Building Ontology in Public Administration: A Case Study

Graciela Brusa[1], Ma. Laura Caliusco[2], Omar Chiotti[3]

[1] Dirección Provincial de Informática, San Martín 2466
Santa Fe, Santa Fe, Argentina, gracielabrusa@santafe.gov.ar
[2]CIDISI, CONICET-UTN-FRSF, Lavaise 610,
Santa Fe, Santa Fe, Argentina, mcaliusc@frsf.utn.edu.ar
[3]INGAR, CONICET-UTN-FRSF, Avellaneda 3657
Santa Fe, Santa Fe, Argentina, chiotti@ceride.gov.ar

Abstract. The inclusion of Semantic Web technologies into some areas, particularly in the public sector, has not been as expected. That is, among others, because government processes require a large amount of information and its semantic is impossible to carry across organizations. Hence, public servers depend on technical and specifics areas to incorporate knowledge about information that crosses the organization structure government. It succeeds too when government administrations aim at web services and people needs access to semantic of services. In this public services transformation, it is necessary incorporate new tools to be used by community whom this services are addressed. Ontologies are important to share information in internal activities of government administration and to facilitate information access in e-government services. This work presents the experiences during the ontology building in a local public sector: the budgetary and financial system of Santa Fe Province (Argentine). Software engineering techniques were use in manner of minimize the impact of technical knowledge required. At last, architecture is proposed in order to show ontologies applications in government areas and their advantages.

Keywords: ontology, public sector, budgetary and financial system.

1 Introduction

During the last years, an important progress on achieving information interoperability between heterogeneous applications in business sector has been made. Public administrations are facing the same problems than business organizations with respect to information integration. In the public sector, however, the direct replication of the experiences from business sector drives several problems [20], mainly since the complexity of the public sector.

The main difference between the business sector and the public sector is not only the complexity but also the bureaucracy and idiosyncrasy. To comprehend the public sector idiosyncrasy it can be adequate to consider the holistic reference model presented by [26], which, based on a socio-technique approach, makes a consideration of the public sector, showing different views, progress of public services and abstraction layers. From a technologic point of view, a main government challenge is to get a set of capabilities to facilitate the interoperability, needed for integration as well as the suitable interpretation of information to make decisions.

The interpretation of information without misunderstanding require to make its meaning explicit. To this aim, ontology can be used. Ontology provides a shared vocabulary for common understanding of a domain.

There are several works on how to develop ontologies methodologically. As example can be mentioned: Grüninger and Fox [9], METHONTOLOGY [8][22], and 101 Method [16], among others. These methodologies were successfully used to define ontologies in different domains [4]. Each of them presents different intermediate representations.

Concerning software platforms that aid in ontology development can be mentioned Protégé 3.1, and WebODE [1], among others.

In this paper we present how to develop a budgetary ontology following different development ontology methodologies and using Protégé 3.1. To this aim, the paper is organized as follow. Section 2 describes budgetary and financial system of the Santa Fe Province. Section 3 discusses the tasks carried out to build the budgetary ontology. Section 4 presents the ontology implementation using Protégé 3.1. Section 5 introduces architecture to support information integration using the implemented ontology. Finally, Section 6 presents our conclusions.

2 Budgetary and Financial System: Domain Description

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 governments. Particularly, in the Santa Fe Province (Argentine) the entities are actors participate:

- Executive Power: it carries out the Provincial Budget Draft. A Rector Organism that conducts all activities and all the Executors Organisms existing in government compounds it that formulates their own budgets.
- Legislative Power: it sanctions the annual budget law.

The interaction among these actors leads different budget states: In Formulation, in Parliamentary Proceeding and Approved. This iterative process is shown in Fig. 1.

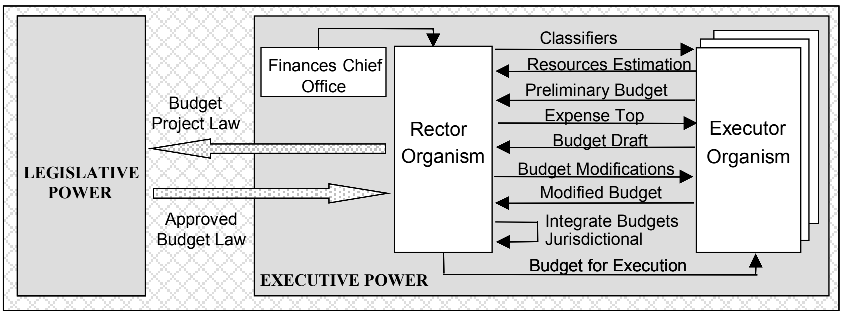

Fig. 1. Iterative process until budget is ready for execution.

In the Executive Power exists a Rector Organism that is responsible for all the budgetary formulation process. This Rector Organism sets the budgetary policies and drives the jurisdictional interactions to complete and integrate its own expenses and resources estimates through this formulation process. Each jurisdiction as Health or Production Ministries has Executor Organism, which are responsible to formulate and execute budget. Formulation process results in the Project of Budget Law issued to Legislative Power for approving.

Local budget life cycle (Fig. 2) is complex because involve a sequence of different instances with a lot of data to each other and a great and specific knowledge is required to operate with them.

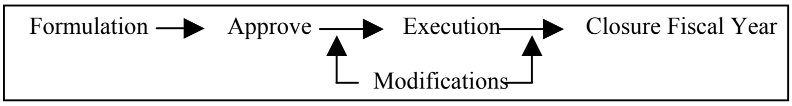

Fig. 2. Local Budget Life Cycle.

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

Table 1. Budget Life Cycle Steps

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

There is common information for all budget life cycle stages: Expenses and resources classifiers. They carry over all budgetary life cycle states bringing a thematic classification for its imports. Primary 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 first case, only government staff with specific knowledge can be involved in this task, concentrating a high responsibility in few persons with much difficult to knowledge transference. In second case, semantics information it is necessary to analyze budgetary data and then to sanction budget law. Here, it is more complex because all the legislators must vote and the majority has not the specific knowledge. For simplicity, the Formulation stage for expenses budget was considered to this study case.

3 Building the Budgetary Ontology

The objective of this section is to discuss the steps we have carried out in order to define an ontology that describes the semantics of the budgetary system domain.

3.1 A Methodology Selection

Before starting to define the ontology, different development methodologies were studied [5][14][24]. From this study, could be identified two main groups. On the one hand, there are experience-based methodologies, such as the methodology proposed for Grüninger y Fox [9], based in TOVE Project or the other exposed by Uschold y King [21] [24] from Enterprise Model. Both issue in 1995 and both belong to the enterprise modeler domain. On the other hand, there are methodologies that propose evolutive prototypes models, such us METHONTOLOGY [8] that proposes a set of activities to develop ontologies based on its life cycle and the prototype refinement; and 101 Method [16] that proposes an iterative approach to ontology development.

There is no one correct way or methodology for developing ontologies. Usually, the first ones are more appropriate when purposes and requirements of the ontology are clear, the second one is more useful when the environment is dynamic and difficult to understand exists [5]. Moreover, it is common to merge different methodologies since each of them provides design ideas that distinguish it from the rest. This merging depends on the ontology users and ontology goals.

At this work, both approaches were merged because in one hand, requirements core are clear but in the other, domain complexity drives to adopt an iterative approach to manage refinement and extensibility.

In general, the ontology development can be divided into two main phases: specification and conceptualization. The goal of the specification phase is to acquire 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. In order to define the ontology for the budget domain we have followed the 101 Method (OD101) guides for creating a first ontology [16] and used the analysis steps from METHONTOLOGY in the conceptualization process. Both consider an incremental construction that allows refining the original model in successive steps and they offer different representations for the conceptualization task.

3.2. Specification: Goal and Scope of the Ontology

The scope limits the ontology, specifying what must be included and what must not. In OD101, this task is proposed in a later step but we considered appropriate to include it at this point for minimizing the amount of data and concepts to be analyzed, especially for the extent and complexity of the budgetary semantic. In successive iterations for verification process, it will be adjusted if necessary.

This ontology only considers the needs to elaborate a project of budget law with concepts related to expenses. It is a first prototype and then, 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 specifics concepts for the formulation.

3.3. Specification: Domain Description

Taking into account that this work was made from scratch it was necessary to make a previous domain analysis. In this analysis, the application for 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 whole budget formulation process in Executive Power, expert professionals of Budget Committee in Legislative Power, public agents of administrative area taking charge of elaborate own budget and software engineers whom bring informatics supports for these tasks.

3.4. Specification: Motivating Scenarios and Competence Questions

We included this step taking into account the opinion of Gruninger y Fox [9] who considers that for modeling ontologies, it is necessary an informal logic knowledge model in addition to requirements resulting from different sceneries. The motivation scenerios show problems that arise when people needs information that the system does not provide. Besides, it contains a set of solutions to these problems in which the semantic aspects to resolve them are. In order to define motivation scenarios and communicate them to involved people, templates have been used. These templates were based on those proposed to specify case uses in object oriented methodology [23]. An example is shown in Table 2. In this template, the main semantic problems and a list of key terms were included.

Table 2. Scenario description.

SCENARIO N°	**1**	
NAME	**Local Budget Formulation**	
DESCRIPTION	**Necessary tasks to estimate expenses for next year, which will be integrate with the other government jurisdictions for elaborating Draft Local Budget. ……**	
SITE	**Executor Organism of a Jurisdiction**	
ACTORS	▪ **Public agents uncharged jurisdictional budget** ▪ **Rector Organism agents** ▪ **Public agents from areas of a jurisdiction**	
PRE-REQUIREMENTS	▪ **Budgetary Policy defined** ▪ **Expenses Classifiers received from Rector Organism** ▪ **Reference documentation**	
ASSOCIATES REQUIREMENTS	▪ **Prepared agents in Budget Formulation tasks.** ▪ **Advisory agents from Rector Organism**	
NORMAL SEQUENCE	**STEP**	**ACTION**
	1	**To receive expenses estimations from jurisdiction areas**
	2	**To bring support to this areas for elaborating own expenses programs.**
	3	**To integrate all expenses programs for jurisdiction.**
	4	**To create Programming Categories and send it to Rector Organism**
	5	**To create the Jurisdictional Budget Project**

	6	To load budget in informatics system and send it to Rector Organism
	7	To receive approved jurisdictional budget from Rector Organism
POST-CONDITION	▪ Jurisdictional Expenses Budget Project ▪ Jurisdictional Programmatic categories	
EXCEPTIONS	STEP	ACTION
	5	To consult the Rector Organism if it is not understands different aspects to formulate budget.
	7	To modify budget if it is not approved
PERMANENT TASKS	▪ To interact with Rector Organism to clarify the knowledge of conceptual domain aspects ▪ To bring support to different areas of jurisdiction	
MAIN PROBLEMS	▪ A lot of time loosed in clarify conceptual doubts ▪ Great problems when an agent must be replaced in key places of work. ▪ The whole process is highly dependent of few persons knowledge. ▪ ……….	
MAIN TERMS	Budgetary classifier, expense a classifier, Institutional, Programmatic Category, Geographic, Expenses Object, Financing Source and Finality Function Classifiers, among others, for working into the budget draft task.	

The competency questions proceed from motivation sceneries, allowing 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 must not be. It means that a concept must not be included if there are not competency questions to use its. This rule is also used to determine if an axiom in ontology must be included or not.

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

Table 3. Samples of Competency Questions

Simple Questions	Complex Questions
Which are budget states?	Which is the institutional code for Department of Labor?
Which are budgetary classifiers?	Which are sector and subsector for Central Administration?
Which are expenses classifiers?	What is the character code for “Decentralized Organism”?
Which are resources classifiers?	Which properties have an Institution?
Which are the executor organisms for Health Minister?	Which is the institutional code for “Pharmacological Producer Laboratory” SAF?

3.5. Specification: Ontology Granularity and Type

According to purpose and level of granularity [8], the ontology proposed here was defined as a domain ontology. Domain ontology describes the vocabulary related to a specific domain. In this case study the ontology describe the budgetary domain of the Santa Fe Province. And, the ontology objective is to facilitate the communication between 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.

3.6. Conceptualization: Conceptual Domain Model Determination

In this step, a list of more important terms was elaborated according to OD101. To this aim, the middle-out strategy [19] was used. With this strategy, the core of basic terms is identified first and then they are specified and generalized if necessary.

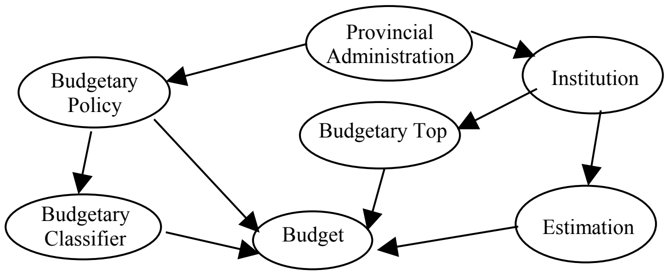

Fig. 3. Basic terms of the budget domain.

Then with these concepts as reference, the key term list was defined. List shown in Table 4 does not include partial or total overlapping of concepts, synonyms, properties, relations and attributes.

Table 4. Key Terms

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

To properly understand the conceptual aspects in the context, we elaborated a Unified Modeling Language (UML) diagram [23], (Fig. 4), with the main relations among defined concepts. UML is a useful tool for ontology modeling though it was not designed for this task [3].

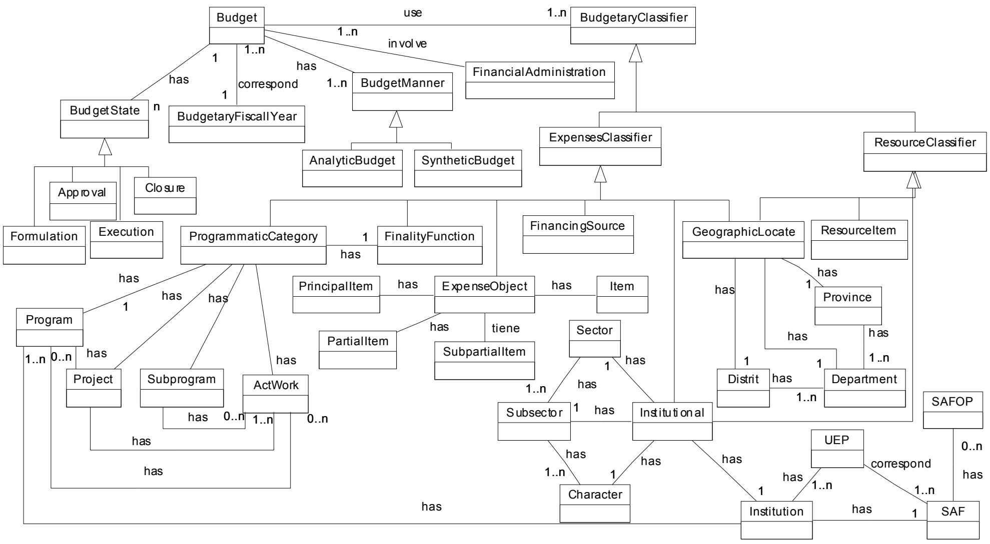

Fig. 4. Domain Model in UML.

This information 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 [12], 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 take 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 the other, Formulation Ontology, contains the semantic specific for formulating it. This is task ontology [10] since it defines concepts related to a specific task, the budget formulation. 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. As Guarino sets [10], it exists ontology types accord with dependence level of task or viewpoint. Hence, ontologies construction implies two different strategies [6]. In one hand, a domain ontology with an application-independent strategy because its general concepts must be available all time. In other hand, task ontology is application-semidependent because different use scenerios can be identified and its conceptualization is associated to real activities.

Working with different ontologies allows the term reusability and usability. These concepts are important goals in ontologies construction [13] and differ finely. While reusability implies to maximize the ontology use among different task types, usability maximizes the number of different applications using the same ontology. From now, the work is concentrated on Domain Ontology development. This Domain Ontology will be able for using in all budget states facilitating term reusability.

3.7. Conceptualization: Identification of Classes, Relations and Attributes

At this step, we considered OD101 recommendations. Besides, we used representations proposed by METHONTOLOGY to knowledge organization as concepts classifier trees (Fig. 5) 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.

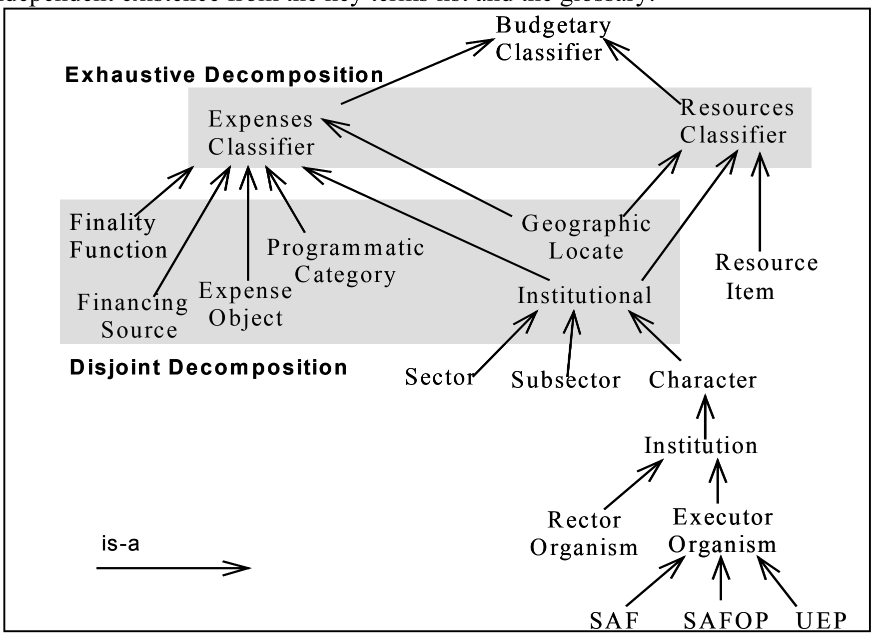

Fig. 5. Taxonomy of Budgetary Ontology Concepts.

Disjoint classes, exhaustive decompositions and partitions [12] 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 example (see Fig. 5), 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. 5), the concepts Expenses Classifier and Resource Classifier make up an exhaustive decomposition of the concept Budgetary Classifier because there are no classifier 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, examining 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 with an uniform criteria. An example is shown in Table 5. Shades rows are self-evident relations between concepts shown in the Concepts Classifier Tree (see Fig. 5) that it results bidirectional relation after analyzing them.

Table 5. Bidirectional Relations

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

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 in their formalization. We individually analyzed the axioms but also in a group of classes to verify if closure restrictions are required.

3.8 Conceptualization: Instances 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, for each instance should be defined: its name, the name of the concept it belongs to, and its attribute values, if known, as Table 6 shows.

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

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

4 Implementing the Budget Ontology with PROTÉGÉ 3.1

In order to implement the ontology, we chosen Protégé 3.1 due to it is extensible, and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development. Protégé ontologies can be exported into different formats including RDF Schema (RDFS) [2] and Web Ontology Language (OWL) [19].

Particularly, we have implemented the Budgetary Ontology in OWL. The first challenge during this task was how to transform the UML diagram from conceptualization phase into the OWL formalism. This task was hard and time consuming. Modeling in OWL implied to transform composition relations into bidirectional relations. In addition, some concepts modeled as classes in UML were properties in ontology. And not all relations in UML were modeled in ontology but only those relations that were necessary to answer competence questions. Moreover, the granularity of domain ontology is coarse and it was adequate select a flat structure for its.

Then, we verified the ontology by using Racer [11]. During the verification process, we have taken into account experiences of CO-ODE Project [15] and [17]. We verified consistency validation and classification. During process for charging classes and attributes, the verification was incremental and continuous to avoid future propagation errors. When a class is unsatisfiable, Racer shows it with a red bordered icon and there are different categories of causes [25] and can be exists propagated errors. At this point is very important how are classes defined (disjoint, isSubclassOf, Partial Class, Defined Class, etc.) and their restrictions (unionOf, allValuesFrom, etc.). Classification process can be invoked either for the whole ontology, or for selected subtrees only. When the test is over whole ontology, errors were searched beginning with minor level class in the hierarchy for minimizing propagation errors.

Fig. 6. An excerpt of Ontology Taxonomy.

To compare the ontology implementation with its conceptualization, graphics by using the OWLViz and Ontoviz plug-ins were generated and compared with UML diagrams. Fig. 6 shows an excerpt of the General Ontology taxonomy.

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. With OWLViz primitive and defined classes can be distinguished, computed changes to the class hierarchy may be clearly seen, and inconsistent concepts are highlighted in red. In the taxonomy shown here, can be seen how to represent a multiple inheritance with twice terms defined for Location Geographic. Another form is to use axioms and lets to reasoner generates inferred classes.

On the other hand, OntoViz generates a graphics with all relations defined in the ontology instances and attributes. It permits visualizing several disconnected graphs at once. These graphs are suitable for presentation purposes, as they tend to be of good clarity with none overlapping nodes. Besides, these graphics are very useful for analyzing when a concept must be modeled as a class and when must be modeled as an attribute. An example of them is shown in Fig. 7.

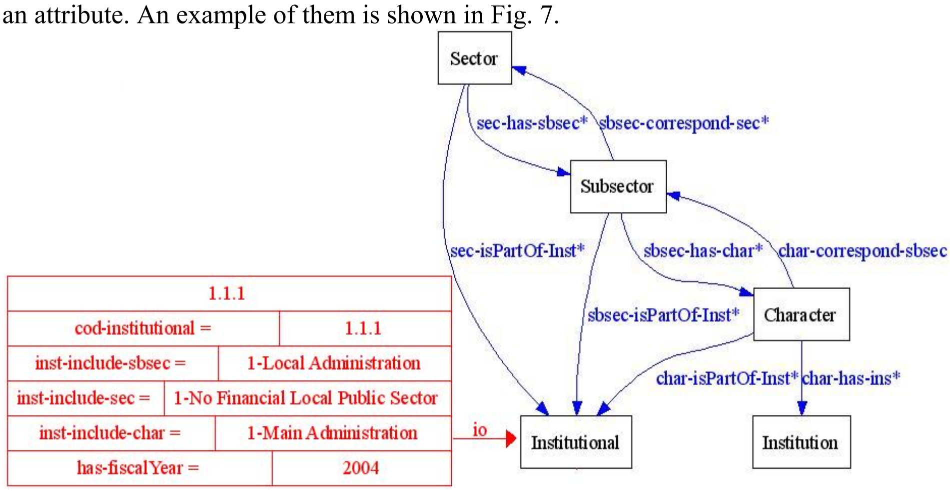

Fig. 7. Main Relations Between Concepts of Institutional Classifier.

4.1 Ontology Querying

In order to verify and validate the ontology regards to competency questions, we used the RDF Data Query Language (RDQL) [18]. RDQL is an implementation of an SQL-like query language for RDF. It treats RDF as data and provides query with triple patterns and constraints over a single RDF model. Another query language is OWL-QL [7], which was designed for query-answering dialogues among agents using knowledge in OWL. Then, OWL-QL is suitable when it is necessary to carried out an inference in the query. This is not the case of the major competency questions, then, RDQL is enough. To implement these queries we used the Jena framework, which provides an API for creating and manipulating RDF models.

Following the RDQL query that models the competency question “What are 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/
```

5 Using the Budget Ontology

The main ontology goal is to provide a mechanism for information sharing between people and applications without misunderstanding, independent of its implementation. Then, the final step to achieve the ontology goal is to design and implement architecture as one shown in Fig. 8. The architecture components are described next.

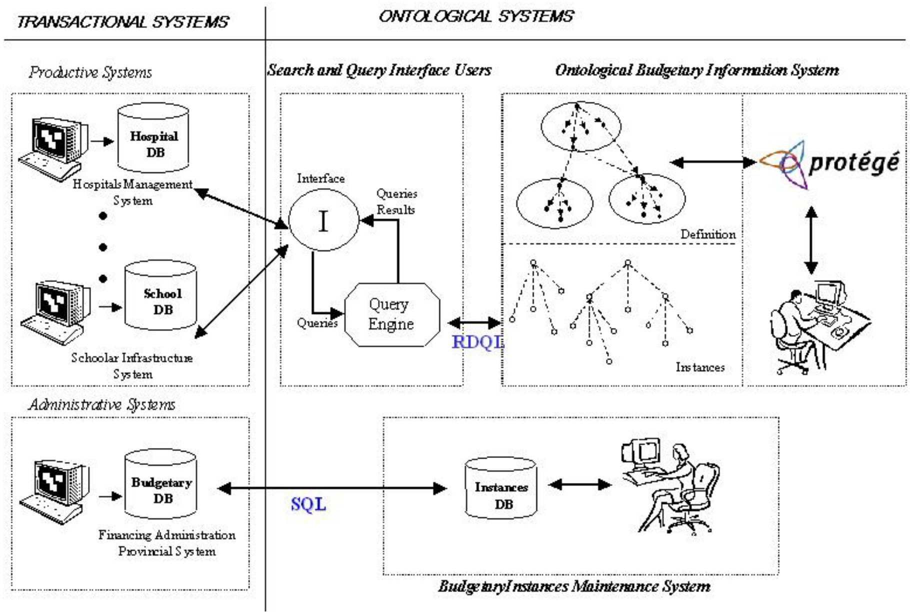

Fig. 8. Ontology-based Architecture for Budget Content Integration in Public Sector.

Ontological Budgetary Information System: Ontology designer team carry out the design, implementation and maintenance tasks using Protégé. This architecture proposes a general ontology for the common concepts for all the budgetary life cycle and specific ontologies for each stage of budget as formulation, approve, execution and closure.

Budgetary Instances Maintenance: expert persons realize the maintenance of instances for general and specific ontologies, requiring the necessary adjusts to ontological designer through the interaction with budgetary system and users.

Search and Query Interface Users: receive queries and return results of them through a friendly interface user. Applications or persons can issue queries through this interface that it uses RDQL as query language support.

Transactional Systems: both administrative and productive government systems. In this study case, a productive system as Hospitals or School Infrastructure Administrative System can access simply to budgetary information for own each interests through Ontological Systems.

6 Conclusions

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

The main conclusions that can be transmitted to the reader are:

- To assign all the necessary time to do a good conceptual analysis because it is considered the most important task during development ontology.
- To modularize the ontology while it is possible for giving it more flexibility and permitting extensibility and reuse. It can be made through relations and attributes observation of conceptual aspects involved.
- To take into account that there are steps that can be applied during the development of any ontology whereas there are steps that are domain-dependent.
- To realize a permanent and iterative validation process, taking into account that partial verifications allow to identify errors propagation between sets of classes.
- To define graphics to transmit the domain conceptualization to the domain experts. Some software engineering techniques that could be familiar for the domain experts, such as UML, can be useful.
- To consider for development who is maintenance responsible expert user and it anticipates a friendly interface user.

References

1. Arpírez JC, Corcho O, Fernández-López M, Gómez-Pérez A (2003) WebODE in a nutshell. AI Magazine, 24(3)-37-47.
2. Brickley, D., Guha, R.V. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-schema/
3. Caliusco M. L., A Semantic Definition Support of Electronic Business Documents in e–Colaboration. PhD Thesis. (Universidad Tecnológica Nacional, F.R.S.F., 2005.
4. Corcho O, Fernández-López M, Gómez-Pérez A, López-Cima A. Building legal ontologies with METHONTOLOGY and WebODE. Law and the Semantic Web. Legal Ontologies, Methodologies, Legal Information Retrieval, and Applications. March 2005.
5. Cristani M. and Cuel R. Methodologies for the Semantic Web: state-of-the-art of ontology methodology. SIGSEMIS Bulletin. SW Challenges for KM. July 2004 Vol. 1 Issue 2.
6. Fernandez Lopez M. (1999) Overview of methodologies for building ontologies. In: Benjamins VR, editor. IJCAI-99 Workshop on Ontologies and Problem-Solving Methods; Stockholm, Sweden: CEUR Publications: 1999.

7. Fikes, R., Hayes, P., Horrocks, I., OWL-QL - A Language for Deductive Query Answering on the Semantic Web. KL Laboratory, Stanford University, Stanford, CA, 2003.
8. Gómez-Pérez A., Fernández López M. and Corcho O. Ontological Engineering with examples from the areas of knowledge management, e-commerce and the semantic web. London: Springer, 2004.
9. Gruninger M. and Fox M. S., Methodology for the Design and Evaluation of Ontologies, IJCAI Workshop on Basic Ontological in Knowledge Sharing, Montreal, Canada, 1995.
10. Guarino N. (1998) Formal Ontology and Information Systems. In Proceedings of FOIS'98, Trento, Italy. Amsterdam, IOS Press.
11. Haarslev V. and Möller R. 2001. RACER System Description. In Proceedings of the First international Joint Conference on Automated Reasoning. IJCAR, June 2001.
12. Horridge M., Knublauch H., Rector A., Stevens R., Wroe C., 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, August 27, 2004.
13. Jarrar M., Towards Methodological Principles for Ontology Engineering. PhD Thesis, Vrije Universiteit Brusell, 2005.
14. Jones D., Bench-Capon T. y Visser P., Methodologies for Ontology Development, en Proc. IT&KNOWS Conference, XV IFIP World Computer Congress, Budapest, August 1998.
15. Knublauch H., Horridge M., Musen M., Rector A., Stevens R., Drummond N., Lord P., Noy N., Seidenberg J., Wang H., The Protégé OWL Experience, Workshop on OWL: Experiences and Directions, Fourth International Semantic Web Conference (ISWC2005), Galway, Ireland, 2005.
16. Noy, N., McGuinness D., Ontology Development 101: A Guide to Creating Your First Ontology, 2001.
17. Rector A., Drummond N., Horridge M., Rogers J., Knublauch H., Stevens R., Wang H., Wroe C., OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns.
18. Seaborne A., RDQL - A Query Language for RDF, W3C Member Submission 9 January 2004, http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/
19. Smith M., Welty C., McGuinness D., OWL Web Ontology Language Guide, W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-guide/
20. Traunmüller R., Wimmer M., Feature Requirements for KM in Public Administration, 2002, http://www.lri.jur.uva.nl/~winkels/eGov2002/Traunmuller.pdf
21. Uschold, M., Building Ontologies: Towards a Unified Methodology, 16th Annual Conference of the British Computer Society Specialists Group on Expert Systems, Cambrige, UK, 16-18 December 1996.
22. Uschold, M., Gruninger M., Ontologies: Principles, Methods and Applications, Knowledge Engineering Review, 1996.
23. Unified Modeling Language. http://www.uml.org/
24. Wache H., Vögele T., Visser U., Stuckenschmidt H., Schuster G., Neumann H., Hübner S., Ontology-Based Integration of Information –A Survey of Existing Approaches, in Proc. IJCAI-01 Workshop: Ontologies and Information Sharing, Seattle, WA, pp. 108-117, 2001.
25. Wang H., Horridge M., Rector A., Drumond N., Seidenberg J. (2005) Debugging OWL-DL Ontologies: A Heuristic Approach. 4th International Semantic Web Conference (ISWC'05), Galway, Ireland.