supports for these tasks. As it can be seen, the group of experts was very heterogeneous. In addition, they do not have much time to assign the meetings. This group was the support for knowledge acquisition during the ontology development. Then, we have to define different intermediate representations to communicate the knowledge adquired to the experts considering the background of each one and the time of meetings.

Following, a brief description of the domain is presented.

3.2.1 Budgetary and Financial Domain

The budget of a government is a plan of the intended revenues and expenditures of that government. The budget is prepared by different entities in different government areas. Particularly, in Santa Fe Province (Argentine) these entities are:

- Executive Power: this government entity elaborates the Provincial Budget Draft. It is constituted by a Rector Organism (governing body) and several Executor Organisms. The first one define all activities for formulating a budget and the others execute these activities.
- Legislative Power: this government entity passes the Annual Budget Law.

Along with the budget life cycle the evaluation and control of actual and financial resources is made, and all of them are assigned to goods and services production. Table 1 shows the steps in detail.

1. Initiate Fiscal Year and Distribute Classifiers
2. Prepare Preliminary Budget and Resources Estimation
3. Define Budgetary Policy and Expenses Projection
4. Determine Expenses Top
5. Formulate Budget Project Draft
6. Present Budget Project Draft to Legislature
7. Approve Budget in Legislature
8. Elaborate new budget according to Budget Law
9. Distribute Budget for executing
10. Elaborate Budgetary Modifications
11. Program Budget executing
12. Reconduct Budget
13. Closure Fiscal Year

Table 1: Budget Life Cycle Steps

There is common information for all budget life cycle stages: Expense and Resource Classifiers. The classifiers used in this work are: Institutional, Expense Object, Geographic Locate, Finality Function, Resource Item, Financing Source, and Programmatic Categories.

There are two situations where the availability of semantic information associated to budgetary data is critical: budget formulation and approval tasks. In the first case, only government staff with specific knowledge can be involved, concentrating a great responsibility on a few people. In the second case, semantics information is necessary for analyzing budgetary data and then having the budget law passed. Here, this is more complex because all legislators must vote and most of them have no specific knowledge. For simplicity purposes, only the Formulation stage for expenses budget was considered for this study case.

3.3 Specification: Motivating Scenarios and Competence Questions

We included this step taking into account the opinion of Gruninger and Fox (1995). The authors consider that for modeling ontologies, it is necessary to count on informal logic knowledge model in addition to requirements resulting from different scenarios. The motivation scenarios show problems that arise when people need information that the system does not provide. Besides, the scenario description contains a set of solutions to these problems that includes the semantic aspects to solve them. In order to define motivation scenarios and communicate them to the involved people, templates have been used. These templates were based on those proposed to specify case uses in object oriented methodology (Uschold & Gruninger, 1996). An example is shown in Table 2. The template describes: the name of the scenario, people who participate in the scenario, a brief scenario description, and a list of possible terms related to the scenario. Since this template shows the most important information in a concise way, it is useful when the experts do not have a lot of time to analyze the scenarios.

Scenario: Local Budget Formulation.
Actors: Participants of the budget formulation for next year.
Description: The scenario proposed here is a person who must participate in the budget formulation task for the next year. This task is carried out along the previous year because it is necessary to have the budget approved before the next year begins. Executor organisms of each government jurisdiction make their own formulation task. The Rector Organism defines policies conducting the budget draft elaboration as well as the main expenses and resources classifiers for the year. Then, each organism elaborates its jurisdictional budget draft.
Terms: budgetary classifier, expense and resource classifier, Institutional, Programmatic Category, Geographic, Expenses Object, Financing Source and Finality Function Classifiers, among others, for working on the budget draft .

Table 2: Scenario Description

Competency questions proceed from motivation scenarios. This allows deciding the ontology scope to verify if it contains enough information to answer these questions and to specify the detail level required for the responses. Besides, it defines expressivity requirements for the ontology because it must be able to give answers using its own terms, axioms and definitions. The scope must define all the knowledge that should be in the ontology as well as those that should not. It means that a concept must not be included if there is not a competency question that uses it. This rule is also used to determine wether an axiom must be included in the ontology or not.

Moreover, competency questions allow defining a hierarchy so that an answer to a question may also reply to others with a more general scope by means of composition and decomposition processes. As an example, some of them are shown in Table 3.

Simple Questions
Which are the budget states?
Which are the budgetary classifiers?
Which are the expenses classifiers?
Which are the resources classifiers?
Which are the executor organisms for Health Minister?
Which are the Health Minister Programs?
Complex Questions
Which is the institutional code for the Education Minister?
Which are the sector and subsector for Central Administration?
Which is the character code for “Decentralized Organism”?
Which properties have an Institution?
Which is the institutional code for “Pharmacological Producer Laboratory” SAF?
Which Institutions have Program Code = 16?

Table 3: Competency Questions

3.4 Specification: Ontology Granularity and Type

According to the level of conceptualization and granularity (Gómez-Pérez et al, 2004), the ontology proposed here is domain ontology. Domain ontology describes the vocabulary related to a specific domain. In this case study, the ontology describes the budgetary domain of Santa Fe Province. And, the ontology objective is to facilitate communication among the members of the central administration staff that must deal with the local budget, bringing adequate terminology to non-expert users.

The term ontology can be used to describe models with different degrees of structure. Particularly, the ontology defined in this paper is a formal structure expressed in artificial formally defined languages.

3.5 Conceptualization: Domain Conceptual Model

In this step, a list of the most important terms was elaborated according to the 101 METHOD guide. To this aim, the middle-out strategy (Uschold, 1996) was used. With this strategy, the core of basic terms is identified first and then they are specified and generalized if necessary. Then with these concepts as reference, the key term list was defined. The list shown in Table 4 does not include partial or total overlapping of concepts, synonyms, properties, relations and attributes.

To properly understand the conceptual aspects in the context, a Unified Modelling Language (UML) diagram (UML, 2006) was elaborated with the main relations among defined concepts.

Activity	Financing Source
Budget	Geographic Location
Budget Analytic	Institutional
Budget Approved	Institution
Budget Project Draft	Jurisdiction
Budget Synthetic / Synthesis?	Jurisdiction Government
Budget States	Program
Budgetary Classifier	Subpartial Item
Budgetary Fiscal Year	Subprogram
Budgetary Policy	Program Executer Unit (UEP)
Budgetary Top	Programmatic Category Project
Executor Organism	Project
Expense	Public Funds Administrative Service (SAFOP)
Expense Classifier	Rector Organism
Expense Object	Resource
Finality Function	Resource Estimation
Financial Administration	Financial Administrative Service (SAF)

Table 4: Key Terms

The UML class diagram can be used to express concepts in terms of classes and relationships among them (Cranefield & Purvis, 1999). In addition, if an ontology-based application is being constructed using object-oriented technology, it may be advantageous to use the same paradigm for modelling ontologies and knowledge (Cranefield, 2001). In the last years, some MOF-based ontology modelling languages were defined (Caliusco, 2005). However, there are not appropriate tools to use them.

Although UML in its standard form is not suitable for semantic representation, the information modelled in the UML class diagram was the base for building the ontology term glossary, trying to include other concepts by means of generalization and specialization techniques. The conflictive assertions over the same entity may be discovered if the concepts are described as completely as possible (Jones et al, 1998). To this aim, definitions were made as complete as possible to contribute to define rules and axioms.

This UML model was useful to verify the ontology scope and to discover two granularity levels for budgetary domain concepts. Then it was necessary to make an important design decision: working with two ontologies. One of them is the Domain Ontology that contains the general concepts for the budget life cycle and a coarse granularity is adequate. The conceptual model of the Domain Ontology is shown in Fig.1. The other, Formulation Ontology, contains the semantic specific for formulating a budget. This is a task ontology (Gómez-Pérez et al, 2004) since it defines concepts related to a specific task and a fine granularity is necessary. So, we have to modify the list of key terms, hierarchical relations, and to group competency questions depending on the ontology concepts they were related with.

Working with different ontologies allows the term reusability and usability. These concepts are important goals in ontologies construction (Jarrar, 2005) and differ finely. While reusability implies to maximize the ontology use among different task types, usability maximizes the number of different applications using the

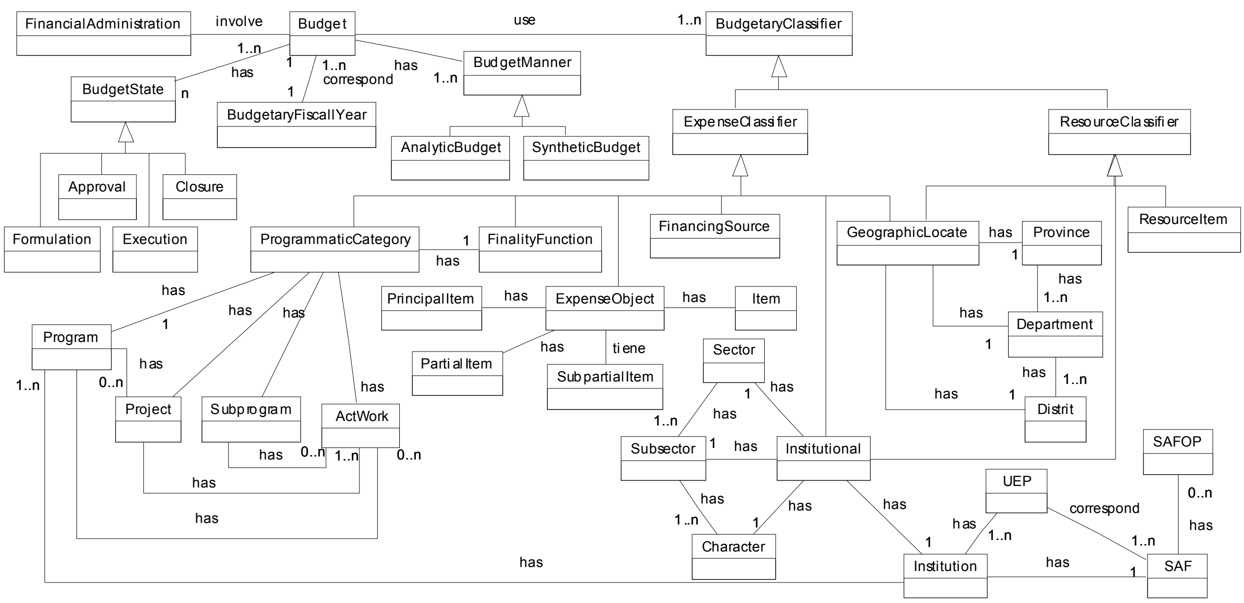

Fig. 1. Domain Model in UML.

same ontology. Therefore, the work is concentrated on Domain Ontology development. This Ontology of general concepts will be able to be used in all budget states facilitating term reusability. Then, we can see that it ontology will be able usability too.

3.6 Conceptualization: Identification of Classes, Relations and Attributes

At this step, we considered 101 METHOD guide and recommendations. Besides, we used representations proposed by METHONTOLOGY to knowledge organization as concepts classifier trees (Fig. 2) to analyze hierarchies and attributes, binary relations, axioms and instances tables. For determining classes, we identified those terms of independent existence from the key terms list and the glossary.

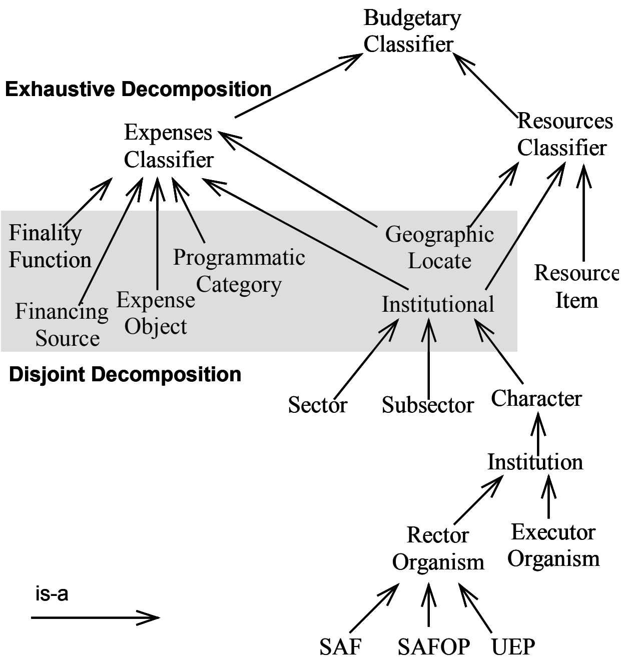

Fig. 2. Concepts Classifier Tree

Disjoint classes, exhaustive decompositions and partitions (Horridge et al, 2001) may be identified in these graphic representations:

- A Disjoint-Decomposition of a concept C is a set of subclasses of C that do not have common instances and do not cover C, that is, there can be instances of the concept C that are not instances of any of the concepts in the decomposition. As an example (see Fig. 2), Finality Function, Financing Source, Expense Object, Programmatic Category, Geographic Locate and Institutional can be mentioned as disjoints.
- An Exhaustive-Decomposition of a concept C is a set of subclasses of C that cover C and may have common instances and subclasses, that is, there cannot be instances of the concept C that are not instances of at least one of the concepts in the decomposition. For example (see Fig. 2), the concepts Expenses Classifier and Resource Classifier make up an exhaustive decomposition of the concept Budgetary Classifier because there are no classifiers that are not instances of at least one of those concepts, and those concepts can have common instances.
- A Partition of a concept C is a set of subclasses of C that do not share common instances and that cover C, that is, there are not instances of C that are not instances of one of the concepts in the partition. In this scenario there are no partitions.

It is always convenient to begin with primitive classes, analyzing which of them are disjoint and verifying if that condition does not produce instances absents.

Once the hierarchies and their features have been identified a table to reflect bidirectional relations may be elaborated by means of assigning names using uniform criteria (or a uniform criterion), identifying domain and range, cardinality and inverse relations. An example is shown in Table 5. Shaded rows are bidirectional relations between concepts shown in the Concepts Classifier Tree. The relation direction depends on competence questions to be solved and the possible conflicts with other defined classes restrictions. A restriction list identifies those necessary and sufficient conditions and those only necessary to work later on their formalization. We analyzed the axioms both individually and in a group of classes to verify if closure restrictions are required.

Concept	Relation	Cardinality	Concept	Inverse Relation
Institutional	inst-include-sec	1	Sector	sec-isPartOf-Inst
Institutional	inst-include-sbsec	1	Subsector	sbsec-isPartOf-Inst
Institutional	inst-include-char	1	Character	char-isPartOf-Inst
Sector	sec-isPartOf-Inst	1,n	Institutional	inst-include-sec
Subsector	sbsec-isPartOf-Inst	1,n	Institutional	inst-include-sbsec
Character	char-isPartOf-Inst	1,n	Institutional	inst-include-char
Character	char-has-Inst	1,n	Institution	inst-correspond-char
Institution	ins-has-SAF	1	SAF	SAF-correspond-inst

Table 5. Bidirectional Relations

3.7 Conceptualization: Instance Definition

Once the conceptual model of the ontology has been created, the next step is to define relevant instances inside an instance table.

According to METHONTOLOGY, each instance should be provided a definition of: its name, the name of the concept it belongs to, and its attribute values if known.

An excerpt of the Instance Table of the Budgetary Ontology is shown in Table 6.

Concept Name	Instance Name	Property	Value
Institutional	Institutional_111	cod-institutional	1.1.1
		has-fiscal-year	2004
		inst-include-sec	1-No Financial Local Public Sector
		inst-include-sbsec	1- Local Administration
		inst-include-char	1- Main Administration
	Institutional_212	cod-institutional	2.1.2
		has-fiscal-year	2004
		inst-include-sec	2-Financial Local Public Sector
		inst-include-sbsec	1-Offcial Banking System
		inst-include-char	2- Official Banks

Table 6. An excerpt of the Instance Table of the Budgetary Ontology.

4 Implementing the Budget Ontology with PROTÉGÉ 3.1

In order to implement the ontology, we chose Protégé 3.1 because of the fact that it is extensible and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development (Knublauch et al, 2005). Protégé ontologies can be exported into different formats including RDF Schema (RDFS) (Brickley & Guha, 2004), and Web Ontology Language (OWL) (Smith et al, 2004). Particularly, we have implemented the Budgetary Ontology in OWL and verified its consistency by using Racer (Haarslev & Möller, 2001). It was very useful for determining unsatisfiability problems and their propagation causes. An OWL class is deemed to be unsatisfiable (inconsistent) if, because of its description, it cannot possibly have any instances (Wang, 2005).

During the verification process, we have taken into account experience of CO-ODE Project (Knublauch et al, 2005), and practical experience of teaching OWL-DL reported by (Rector et al, 2004).

To compare the ontology implementation with its conceptualization, graphics using the OWLViz and Ontoviz plug-ins were generated and compared with UML diagrams. On the one hand, OWLViz enables the class hierarchies in OWL Ontology to be viewed, allowing comparison of the asserted class hierarchy and the inferred class hierarchy. OWLViz integrates with the Protege-OWL plugin, using the same color scheme so that primitive and defined classes can be distinguished, computed changes to the class hierarchy may be clearly seen, and inconsistent concepts are highlighted in red. Fig. 3 shows the Domain Ontology taxonomy.

Fig. 3. Domain Ontology Taxonomy.

On the other hand, OntoViz generates diverse combinations of graphics with all relations defined in the ontology, instances and attributes. OntoViz allows visualizing several disconnected graphs at once. These graphs are suitable for presentation purposes, as they tend to be of good clarity with no overlapping nodes. An example of them in Fig. 4 shows The main relations of the concept Institutional with other concepts, and an instance of this concept, Local Administration.

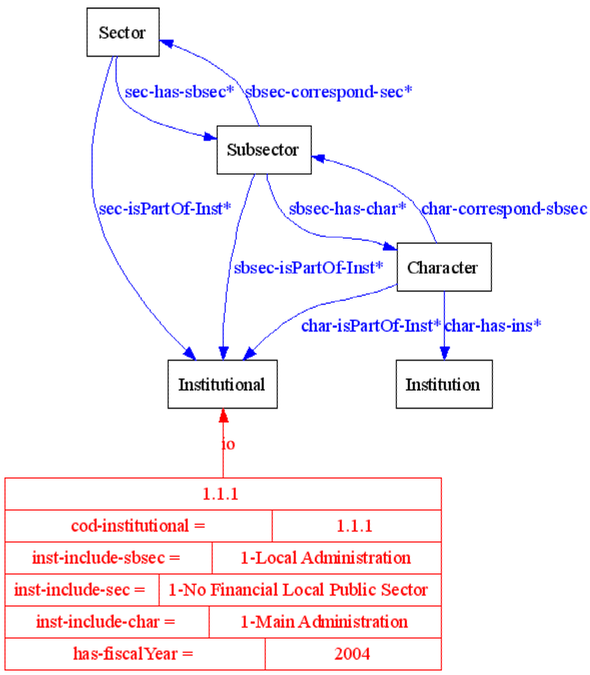

Fig. 4. Main Relations Between Concepts of Institutional Classifier

4.1 Ontology Querying

In order to verify and validate the ontology as regards competency questions, we used the RDF Data Query Language (RDQL) (Seaborne, 2004). RDQL is an implementation of an SQL-like query language for RDF. It treats RDF as data and provides queries with triple patterns and constraints over a single RDF model. Another query language is OWL-QL (Fikes et al, 2003), which was designed for query-answering dialogues among agents using knowledge in OWL. Then, OWL-QL is suitable when it is necessary to carry out an inference in the query. This is not the case of the major competency questions; then, RDQL is enough. Hence, RDF ontology was created from Protégé Project. Following the RDQL query that models the competency question "Which are the sector and subsector for Main Administration?" is shown.

```
SELECT ?x ?y ?z ?nsec ?nsbsec

WHERE (x,<adm:rdfsec-hassbsec>,?y)
 (?y,<adm:rdfsbsec-has-char>,?z)
 (?z,<rdfn:label>,'1-Main Administration')
 (?x, <rdfn:label>, ?nsec),
 (?y, <rdfn:label>, ?nsbsec)

USING rdfn FOR
      http://www.w3.org/2000/01/rdf-schema#
       adm FOR http://protege.stanford.edu/
```

To implement the queries, Jena framework has been used. Jena is a Java toolkit which provides an API for creating and manipulating RDF models. Jena sources can be retrieved at http://jena.sourceforge.net/.

5 Discussion

In order to develop the ontology presented in this paper, the methodology outlined in Fig. 8 has been followed. This methodology was divided into three phases: Specification, Conceptualization and Implementation according to the METHONTOLOGY Framework. These phases constitute an iterative process. This framework provided the idea of support activities: Knowledge Adquisition and Validation/Verification.

The innovation of the methodology presented in this paper consists of the tasks that compose each phase. These tasks were imported from 101 Method and Grüniger & Fox Methodology. In addition, they were enriched with some Software Engineering techniques.

The most important task in the methodology is the definition of a Domain Conceptual Model. Then, it is important to assign all the necessary time to carry out a good conceptual analysis. The conceptual model resumes the knowledge adquired during the specification phase and it is the basis of conceptualization. This conceptualization has to be agreed on by domain experts. Then, the use of a graphical representation is essential in order to facilitate communication between ontology engineers and experts. So, software engineering techniques that could be familiar for the domain experts, such as UML, can be useful. Although UML in its standard form is not suitable for ontology representation, we cannot ignore that UML is a standard and its use is widely spread among different communities.

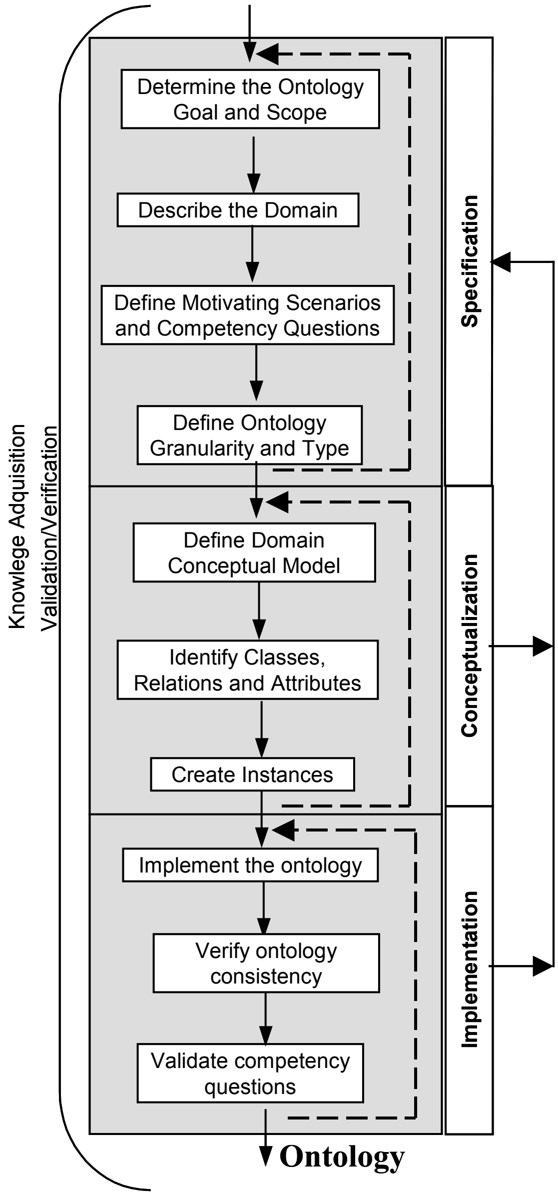

Fig. 8 . A Domain Ontology Development Process.

Another important aspect to consider in developing a good ontology is to carry out a permanent and iterative validation process, taking into account that partial verifications allow identifying errors propagation between sets of classes.

Furthermore, for the purpose of making the ontology more flexible and allowing extensibility and reuse, it is important to modularize the ontology if possible. This modularization can be made through relations and attributes observation of conceptual aspects involved.

6 Conclusions

Building domain ontologies is not a simple task when domain experts have no background knowledge on engineering techniques and/or they have not much time to invest in domain conceptualization.

In this paper, we have shown how ontologists could develop domain ontologies merging different methodologies and software engineering techniques, taking advantages of them. Particularly, this approach has been used to define a Domain Ontology for a Budgetary and Financial System, which could be extended by Task Ontologies and used by different government applications.

Sharing the best practice on ontology building can be useful for the whole community. Then, the contribution of this paper is the implementation and improvement of a systematic process for the development of domain ontologies.

7 References

Brickley, D., Guha, R.V. (2004) RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation. http://www.w3.org/TR/rdf-schema/

Caliusco M. L. (2005) A Semantic Definition Support of Electronic Business Documents in e–Colaboration. PhD thesis. UTN - F.R.S.F. Santa Fe, Argentina.

Corcho O, Fernández-López M, Gómez-Pérez A, López-Cima A. (2005) Building legal ontologies with METHONTOLOGY and WebODE. Law and the Semantic Web. Legal Ontologies, Methodologies, Legal Information Retrieval, and Applications.

Cranefield, S. (2001) UML and the Semantic Web. Proceedings of SWWS'01, The first Semantic Web Working Symposium, Stanford University, California, USA:113-130.

Cranefield, S., Purvis, M. (1999) UML as an Ontology Modelling Language. Proceedings of the IJCAI-99 Workshop on Intelligent Information Integration, Held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence City Conference Center, Stockholm, Sweden.

Falbo, R.A. (2004). Experiences in Using a Method for Building Domain Ontologies. Proceedings of the Sixteenth International Conference on Software Engineering and Knowledge Engineering, SEKE'2004, pp. 474-477, International Workshop on Ontology In Action, OIA'2004. Banff, Alberta, Canada.

Fikes, R., Hayes, P., Horrocks, I. (2003) OWL-QL - A Language for Deductive Query Answering on the Semantic Web. KL Laboratory, Stanford University, Stanford, CA.

Gómez-Pérez A., Fernández López M. and Corcho O. (2004) Ontological Engineering with examples from the areas of knowledge management, e-commerce and the semantic web. London: Springer.

Gruninger M. and Fox M. S. (1995) Methodology for the Design and Evaluation of Ontologies, IJCAI Workshop on Basic Ontological in Knowledge Sharing, Montreal, Canada.

Haarslev V. and Möller R. (2001). RACER System Description. In Proceedings of the First international Joint Conference on Automated Reasoning. IJCAR.

Horridge M., Knublauch H., Rector A., Stevens R., Wroe C. (2004) A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0, The University Of Manchester Stanford University.

Jarrar M. (2005) Towards Methodological Principles for Ontology Engineering. PhD Thesis, Vrije Universiteit Brusell.

Jones D., Bench-Capon T. y Visser P. (1998) Methodologies for Ontology Development, en Proc. IT&KNOWS Conference, XV IFIP World Computer Congress, Budapest.

Klischewski R., Lenk K., (2002), Understandind and Modelling Flexibility in Administrative Processes. In: Proceedings of EGOV 2002, Ed. Traunmuller R., and Lenk K., pp.129-136.

Knublauch H., Horridge M., Musen M., Rector A., Stevens R., Drummond N., Lord P., Noy N., Seidenberg J., Wang H. (2005) The Protégé OWL Experience, Workshop on OWL: Experiences and Directions, Fourth International Semantic Web Conference (ISWC2005), Galway, Ireland.

Noy N., McGuinness D., (2001) Ontology Development 101: A Guide to Creating Your First Ontology.

Rector, A., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., Wroe, C (2004) OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns (2004) in E Motta and N Shadbolt, et al. (eds) Proceedings of the European Conference on Knowledge Acquistion, Northampton, England, LNAI3257, pp 63-81

Seaborne A., (2004) RDQL - A Query Language for RDF, W3C Member Submission http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/

Smith, B. “Ontology”, (2003), in L. Floridi (ed.), *Blackwell Guide to the Philosophy of Computing and Information*, Oxford: Blackwell, 155–166.

Smith, M., Welty C., McGuinness D. (2004) OWL Web Ontology Language Guide, W3C Recommendation 10 http://www.w3.org/TR/owl-guide/

UML (2006) Unified Modeling Language. http://www.uml.org/

Uschold, M., Building Ontologies: Towards a Unified Methodology, (1996) 16th Annual Conference of the British Computer Society Specialists Group on Expert Systems, Cambrige, UK.

Uschold, M., Gruninger M. (1996) Ontologies: Principles, Methods and Applications. Knowledge Engineering Review.

Wache H., Vögele T., Visser U., Stuckenschmidt H., Schuster G., Neumann H., Hübner S. (2001) Ontology-Based Integration of Information –A Survey of Existing Approaches. Proc. IJCAI-01 Workshop: Ontologies and Information Sharing, Seattle, WA, 108-117.

Wang H., Horridge M., Rector A., Drummond N. and Seidenberg J., (2005), Debugging OWL Ontologies: A Heuristic Approach. 4th International Semantic Web Conference (ISWC'05), Galway, Ireland.