Centro Nacional de Investigación y Desarrollo Tecnológico

Departamento de Ciencias Computacionales

TESIS DE MAESTRÍA EN CIENCIAS

Enriquecimiento de Modelos Organizacionales a través de Anotación Semántica

presentada por

Blanca Hilda Vázquez Gómez

Lic. en Informática por el I. T. de Tuxtla Gutiérrez

como requisito para la obtención del grado de:

Maestría en Ciencias en Ciencias de la Computación

Director de tesis:

Dra. Alicia Martínez Rebollar

Co-Director de tesis:

Dra. Anna Perini

Jurado:

Dr. Hugo Estrada Esquivel - Presidente

Dr. Jorge Hermosillo Valadez - Secretario

Dra. Alicia Martínez Rebollar- Vocal

Dra. Blanca Alicia Vargas Govea – Vocal Suplente

Cuernavaca, Morelos, México. 14 de diciembre de 2012

The research reported in this thesis has been financially by National Council of Science and Technology “*CONACYT*” (Consejo Nacional de Ciencia y Tecnología)

Printed in Mexico

Cuernavaca, Morelos

SUBSECRETARÍA DE EDUCACIÓN SUPERIOR
DIRECCIÓN GENERAL DE EDUCACIÓN SUPERIOR TECNOLÓGICA
CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO

SECRETARÍA DE
EDUCACIÓN PÚBLICA

ANEXO No.11

M10

ACEPTACIÓN DEL DOCUMENTO DE TESIS

Cuernavaca, Mor., a 30 de noviembre de 2012

Dra. Alicia Martínez Rebollar
Jefa del departamento
de Ciencias Computacionales
Presente.

At´n: Dr. Raúl Pinto Elías
Presidente del Consejo del Posgrado
de Ciencias Computacionales

Nos es grato comunicarle, que conforme a los lineamientos para la obtención del grado de Maestro en Ciencias de este Centro, y después de haber sometido a revisión académica la tesis titulada **"Enriquecimiento de modelos organizacionales a través de anotación semántica (Enriching Organizational Models through Semantic Annotation)"** realizada por la alumna Blanca Hilda Vázquez Gómez y dirigida por el Dra. Alicia Martínez Rebollar y Codirigida por la Dra. Anna Perini, habiendo realizado las correcciones que le fueron indicadas, acordamos ACEPTAR el documento final de tesis, así mismo le solicitamos tenga a bien extender el correspondiente oficio de autorización de impresión.

Atentamente
La Comisión de Revisión de Tesis

Dr. Jorge Hermosillo Valadez	Dr. Hugo Estrada Esquivel	Dr. Javier Ortiz Hernández
Revisor	Revisor	Revisor

c.c.p. Dr. Gerardo Vicente Guerrero Ramírez.- Subdirección Académica
L.I. Guadalupe Garrido Rivera.- Jefe Departamento de Servicios Escolares
Dra. Alicia Martínez Rebollar, Dra. Anna Perini.- Directores de tesis
Interesado

SUBSECRETARÍA DE EDUCACIÓN SUPERIOR
DIRECCIÓN GENERAL DE EDUCACIÓN SUPERIOR TECNOLÓGICA
CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO

SECRETARÍA DE EDUCACIÓN PÚBLICA

ANEXO No. 12

M11

AUTORIZACIÓN DE IMPRESIÓN DE TESIS

Cuernavaca, Mor., a 03 de diciembre de 2012

C. Blanca Hilda Vázquez Gómez
Candidato al grado de Maestro en Ciencias
en Ciencias de la Computación
Presente.

Después de haber atendido las indicaciones sugeridas por la Comisión Revisora de la Academia de Ciencias de la Computación en relación a su trabajo de tesis cuyo título es **"Enriquecimiento de modelos organizacionales a través de anotación semántica"** me es grato comunicarle que conforme a los lineamientos establecidos para la obtención del grado de Maestro en Ciencias en este centro se le concede la autorización para que proceda con la impresión de su tesis.

Atentamente

Dra. Alicia Martínez Rebollar
Jefa del Departamento de Ciencias Computacionales

C.p. Dr. Gerardo Vicente Guerrero Ramírez.- Subdirección Académica
Dr. Raúl Pinto Elías.- Presidente del Consejo del Posgrado de Ciencias Computacionales
L.I. Guadalupe Garrido Rivera.- Jefe del Departamento de Servicios Escolares
Expediente

Acknowledgments

It is a pleasure to thank the many people who made this thesis possible.

I would like to express my deep gratitude to my advisor Alicia Martinez for her patient guidance, enthusiastic encouragement and useful critiques of this research work. With her enthusiasm and her inspiration today I can finish this dream.

I would like to offer my special thanks to my co-advisor Anna Perini for her valuable and constructive suggestions during the planning and development of this research work. Her willingness to give her time so generously has been very much appreciated. Thank you for give me the opportunity of working with you in the research group of the Foundation Bruno Kessler during a six month period. Working with you and your group was the most beautiful experience in all my life: "*Grazie Mille*"

Special thanks should be given to Mirko Morandini for his professional guidance and valuable support. Thank you very much for your time, patience and for the long hours of work: "*Dankesehr*". I wish to acknowledge the help provided by my adviser Hugo Estrada for his useful and constructive recommendations on this project. Thanks for trusting me. Thanks also to Angelo Susi for his invaluable suggestions and contributions to my master´s thesis.

I wish to thank my advisers Javier Ortiz and Jorge Hermosillo for their advice and assistance during the development of this research work. I would also like to extend my thanks to Monica Pichardo, Manuela Bacca and Patricia Armas for assisting me in many different ways.

I am particularly grateful with the Dr. Iván García and Itzel Morales. Thank you very much to teach me to see light where I only saw darkness. Thank you for accompany me in all my adventures. Thank you for your advices, encouragement and for not losing faith in me. I want to say thank you so much to Ilse Landa for her invaluable friendship, for supporting me, for all the adventures and the long talks and although I was far, you were always there for me: "*Gracias Amiga*".

I want to extend my acknowledgement to CONACYT for the support during my master, and to all my teachers at CENIDET for sharing their knowledge and especially to Dr. David Pinto for his advices and his trust in me.

I am in debted to my many colleagues for providing a stimulating and fun environment in which to learn and grow. In alphabetical order by name, I am especially grateful to: Adrián Cruz, Alejandra Molina, César Pérez, Chiara Di FrancescoMarino, Gabriela De León, Ilse Grau, Irving Linarte, Karen Nájera, Luis Santillán, Rafael Alcántara, Said Martagón, Samuel Vieyra, Simone Sain, Surafel Lemma, the MGN, Vinicio del Toro and Wilfrido Campos.

Muchas gracias a la señora Georgina Castro y al señor Alfredo, gracias por aceptarme como un miembro más de su familia. Que Dios los bendiga siempre.

Dedico especialmente este trabajo de tesis a mi hermosa familia. Son ustedes quienes han sentido mi ausencia durante estos casi dos años fuera de casa, la larga ausencia y la larga espera; sin embargo ustedes fueron mi motor y mi impulso para continuar día a día. Este esfuerzo te lo dedico con todo el corazón a ti mamita por enseñarme a luchar por mis sueños y a nunca bajar la guardia. Gracias por apoyarme siempre y por hacer de mí la mujer que hoy soy. Con todo mi cariño y respeto te dedico este logro que también es tuyo mamá.

Dedico también este esfuerzo a un ser especial que Dios me puso en camino, gracias por tu paciencia, tu confianza, por tu aliento y porque nunca me cortarme las alas, para ti mi Amor con todo mi cariño.

Agradezco también tu apoyo Bernardo y a ti Rosario por darme ánimos día a día y a impulsarme a seguir siempre adelante a pesar de las dificultades. También este esfuerzo es para ustedes y gracias por darme un angelito que dulcemente me llama "*Tía*" para ti Adriancito con todo mi amor también dedico este trabajo de tesis.

De todo corazón muchas gracias a todos por su apoyo y por confiar en mí.

Blanca Hilda Vázquez Gómez
14 diciembre 2012.

Abstract

Nowadays the relevance of the early requirements engineering techniques are widely recognized during the development of information systems. These modeling techniques allow a better understanding of the "*as-is*" and "*to-be*" of an organization with its social context, structure, processes and resources. The information obtained during the requirements elicitation phase is represented in diagrammatic models. Visual modeling has been recognized to be relevant in software development, supporting the communication among stakeholders involved in the development process, and become part of the process documentation. Visual modeling helps in the identification, analysis and description of essential concepts and constraint of a specific domain. Our research work is concerned with conceptual modeling for organizations, a relevant aspect in business reengineering, and in requirements analysis for organizational information systems. In this context, the *i** framework is one the of most relevant organizational modeling techniques today. In i* it is possible to express explicitly social relations and dependencies among stakeholders, and the representation of their internal goals and behaviors, required to satisfy actor dependencies. Throughout the years, different research groups have formulated variations to this framework to adapt it to the specific needs of their users. One of the current issues in business modeling today is the explicit representation of the meaning of the model components, this process, that it is called labeling, is one of the current topics of interest of several research groups. Moreover, the process of labeling the elements of the visual modeling is usually an activity, which is not rigorous and well documented for designers. It is often performed with freedom and subjectivity, resulting in unclear labels that describe the modeled entities, with mismatching and overlapping. This makes it hard to understand them outside the organizational context. Furthermore the amount of information that can be encoded in a human readable label is necessarily limited. In this way, the organizational models become complex due the fact that acquisition of knowledge which is not an easy task, and consequently model analysis and maintenance become inefficient or difficult for people who didn´t take part in building the original model.

We propose an approach to enrich an organizational model, described in the variants i*, Tropos and Service-oriented i*, with semantic annotation taken from domain ontologies. This allows the standardization of concepts, clarifying the label of the elements and achieving a common understanding in the community. A set of semantic annotation suggestions applied to a domain ontology and to a generic ontology has been described to guide the annotation process. The extensions of iStarML interchange format and an existing plug-in to export the model into an interchange format adding domain concepts for each element of the model and the presentation of TAGOOn+ (Tool for the Automatic Generation of Organization Model Ontology and Integration) to integrate the annotated model into a domain ontology have been developed for the accomplishment of this research work.

Keywords: Visual modeling, semantic annotation, ontologies, i* framework.

Resumen

Hoy en día la relevancia de las técnicas de ingeniería de requerimientos tempranos, son ampliamente reconocidas durante el desarrollo de los sistemas de información. Estás técnicas de modelado permiten un mejor entendimiento del "*como-es*" y del "*ser*" de un organización en el contexto social, estructural, procesos y recursos. La información obtenida durante la elicitación de requerimientos se representa en modelos esquemáticos. Los modelos visuales han sido reconocidos a diferentes niveles de relevancia en el desarrollo de software, soportando la comunicación entre actores, involucrando el desarrollo de procesos y llegan a ser parte de proceso de documentación. Un modelo visual ayuda en la identificación, análisis y la descripción de los conceptos esenciales y restricciones de un dominio específico. Nuestro trabajo de investigación se interesa en el modelado conceptual para sistemas de información organizacionales. En este contexto, en la actualidad el marco de trabajo i* es una de las más relevantes técnicas de modelado organizacional. En i* es posible expresar relaciones sociales explícitas y dependencias entre actores, y la representación de sus metas y comportamientos internos requeridos para satisfacer las dependencias de actores. Sin embargo, a través de los años, diferentes grupos de investigación han formulado variaciones a este marco de trabajo principalmente con el objetivo de adaptar i* a la necesidades específicas de los usuarios. Uno de los problemas comunes en modelado de negocios hoy en día es la representación explícita del significado de los componentes del modelo, este proceso es llamado etiquetado, es uno de los tópicos comunes de interés en diferentes grupos de investigación. Más aún, el proceso de etiquetado de los elementos de los modelos visuales es usualmente una actividad no rigurosa y bien documentada para diseñadores. Frecuentemente se lleva a cabo con cierta libertad y subjetividad, resultando etiquetas no claras que describen las entidades del modelo, discordancia y sobrelapamiento dificultando el entendimiento fuera del contexto organizacional. Además la cantidad de información que puede ser codificada en una etiqueta legible por un humano es necesariamente limitada. De esta manera, los modelos llegan a ser complejos debido al hecho que la adquisición de conocimientos no es una tarea fácil, y consecuentemente el análisis y mantenimiento de los modelos pueden llegar a ser ineficiente o difícil para personas quienes no han tomado parte de la construcción del modelo original. Nosotros proponemos un enfoque para enriquecer los modelos organizacionales descritos en las variaciones de i*, Tropos y orientada a servicios i* con anotaciones semánticas tomadas de ontologías de dominio permitiendo la estandarización de conceptos, clarificando las etiquetas de los elementos y logrando un común entendimiento en la comunidad. Un conjunto de sugerencias de anotación semántica aplicadas a cualquier ontología de dominio y hacia una ontología genérica han sido descritas para guiar el proceso de anotación. Las extensiones del formato de Intercambio iStarML y de un plug-in existente para exportar el modelo hacia un formato de intercambio añadiendo conceptos de dominio para cada elemento del modelo y la presentación de TAGOOn+ (Herramienta para la Generación Automática de Ontologías Organizacionales e Integración) para integrar el modelo anotado hacia una ontología de dominio han sido desarrollados para el cumplimiento de este trabajo de investigación.

Palabras claves:

Modelo visual, anotación semántica, ontologías, marco de referencia i*

Enriching Organizational Model through Semantic Annotation

Content

List of figures

List of tables

Acronyms

Abox: The Abox contains the assertions about the individuals in an ontology.

GSAS: Acronym of "*General Semantic Annotation Suggestions*" these set of suggestions is applied into any domain ontology.

OntoSem: OntoSem is the implementation of the theory of Ontological Semantics proposed by Nirenburg and Raskin (2004). The Ontological semantics is a theory of meaning in natural language and an approach to natural language processing which uses a constructed world model, or ontology, as the central resource for extracting and representing meaning of natural language texts, reasoning about knowledge derived from texts as well as generating natural language texts based on representations of their meaning.

OWL: OWL is part of the growing stack of W3C recommendations related to the Semantic Web. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full. It adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "*exactly one*"), equality, richer typing of properties and characteristics of properties (e.g. symmetry), and enumerated classes.

Plug-in: It is a set of software components that adds specific abilities to a larger software application. If supported, plug-ins enables customizing the functionality of an application. For example, plug-in is commonly used in web browsers to play video, scan for viruses, and display new file types.

Protégé: It is a free, open source ontology editor and knowledge-based framework. Protégé is based on Java, is extensible, and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and application development.

SSAS: Acronym of "*Specific Semantic Annotation Suggestions*" these set of suggestions is applied into the generic ontology called "*OntoSem*" and with any ontology that extended to OntoSem.

Tbox: The Tbox contains the axioms defining the classes and relations in an ontology.

TAGOOn+: Acronyms of the tool "*Tool for Automatic Generation of Organizational Ontologies and Integration*". This tool allows the integration of an organization model represented as ontology into a domain ontology. TAGOOn+ is a contribution of this research work.

Chapter 1 Introduction

1.1 Context and motivation

Nowadays, the complexity of information systems has forced software engineers to get deep understanding the organization before starting the construction of a software system to automate business processes. Therefore, it is now widely recognized the importance of the early requirements engineering techniques during the development of information systems[1]. In this sense, software engineers recognize the importance of organizational modeling techniques to determine precisely the requirements of software systems.

In this context, the *i** framework [2], is one the of most important approaches in organizational modeling for its ability to express intentional and explicit social relations among stakeholders [3] and describe internal behaviors to satisfy actor dependencies [4]. For this reasons, this framework is a relevant paradigm in Requirements engineering and agent orientation in Software Engineering [5].

The i* framework provides the need infrastructure to model concepts such as actors, roles and agents, and to reason about them [6]. i* has been applied for modeling organizations, business processes and system requirements, among others [7].

At present, research groups have formulated variations to this framework [7][8]. There are basically two reasons behind this fact: i) The definition of the i* language is loose in some parts, and some groups have adopted different solutions or proposed slight changes to the original definition and ii) the opportunity of adapting the framework to the specific needs of its users. Among the several variants of i* framework are: Tropos [9], GRL [10] and Service-Oriented i* [4].

1.2 Problem statement

In recent years, considerable attention has been paid to organizational modeling techniques. This interest is motivated by the need of achieving a better understanding of the “*as-is*" and “*to-be*" of an organization with its social context, structure, processes and resources. Moreover, organizational models help representing business behavior. Views on models are built using graphical primitives, namely different symbols are used to represent roles, goals, resources, tasks and their relationships between them.

In the graphical description each business elements is labeled according to the information obtained during the elicitation of requirements.

Various problems have been detected due to weaknesses in this labeling activity. Labeling is not a rigorous task performed by the designers, and frequently it is carried out with certain freedom, resulting in situations of label inconsistency. Sometimes, large organizational models (depending on the domain and their description) become complex and inconsistent due to bad labeling and irrelevant information, creating difficulties in the explanation, analysis, and re-engineering of a model. This situation occurs also when the same label is used for different elements or different labels for describing the same element. Moreover, the amount of information that can be encoded in a human readable label is limited, affecting their understandability outside the organizational context.

At present, there are many researches that emerge using the annotation semantic and ontologies. In her PhD work, Chiara Di Francescomarino [11] describes BPMN processes enriched with semantic annotations belonging to one or more domain ontologies [11]. The argument of this work is that the process of annotating, with a set of labels taken from a set of domain ontologies, would provide an additional information to the models facilitating the activity of modeled.

While Lin et al. [12] presents the problem of semantic heterogeneity among the reasons for the difficulty to manipulate the process models in a centralized manner. Ontology-based semantic annotation is the solution presented in this work. The authors specify the advantages of using ontologies, namely: domain concepts. The domain concepts allow us to improve the labeling activity, an additional support to the business analysis during the modeling activity and reuse of information.

In this thesis we present an approach for enriching organizational visual models with annotations, characterized by a semantics encoded in a structured source of knowledge, such as a domain ontology. This enrichment of the organizational models allows us to clarify each element in the model through domain concepts, providing additional support to business analysis during modeling and enabling reuse of information.

The propose approach uses the i* framework, one of the most widespread goal-oriented modeling languages, and the two variants: Tropos and service-oriented i*. Our approach improves the label unification using domain concepts and abstracting them into meaningful generalizations. The goal is to obtain clear models accessible to humans and machines (thus enabling automated reasoning mechanisms). We do not plan to modify the initial organizational model because each element of the model preserves its original label. Ensuring the enrichment of models provides a precise and formal meaning of the elements of the model; such as the reuse of parts of the elements when creating new models, the detection of cross-item relationships and the achievement of semi-automated reasoning between the elements.

1.3 Proposed solution

The main goal of this thesis is to enrich the organizational models with annotations, characterized by means of an explicit semantic organized in a structured source of knowledge. This annotation provides a precise and clear formal meaning to the elements of the model. The idea is to obtain an organizational model integrated with a domain ontology in order to discover and implement services. This is achieved adding semantic annotation suggestions to the elements of the model by means of domain concepts. In this way, we propose to improve labeling activity among the i* variants, Tropos and Service-oriented i* adding annotations into the elements of model allowing the unification and categorization of the elements defined in different techniques.

In this context, the recent work on enrichment of models such as the enrichment of BPMN business process models with a domain ontology concepts through the semantic annotation of process elements and its formalization is proposed in [11]. In [13] is presented an extension of Formal Tropos (FT) to semantically enrich FT specifications with SBPM (Semantic Business Process Management) ontological annotations and to map these specifications to business process models. In this thesis, our objective is to enrich the organizational models taking benefits from the use of ontologies, namely allowing the standardization of concepts, improving the label of the elements and the reuse of information.

The process for enriching the organizational models with semantic annotation is based on two phases. The first phase corresponds to the "***Organizational model semantic annotation*"** which consists of representing an annotated model to iStarML file. In order to carry out this result, this phase is decomposed into two processes.

The first process "*Semantic Annotation suggestions development*" consists of developing a set of semantic annotation suggestions to guide the annotation process. The suggestions are applied to any domain ontology and to a general ontology. The second process *"Extension of iStarML"* consists of generating the annotated model to iStarML format. We propose to extend an existing plug-in for jUCMNav [14] and the iStarML interchange format. The extension consists of exporting the model to the iStarML format, adding the XML attribute "*sannotation*" to store the semantic annotation of each element of the model.

The second phase corresponds to the "***Integration of organizational model ontology with a domain ontology***". We propose to extend TAGOOn (Tool for the Automatic Generation of Organizational Ontologies [15]), in order to support the automatic transformation and integration from an i* base model represented with the variants: Tropos and Service-oriented i* into a domain ontology. The modules of "*Automatic parsing process*", "*Automatic linking process*" and "*Automatic Documentation Process*" are added. This integration consists of creating "*is a*" links among the domain ontology and the organizational model supported by semantic annotations.

The main solution of our approach is the development of semantic annotation suggestions. In order to carry out the suggestions three steps have been proposed.

The first process, called "*Semantic analysis of primitives of i*, Tropos, Service-oriented i**" analyze the semantic of each primitive in i*, Tropos and service-oriented i*. The primitives analyzed are: actor node and its types (agent, role, position) and each one of intentional elements (goal, softgoal, task, plan, resource, service and process). In spite of that Tropos and service-oriented i* used the primitives of the framework i*, these variants have its own definition of them. Therefore, we carry out a comparative analysis among primitives.

The analysis consists of examining and comparing the definition of the primitives for each variant.
The objective is to obtain a single definition for each primitive of the variants. The second process, which is called "*Analysis of general and domain ontologies*" consists of analyzing the hierarchy of several general and domain ontologies, in order to establish relationships between the definition of element of the model and the domain concepts.

The third process "*Development of semantic annotation suggestions*" consists of formally establishing each primitive into one or more domain concepts. The result of this step is a set of general semantic annotation suggestions and a set of specific semantic annotation suggestions. The first kind of suggestions can be applied to any domain ontology. The second kind can be applied to OntoSem and to its extensions.

1.4 Objectives

The main objective of this thesis is: "To *enrich the organizational models with annotations characterized by a semantics explicitly organized in a structured source of knowledge to provide a precise and clear formal meaning to the elements of model*". For the accomplishment of the main objective, the below specific objectives have been identified:

1. The development of an approach for building of ontologies integrated with an organizational model ontology.

2. The development of the semantic annotation suggestions to annotate the organizational model using the domain concepts.

3. The extensions of an existing plug-in to export an iStarML file adding the semantic annotation for each element of the model.

4. The application of the approach to the variants: i*, Tropos and Service-oriented i* integrating towards a domain ontology. This, by the extension of TAGOOn (Tool for the Automatic Generation of Organizational Ontologies) in order to demonstrate the effectiveness of the guideline.

5. Validate the whole semi-automated semantic annotation process on a set of examples.

1.5 Research design

The core of this thesis is the presentation of a guideline that formalizes organizational models enriched with semantic annotation from a source of knowledge. The semantic annotations of organizational models, in fact, can be used to provide a precise, formal meaning for the elements of the model, thus making them more understandable to people and allowing further analysis.

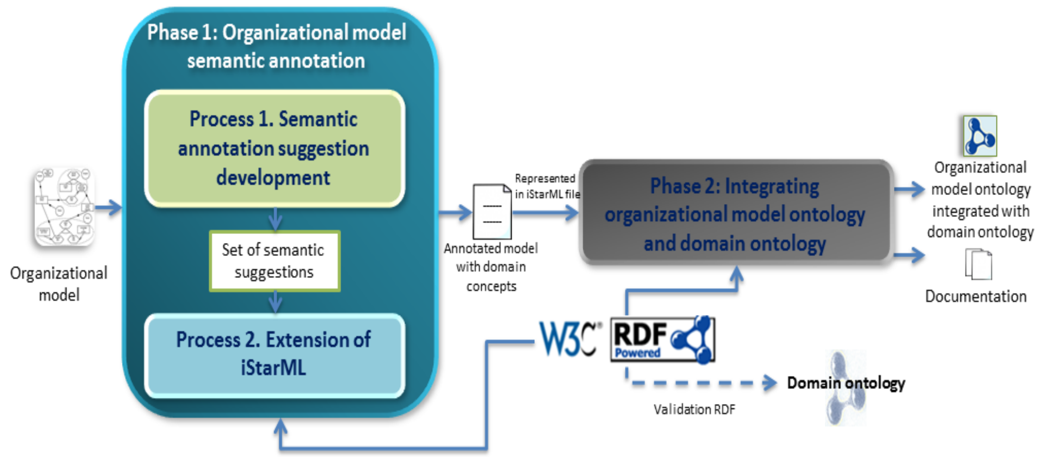


Figure 1.1 Developed processes in this thesis

For the accomplishment of this main objective we have identified three specific objectives represented in Figure 1.1. The first objective "***Semantic annotation suggestion development***" corresponds to the development of semantic annotation suggestions to annotate the elements of the model using domain concepts. These suggestions are classified in generals and specifics. The general semantic annotation suggestions are applied to any domain ontology and the specific semantic annotations are applied to OntoSem ontology. This phase is described in the section 4.2.1.

The second objective "**E*xtension of iStarML***" is related to the extension of a plug-in to export a model with semantic annotations to iStarML interchange format. We propose to add a XML attribute called "*sannotation*" to iStarML file. The value of this attribute stores the domain concept of each element of model. The input in this phase is a set of semantic annotation suggestions, the organizational model and the domain ontology. The output is an iStarML file annotated semantically. This phase is described in the Section 4.2.2

Finally, the third objective "***Integrating organizational model ontology and domain ontology***" corresponds to the development of a tool. The inputs of our tool are an organizational model represented in i*, Tropos or Service-oriented i* and the domain ontology. The outputs are an organizational model ontology integrated with a domain ontology represented in OWL and a text document that describes each element of the model with the domain concept is generated. This phase is described in the Section 4.4

1.6 Thesis outline

This thesis aims to enrich an organizational model with semantic annotations. We achieve a precise meaning to the elements of the annotated using ontologies. The structure of the thesis is presented in**¡Error! No se encuentra el origen de la referencia.**

Chapter II Background: Describes the important concepts used in this thesis. For example the visual modeling, ontologies and semantic annotation are presented in this chapter.

Chapter III State of the art: Introduces a brief overview of the state of the art from ontologies and semantic annotation process that are relevant to this research.

Chapter IV Organizational model Semantic annotation: Describes the guideline proposed to annotate the organizational model using domain ontologies.

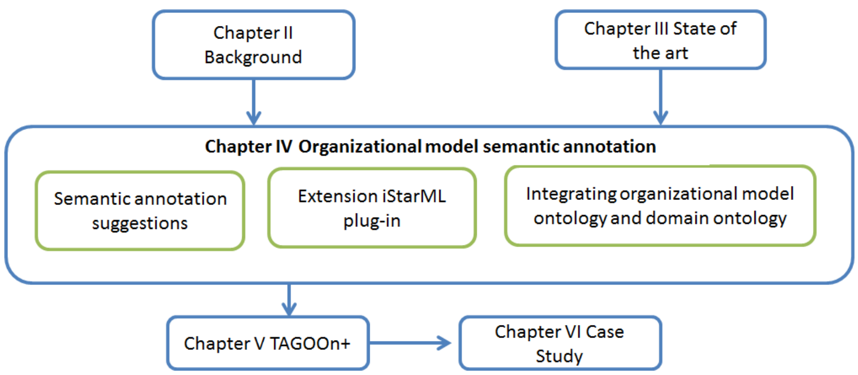


Figure 1.2 Thesis outline

Chapter V TAGOOn+: Describe TAGOOn+ tool and its module, these are: Mapping process from iStarML to OntoiStar, OWL file manager, Mapping process, documenting process and linking process.

Chapter VI Case study: Different cases of studying are presented. The organizational model are described in the variants i*, Tropos and Service-oriented.

Chapter 2 Background

This chapter describes the relevant concepts used in this thesis; the chapter is organized as follow: Section 2.1 presents an overview of visual modeling. Moreover, the framework i* [2], Tropos [9] and services-oriented i* [4] are explained in the same Section; Section 2.2 introduces the concept of ontology and its classification; finally, in Section 2.3 the semantic annotations and its applications are shown.

2.1 Visual modeling

Visual modeling is the graphic representation of objects and systems of interesting using graphical languages. The conceptual modeling by means the use of a diagrammatic visual notation (visual modeling), is a practice in software development, which become more and more popular. Perini and Susi in [16] mentioned that visual modeling has been recognized a different level relevance in software development, supporting the communication among stakeholders, involving the development process and the project documentation.

This visual modeling drives workflow in the model-based development approach to software engineering. A widely used visual modeling in object oriented software development is the Unified Modeling Language (UML) [17]. Although UML provides also basic mechanisms for its extension to include other non-object oriented concepts (i.e. an extensible metamodel) and for customizing it to specific domain (i.e. UML stereotypes), several modeling languages have been proposed for agent-oriented software development [18], goal-oriented analysis [6] and business process modeling [19].

Guizzardi et al [20] mentions that conceptual modeling is concerned with identifying, analyzing and describing the essential concepts and constraints of a domain with the help of a (diagrammatic) modeling language that is based on a small set of basic meta-concepts. Our thesis is addressed with conceptual modeling for organizations, a relevant aspect in business reengineering, and in requirements analysis for organizational information systems.

Nowadays is widely recognized the importance of the early requirements engineering techniques, also known as "*Organizational Modeling Techniques*" [1], during the development of information systems. In this context, the *i** framework is one the of most important techniques in modeling organizational proposed by Yu [2] for its ability to express explicitly social relations and intentional among stakeholders [3] and the representation of the internal behaviors required to satisfy actor dependencies [4] for this reasons this framework is a relevant paradigm in Requirements engineering and agent orientation in Software Engineering [5]. In [7],[8] is mentioned that throughout the years, different research groups have formulated variations this framework. There are basically two reasons behind this fact: i) The definition of the i* language is loose in some parts, and some groups have opted by different solutions or proposed slight changes to the original definition and ii) the adapting the framework to the specific needs of its users. Several variants of i* framework are: Tropos [9] and Service-Oriented i*[4]. Due to this the following section describes the organizational modeling techniques: i*, Tropos and Service-oriented i* which are the subject of our research study.

2.1.1 Framework i*

The *i** framework [2] is focused on the modeling of intentional, strategic actor relationships as a way of enriching models of organizations and organizational processes [2]. This framework incorporates goal- and agent-oriented modeling and reasoning tools, it has been a milestone for providing the basis, developing and spreading goal-orientation as a relevant paradigm in Requirements Engineering and agent orientation in Software Engineering [5].

*i** has spread and successfully been implemented in different contexts, e.g. organizational modeling, requirements elicitation, software design, and security [21]. Its explicit representation of goals and actors has allowed it to use it in Goal-Oriented Requirements Engineering (GORE) and Agent-Oriented Requirements Engineering (AORE).

2.1.1.1 i* Models

This framework allows for the clear and simple statement of actor's goals and dependencies among them. It also includes a graphical notation which allows for a unified and intuitive vision of the environment being modeled, showing its actors and the dependencies among them. *i** is characterized by defining of two models, each one corresponding to a different abstraction level: i) Strategic Dependency (SD) model represents the intentional level and ii) the Strategic Rationale (SR) model represents the rational level.

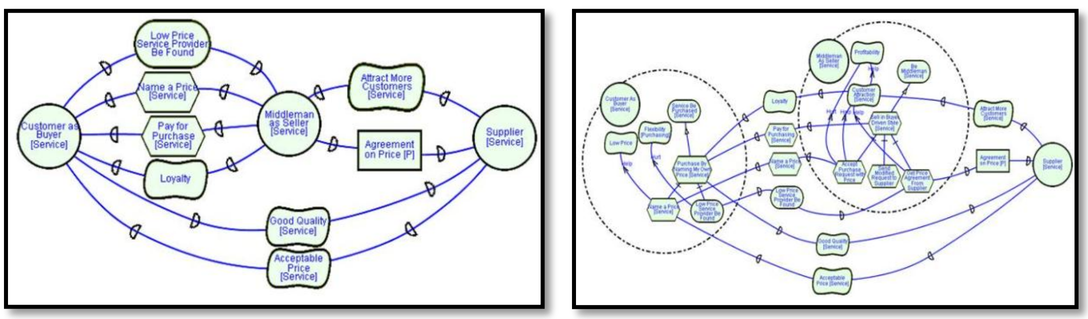
Strategic dependency (SD)

The SD model is used to express the network of intentional, strategic relationships among actors. SD diagrams depict the strategic dependencies between actors, but do not depict the internal rational behind these dependencies (Figure 2.1 incise a). The elements that characterize this model are below:

Actors: It is an active entity that carries out actions to achieve goals by exercising its know-how; an actor can be specialized into **agents**, **roles** and **positions**. A position covers roles. The agents represent particular instances of people, machines or software within the organization and they occupy positions.

Actor association links: The relationships between actors are described by graphical association links between actors. The types of actor association link are: **Is-part-of** it is used when an actor is part of another actor, **is a** it is used to represented a generalization, **plays association** it is used between an agent and a role, with an agent playing a role, **covers relationship** it is used to describe the relationship between a position and the roles that it covers, **occupies relationship** it is used to show that an agent occupies a role, meaning that it plays all of the roles that are covered by the position, **ins relationship** it is used to represent a specific instance of a more general entity.

Dependency: A SD model is formed by a set of nodes that represent actors and a set of dependencies that represent the relationships among them, expressing that an actor (**depender**) depends on some other (**dependee**) in order to obtain some objective (**dependum**). The dependum is an intentional element that can be a resource, task, goal or softgoal [2]. The types of strategic dependencies, based on the type of the dependum are: Goal dependency, Task dependency, Resource dependency and Softgoal dependency.



a) Strategic Dependency **b) Strategic Rationale**

Figure 2.1: i* models

Strategic rationale (SR)

This model consists of defining the internal operations that all actors carry out in order to reach its dependencies (**¡Error! No se encuentra el origen de la referencia.** Incise b). This graph contains our types of nodes (goal, task, resource and softgoal) and three types of internal links to i* actor (means–end link, decomposition links and contribution links). These elements are described below:

Boundary /Actor boundary: This element indicates intentional boundaries of a particular actor. All of the elements within a boundary for an actor are explicitly desired by that actor.

Intentional elements: An intentional element is an entity which allows it to relate different actors conforming a social network or, also, to express the internal rationality of an actor. This intentional elements are: a **goals** represents an intentional desire or strategic interest of an actor, a **softgoal** is similar to goals except that the criteria for the goal's satisfaction are not clear-cut, it is judged to be sufficiently satisfied from the point of viewing of the actor, a **task** represent the desire of an actor to accomplish some task, performed in a specific way and a **resource** represents information of information entities produce as result of the organization tasks.

Intentional element relationships: An intentional element link represents an n-ary relationship among intentional elements. The types of intentional element relationship are: Means-ends, decomposition links, contribution links.

Means-Ends links: These links indicate a relationship between an end, and a means for attaining it. The "*means*" is expressed in the form of a task, since the notion of task embodies how to do something, with the "*end*" is expressed as a goal. **Decomposition links:** A task element is linked to its component nodes by decomposition links. A task can be decomposed into four types of elements: a subgoal, a subtask, a resource, and/or a softgoal. The task can be decomposed into one to many of these elements. **Contribution links:** The contribution Links are: make, some+, Help, Break, some-, hurt, unknown, and, and OR. These contribution links can be used to link any of the elements to a softgoal and contributes to the satisfaction or fulfillment of the softgoal.

2.1.2 Tropos framework

Tropos is a software development methodology [22], where concepts of the agent paradigm are used along the whole software development process. The creators of Tropos [9] claim that this methodology is based on two key ideas. First, the **notion of agent** and all related mentalist notions (for instance goals and plans) are used in all phases of software development, from early analysis down to the actual implementation. Second, Tropos covers also the very **early phases** of requirements analysis, thus allowing for a deeper understanding of the environment where the software must operate.

Requirements analysis in Tropos is split in two main phases: "*Early Requirements*" and "*Late Requirements analysis*". This methodology introduces the five main development phases are: **early requirements** is identified the domain stakeholders and model them as social actors, who depend on one another for goals to be achieved, plans to be performed, and resources to be furnished, **late Requirements** analysis, the conceptual model is extended including a new actor, which represents the system, and a number of dependencies with other actors of the environment.

The **architectural design is** defined the system's global architecture in terms of sub-systems, interconnected through data and control flows. Sub-systems are represented, in the model, as actors and data/control interconnections are represented as dependencies, **design phase** this phase deals with the specification of the agents' micro level. Agents' goals, beliefs, and capabilities, as well as communication among agents are specified in detail, **implementation** activity follows step by step, the detailed design specification on the basis of mapping between the implementation platform constructs and the detailed design notions.

2.1.2.1 Tropos models

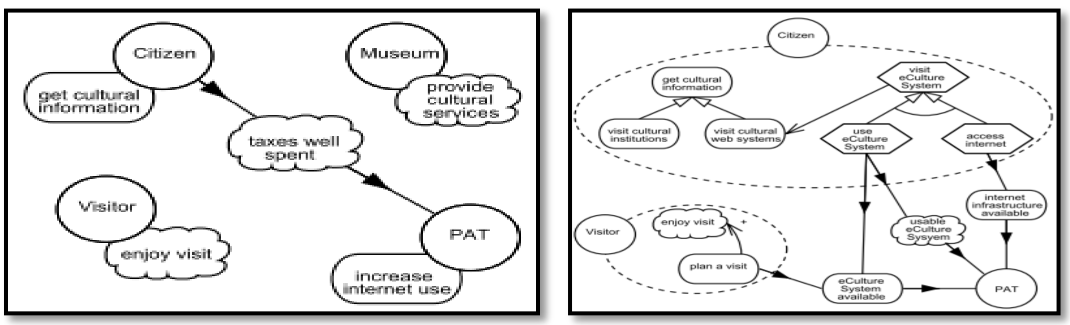
Tropos mainly purpose to define an agent-oriented software development method, using a variant of i* as modeling language. Two main types of diagrams are provided for visualizing the model: "*actor diagram*" and "*goal diagram*".

Actor diagram
In this model the nodes (the actors) are connected through dependencies (labeled arcs) (**¡Error! o se encuentra el origen de la referencia.** incise a). The elements that characterize this model are below:

Actor: It is an entity that represents a physical agent or a software agent as well as a role or a position. The types of actor are: agent, role and position.

Actor association links: An actor association link represents a relationship between actors. The types included this methodology are: occupies, covers, plays.

Dependency: A dependency between two actors indicates that one actor depends on another in order to attain some goal, execute some plan, or deliver a resource. A dependency in Tropos is equivalent to a dependency in i*. The types of dependencies are: Goal dependency, Task dependency, Resource dependency and Softgoal dependency. It includes a depender, a dependee and a dependum.



a) Actor diagram **b) Goal diagram**

Figure 2.2: Tropos models

Goal diagram
In this model represented as a balloon labeled with a specific actor name and containing goal and plan analysis, conducted from the point of view of the actor (**¡Error! No se encuentra el origen de a referencia.** incise b). The elements that characterize this model are below:

Boundary /Actor boundary: An actor boundary in Tropos is equivalent to an actor boundary in i* framework. However, when the actor boundary is expanded and its internal elements are associated to a dependency Tropos use the "*WHY*" label to express a link between an internal element and a dependency.

Intentional elements: An intentional element in Tropos is equivalent to an intentional element in i*. The types of intentional elements are: **goal**, **softgoal**, **plan** it is equivalent to a task in the i* framework, **resource**, **capability** it is represents the ability of an actor to define, choose and execute a plan to fulfill a goal, given a particular operating environment, **b*elief*** it is used to represent each actor's knowledge of the world.

Intentional element relationship: An intentional element relationship is equivalent to an intentional element relationship in i*. These are: Means-ends, decomposition and contribution links.

Means-Ends links: In particular, means-end analysis aims at identifying plans, resources and softgoals that provide means for achieving a goal. This means-ends link is equivalent to means-ends in i*. **Decomposition links:** This links represents the decomposition of goals, softgoal and plan, i.e. subgoals and subtask. The decompositions can be to define how AND-decomposition and OR-decomposition.

Contribution links: it is identifies goals that can contribute positively or negatively in the fulfillment of the goal to be analyzed. The contributions links are: ++, +,--,-. Any these links can be used to link any of the elements to a goal of softgoal to model the way any these elements contributes to the satisfaction or fulfillment of the goal or softgoal.

2.1.3 Service-oriented i* framework

This framework is the result of revisiting and extending the semantic of the i* modeling concepts, according to [4] despite the well-known advantages of the i* modeling approach, there are certain issues that still need to be improved to assure its effectiveness in practice. In this methodology, the modeling process starts by considering the enterprise as a service provider and by eliciting the services that the enterprise offers to end customers.

The following step consists of determining the way in which the business services satisfy the goals of the enterprise. Once the services have been elicited, is needed to refine each service in the set of business processes needed to perform it. As a result of this new approach, the mechanisms for decomposition, refinement, and modularity are focused on business services.

The proposed architecture distinguishes three abstractions levels (services, process and protocols) and describes a methodological approach to align the business models produces at these abstraction levels.

2.1.3.1 Service-oriented i* models

The key idea of the Service-oriented approach is to use the business services as building blocks that encapsulate internal and social behaviors. Therefore, complementary models were defined to make it possible to reify the abstract concept of service in low-level descriptions of its implementation. The main idea of this methodology is to promote the granularity of the service definition by isolating the organizational behavior of each business service in a separate business description [4], due to this the models are: global model, process model and protocol model.

Global Model

The global model permits the representation of the business services and the actor that plays the role of requester and provider. The global model has two different views: **abstract view** only shows the actor and its offered business services and **concrete view** the offered business services are linked with the internal goals of the provide actor (Figure 2.3). The elements that characterize this model are below:

Business actor: It is an independent intentional organizational entity (person, functional area, department, or enterprise) that uses or offers services. The actor could be specialized into agent, roles and positions.

Actor association links: The relationships between actors are described by graphical association links between actors. The types of actor association links are equivalent to an actor association link in the i* these are: **Is-part-of, is a**, **plays association**, **covers relationship**, **occupies relationship**, **ins relationship**. Additionally it includes the **subordination** this implies that if one actor subordinates to another actor, then the first one is responsible for the behavior of the second and it can implement monitoring mechanisms to control and evaluate the subordinated actor's work

Dependency: This kind of relationship must be used to represent the delegation of responsibilities between actors. A dependency in this framework is equivalent to a dependency in i*. It includes a depender, a dependee and a dependum. The additional dependency in service-oriented i* is **service dependency** is created between the enterprise and the customers.

Business services: It is a self-contained, stateless business functionality that is offered to potential customers through a well-defined interface. A business service should be viewed as an abstract set of business functionalities that are provided by a specific actor. There are basic services and composite services. A basic service is decomposed in processes without further decomposition. A composite service aggregates multiple services and implements mechanisms that coordinate the aggregated services.

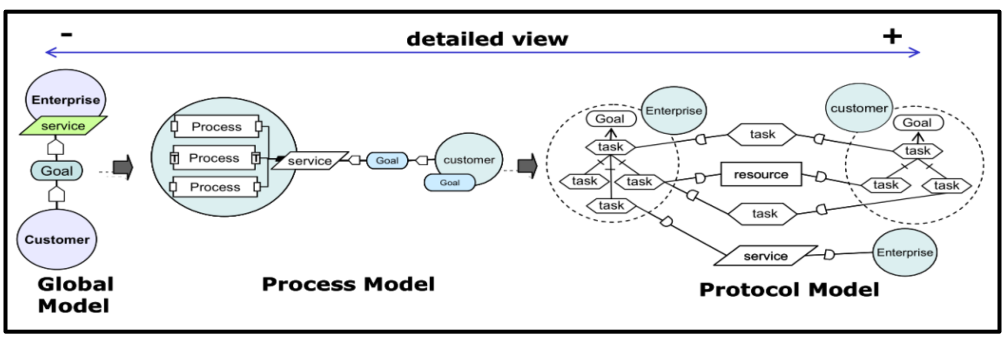


Figure 2.3: Services-oriented i* models

Process model

The process model represents the functional abstractions of the business process for a specific service. This model provides the mechanisms required to describe the flow of multiple processes. A process model represents a view of the processes needed to satisfy a service but without giving details of its implementation. The elements that characterize this model are below.

Boundary /Actor boundary: This element is same to a boundary in i* framework.

Business process: This concept represents a set of structured activities for producing a specific business service for a particular customer. A process can be transactional or no transactional. Some aspects to consider in the definition of the business processes are process composition, process delegation, actor composition, process visibility.

Relationships: In this framework there are additional relationships between the concepts de business and process. These relationships are: **Service relationship** this relationship connects a composite service with multiple basic services. There are four ways to connect the services: mandatory, optional, alternative, or. **Service-goal relationship** this relationship indicates that a service is associated with a specific goal of the provider of the service, **process relationship** this indicates that a process depends of other process to be executed and the p**rocess dependency** this represents the process association with a specific service.

Protocol model

The protocol model provides a description of a set of structured and associated activities that produce a specific result or product for a business service. This model is represented using the redefinition of the i* modeling primitives. The elements that characterize this model are below.

Intentional elements: An intentional element in this framework is equivalent to an intentional element in i*. The types of intentional elements are: **goal**, **softgoal**, **task, resource,**

Intentional element relationship: An intentional element relationship is equivalent to an intentional element relationship in i*. These are: Means-ends, decomposition and contribution links.

Means-end link: This relationship should be used when there is enough evidence to assure that the alternative subcomponents (means) fully satisfies the root component (end). **Decomposition links**: plan, goal or softgoal can be root and a sub element of the same type as leaf, i.e. task to task, goal to goal and softgoal to softgoal. This relationship has a semantic of AND-decomposition or OR-decomposition. **Contribution link:** This permits the analyst to represent partial and full satisfaction relationships among instances of modeling concepts. The contribution Links are: ++, +, --, -.

2.2 Ontologies

Ontology is "*explicit specification, formal and shared conceptualization*" [23]. Explicit means that the type of concepts used are explicitly defined; this is that if other concepts can also describe the same type, defined in detail. Formal refers to the fact that the ontology should be machine readable, such as it is stored in a digital format. This concept is based on the idea of a simplified conceptualization of the world. According to [24] an ontology differs from existing methods and technologies in the following way: (i) the primary goal of ontologies is to enable agreement on the meaning of specific vocabulary terms and, thus, to facilitate information integration across individual languages; (ii) ontologies are formalized in logic-based representation languages. Its semantic is thus specified in an unambiguous way. (iii) The representation languages come with executable calculi enabling querying and reasoning at run time.

Ontology is defined in terms of concepts and relationships, the concepts capture the entities of the domain under consideration. This knowledge is decomposed by means concepts, so that the knowledge representation becomes the representation of concepts that are some as interrelated and generate such knowledge or idea about the domain. Knowledge in ontologies is mainly formalized using five kinds of components: classes, relations, functions, axioms and instances.

2.2.1 Ontology categories

Ontologies are considered as key elements for semantic interoperability and to share vocabularies for describing information relevant to a certain area of application. The rise of ontologies is because it facilitates interoperability [25]. From the content viewpoint, instead, ontologies can be classified according to its generality [26] such as: **Top-level ontologies** describe very general concepts like space, time, matter, object, event, action, etc., which are independent of a particular problem or domain.

Domain ontologies and task ontologies describe, respectively, the vocabulary related to a generic domain (like medicine, or automobiles) or a generic task or activity (like diagnosing or selling), by specializing the terms introduced in the top-level ontology. **Application ontologies** describe concepts depending both on a particular domain and task, which are often specializations of both the related ontologies. These concepts often correspond to roles played by domain entities while performing a certain activity, like replaceable unit or spare component. In Figure 2.4 the categories ontologies is shown.

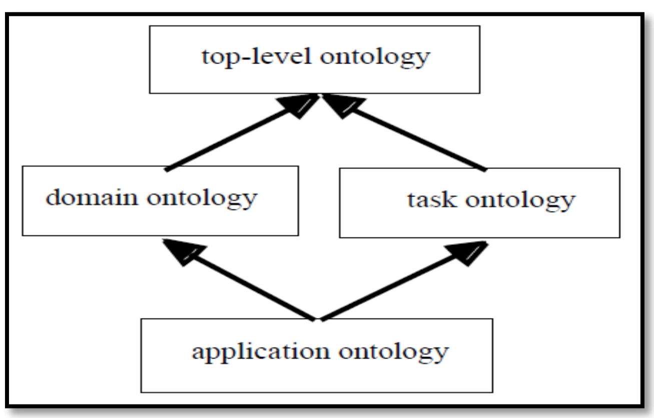


Figure 2.4: Ontology categories [26]

2.2.2 Mapping approaches

Ontologies are considered a key element for semantic interoperability and act as shared vocabularies for describing the relevant notions of a certain application area, whose semantics is specified in a (reasonably) unambiguous and machine processable form [27]. However, there is more of an ontology to the same domain. Due to this point, researchers groups are focusing to create a "*bridge*" among scattered ontologies. There are several approaches for ontology linking; these approaches include composition, merging and mapping.

Composition this approach is defined as composing a new ontology by reusing an existing ontology. This concept is especially relevant when one considers the creation of "*utility*" ontologies of commonly used concepts. **Merge** is defined as to create a single coherent ontology that includes the information from all the sources. **Mapping** this approach includes mapping to a standard upper ontology, to a common upper ontology, to reference ontology, or directly from one domain ontology to other domain ontology. Ontologies can also mapping to a reference ontology that includes key concepts, but no instance data.

2.3 Semantic annotation

The semantic annotation is clearly specification, easy to understand, enables several advanced analyses and manipulations [19]. Semantic Annotation helps to bridge the ambiguity of the natural language when expressing notions and its computational representation in a formal language. By telling a computer how data items are related and how these relations can be evaluated automatically, it becomes possible to process complex filter and search operations [28].

2.3.1 Applications

Semantic annotations can be added to documents, web pages, models, text in databases, to any sort of text, aim order to define or clarify its hidden semantics [19]. The most important application of the semantic annotation is the semantic web. The semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation [29].

The idea the semantic web is driving the evolution of the current Web by enabling users to find, share, and combine information more easily. The Semantic Web, as originally envisioned, is a system that enables machines to "*understand*" and respond to complex human requests based on its meaning. Such an "*understanding*" requires that the relevant information sources be semantically structured, a challenging task.

The Semantic Web involves publishing in languages specifically designed for data: Resource Description Framework (RDF), Web Ontology Language (OWL), and Extensible Markup Language (XML). These technologies are combined in order to provide descriptions that supplement or replace the content of Web documents. In this way, the machine can process knowledge itself, instead of text, using processes similar to human deductive reasoning and inference, thereby obtaining more meaningful results and helping computers to perform automated information gathering and research. The semantic annotation join with the ontologies are being applied in different scope, for example to annotate: Geo-services [30], business process models [11], Web Feature Services [31], on-line glossaries [32], textual documents [33] and others.

2.3.2 Annotation Tools

The semantic annotation can be classified by the degree of automatizing annotation task. We can distinguish: manual systems, semi-automatic and automatic. **The manual annotation system** permits the user to see and to navigate simultaneous to ontologies and web resources, using the knowledge of ontologies by adding annotations to the resources. In the **semi-automatic annotation systems** the agents Web can be designs to try the information of pages Web semi-automatically. Mediate technics of Natural Language Processing can be extracted references from the text to domain concepts.

These systems generally require a certain amount of manually annotated documents, from which the system can be trained. **Automatic systems** are tools used technics of information extracting of natural language to annotate automatically in pages Web. However, these tools are not reliable totally. Nowadays there are many tools that permit to annotate using the linguistic analysis; a table comparative of several annotation tools is presented below.

Table 2-1 Comparison of annotation tools

Annotation Tool	Spaces technology	Ontology support	Automatic tool	Type of analysis	Features
Annotea [34]	RDF schema, Xpointer,	Annotation server	No	Similarly a bookmark schema	System for creating and publishing shareable annotations of Web documents.
SHOE [35]	SHOE	Prompting	No	Running SHOE (wrappers)	System which allowed users to mark-up HTML Pages.
CREAM [35]	RDF, OWL, XPointers		Semi	Annotating of databases	Application that considered the possibility of annotating the deep web.
OWLIR [36]	DAML+OIL, DAMLJessKB.		Semi	Inference	An approach for information retrieval over documents.
SMORE [37]	RDF, OWL	Web browser & editor	Semi	Screen scraper	Application designed to enable users to markup HTML documents in OWL using Web Ontologies.
APOLDA [38]	OWL	OWL annotation properties	Yes	Lexical denotations	Application by annotation texts with labels of concepts from an arbitrary OWL-ontology.
Armadillo [39]	RDF	RDF ontology and a knowledge base.	Yes	String matching, POS tagging, Named Entity Recognition	System for unsupervised creation of knowledge bases from large repositories as well as document annotation.
KIM [40]	RDF, OWL	Upper-level ontology KIMO	Yes	String matching, POS tagging, Named Entity Recognition	Automatic semantic annotation, Indexing, and retrieval of documents.
Melita [41]	RDF, DAML+OIL	Control of intrusivenes s of IE	Yes	String matching, POS tagging, Named Entity Recognition	Tool for the definition and development of ontology-based annotation services.
OntoMat [42]	DAML+OIL, OWL, SQL, XPointer	OntoBroker annotation inference server	Yes	Drag&drop interactions	A user-friendly interactive webpage annotation tool.
PANKOW [43]	OWL	TAP ontology	Yes	Exploits surface patterns	Application that categorize automatically named entities found in text with respect to a given ontology
Seeker [44]	RDF	TAP ontology	Yes	Similarity, TBD	Platform for large-scale text analytics
SemTag [44]	RDF	TAP ontology	Yes	Seeker, similarity, TBD	Application that performs automated semantic tagging of large corpora

2.4 Summary

This chapter introduced the main concepts and notations necessary in the remainder of the thesis. An overview of the visual modeling has been introduced. The i* framework and its main variants Tropos and service-oriented were presented, describing the primitives and the models of each model. Then ontologies and its categories were introduces. The mapping approaches and its main features were described. Finally the semantic annotation and its applications were presented. Several annotation tools and a summary table of features of these tools were also presented.

Chapter 3 State of the art

3.1 Introduction

This chapter introduces a brief overview of the state of the art in the research areas that are considered to be relevant to this research. In the Section 3.2 analysis criteria are presented to evaluate the applicability of these approaches. Section 3.3 address the topic related with semantic annotation in organizational models. The first two proposals have the objective of enriching models with semantic annotation and to derive business models and the third proposal enriching process elements with domain concepts are presented.

Section 3.4 address the topic related with interoperability problem using ontologies. Firstly proposal fusion two ontologies and the other proposals are focused in the problem of semantic heterogeneity and proposed to used domain ontology is presented. Section 3.5 is addressed the topic related with semantic annotation of documents. The proposals provide a way to annotate documents proposing labels to add the annotation are presented. Finally, in Section3.6 a summary of the proposal is presented according the criteria to illustrate the relevance of each related work to this thesis.

3.2 Analysis criteria

Each related work presented in state of the art has been described according analysis criteria to evaluate the applicability of the works to this thesis. With this purpose to carry out the description of each work, we have identified the next criteria: scope, objective of the approach, resources, type of processing, annotated resource, and technology space and proposal label. The analysis criteria are detailed below. It is important to note that if one criterion is not applicable to a specific work, it is omitted.

Scope: this criterion identifies the field or context in which it develops throughout the investigation.

Objective of the approach: this criterion describes the objective of the approach of the research work. This is important for given to the reader a feeling for what the related work is all about.

Resources: this criterion shown the modeling languages, ontology languages, or languages used in the related work.

Type of processing: this criterion indicates if the research presents a focus manual, semi-automatic or automatic.

Annotated resource: this criterion presented the resource which is added the annotation in the related work.

Technological Space: The term "*technological space*" refers to the different technologies (hardware, software) used in the research. This criterion identifies languages, techniques used in the research reviewed, such as MDA, XML, etc.

Proposed label: this criterion show the tag on which should carry out the semantic annotation.

3.3 Generate business models via semantic annotation

The i* framework [2] is an organizational modeling framework that supports a representation of the social, intentional, and strategic aspects of organizational structures. Many research groups have contributed and have extended this framework, due to this several variants have been proposed, such as Tropos [45], service-oriented i*, and others.

Semantic annotation is clear specification, easy to understand, and can serve as a basis for number of useful applications. However, in the context of Semantic Business Process Management there is a current lack of requirements engineering methodologies to acquire correctly semantically annotated business process models. Three proposals that generated business models via semantic annotation are presented in this section.

3.3.1 Mapping semantically enriched Formal Tropos to Business Process Models

In this work [13] the authors are focused on Goal-Oriented Requirements Engineering methods related to the SUPER platform, which support the vision of Semantic Business Process Management. This proposal mentioned that there is a current lack of requirements engineering methodologies to acquire correctly semantically annotated business process models. The objective in this research is an extension of Formal Tropos (FT) to semantically enrich FT specifications with SBPM ontological annotations and map these specifications to business process models.

The annotating FT specifications with SBPM concepts achieved using there SBPM ontologies (domains, functions and process goals). The ontologies of organizational models are written using the Web Service Modeling Language (WSML). In this work the authors proposed to insert references to the matching SBPM concepts into the FT code by means of attributes, this tag is: "*SBPM_annotation*"; whereas type have the format "*OntologyName#ConceptName*". For instance, is proposed to annotate a business function X to an actor Y without mentioning the targeted BPMO relationship "*actor Y hasBusinessFunction function X*". Detailed mapping between FT and BPMO are summarized in Table 3-1

Table 3-1: Mappings between FT and BPMO

FT construct	BPMO construct
Actor	Actor & Element from BusinessFunctionsOntology
Goal	Process (Top Goal) / SubProcess (Children)
Task	GoalTask / WebServiceTask / ManualTask
Softgoal	Element from BusinessGoalOntology
Resource	Element from BusinessDomainOntology
Entity	Element from BusinessDomainOntology

This work is important because is described the creation of a set of rules for mapped between FT and BPMO, and a set of general and specific semantic suggestions are the guidelines to integrate a domain ontology and an organizational model were proposed in our research.

3.3.2 Actor Eco-systems: From High-level agent models to executable process via semantic annotations

In this work [46] the authors describe how semantic annotation of abstract models of actor eco-systems can be used to derive executable process models that realize such systems. In this approach used semantic annotations for i* models in order to obtain a high-level description of the sequencing in the underlying processes. The objective is to describe actor eco-systems using high-level abstractions, requirements and artifacts, and obtain from such representations executable artifacts (such as programs, or business process).

The formal analysis and design or organizations used Tropos methodology specifically Formal Tropos (FT). The notations can be formal (for instance, in first order logic) or informal (via Controlled Natural Languages (CNL)). The formal annotations are proposed use automated reasoners, while informal annotations should analyze to check for consistency.

The transformation of actor ecosystems via BPMN is supported by applying well known planning techniques. In this approach an annotated BPMN model, is one in which every task (atomic, loop, compensatory or multi-instance) and every sub-process has been annotated with descriptions. The verification of a business process model with a set of compliance rules, the aim is to verify the consistent it. In this research is assumed that the effect annotations have been represented in Conjunctive Normal Form or CNF. This work is relevant for this thesis because the idea of verification of the semantic annotation is carried out by rules and the analysis of each process elements is useful for the mapping process implemented in our approach using axioms and domain concepts.

3.3.3 Semantic annotation of Business Process Models

In this work [11] the author propose to enrichment of BPMN business process models with domain ontology concepts, by means of the semantic annotation of process elements and the formalization of such information, as well as of process structural information, in a knowledge base (Figure 3.1). In this research proposed a technique for the reverse engineering of BPM, such as to investigate the use of process metrics as early indicators of the recovered process model quality. The author defined a visual language (BPMN VQL) to query business process models and document scattered and tangled business concerns. A technique is proposed (based on Formal Concept analysis) for the semi-automatic retrieval and documentation of crosscutting concerns in semantically annotated business process.

An aspect-oriented language (BPMN VRL) is defined to modularize crosscutting concerns in process models. In this proposal is suggested to enrich BPMN business process with domain annotations, thus clarifying the process domain semantics and to encode the annotated process into an OWL knowledge base, thus providing a starting point for exploiting reasoning on the processes. The research proposed the use of linguistic analysis of the process element labels and of the concept names for providing semantic annotation suggestions to business designers. The label proposed to add the semantic annotations to process elements is "*bpmn:TextAnnotation*".

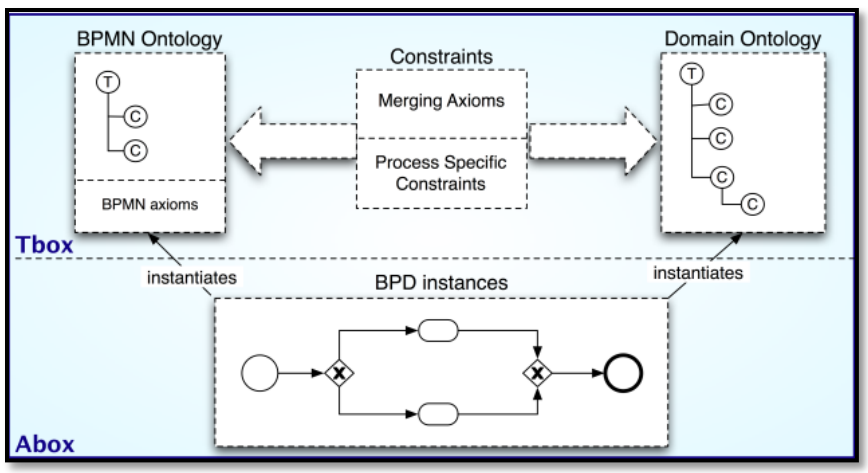


Figure 3.1: The Business Process Knowledge Base

This research is relevant for this thesis because the methodology followed for the semantic annotation and labeling the process elements is useful for the carry out the guideline proposed to add semantic annotation using domain concepts and proposed a set de semantic suggestions and the label "*sannotation*" in the elements of the models.

3.4 Dealing with interoperability problem using ontologies

Ontology is "*explicit specification, formal and shared conceptualization*" [23]. Explicit means that the type of concepts used are explicitly defined; this is that if other concepts can also describe the same type, defined in detail. Formal refers to the fact that the ontology should be machine readable, such as it is stored in a digital format. This concept is based on the idea of a simplified conceptualization of the world. In the section four proposals that used the advantages of the ontologies are presents. Firstly proposal fusion two ontologies and the other proposals are focused in the problem of semantic heterogeneity and proposed to used domain ontology is presented.

3.4.1 Ontology fusion using semantic properties

In this research [47] the author presents a process for ontology merging which is automatic and robust. Automatic since the computer detects and solves the problems arising during the fusion and robust because merging occurs in spite of ontologies being mutually inconsistent and present information from different viewpoints. The efficiency of this algorithm is shown by converting by hand several documents in internet to ontologies in this notation, and the automatically fusing them. The technologies space in this work is XML.

In this work resolve the problem of merging ontologies (two ontologies) and to build a new ontology, this new ontology contained all the information of both ontologies without repetitions or contradictions. In this research is developed the language Ontology Merging (OM) the aim is to design ontologies with concepts and relationships that contained more semantics (Figure 3.2). This approach suggests the use of the label *"<concept/>"* to add the semantic annotation.

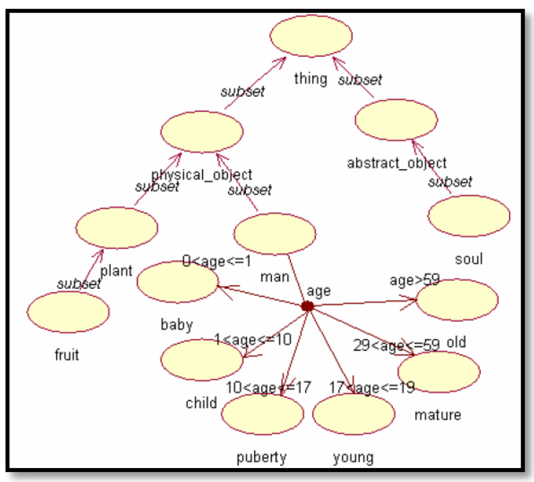


Figure 3.2: An ontology with the OM annotation

This work is relevant in our thesis because is presented a methodology to merge two ontologies and obtain a new ontology is useful for the development of our proposed. Our proposed join an organizational model ontology and a domain ontology preserving its original domain concepts through domain concepts.

3.4.2 Applying the UFO Ontology to Design an Agent-Oriented Engineering Language

In this work [48] the authors describing the application of a foundational ontology named UFO in the design of an agent-oriented modeling language for the ARKnowD (Agent-oriented Recipe for Knowledge Management System Development) methodology, combining two different approaches, namely Tropos and AORML (Agent-Object-Relationship). This research proposes some mapping rules between the notations, inspired in the Model Driven Architecture (MDA) metamodel transformation method; this permitted to guarantee a smooth transition from Requirement Analysis to System Design.

In this approach, for mapping the two notations, a theoretical analysis was made with the use of UFO foundational ontology. Then a set of rules in order to map from the modeling constructs (Tropos notation) to the destiny language (AORML) is proposed Table 3-2). Then to provide automated support to ARKnowD, is proposed in order to integrate AORML into an existing Tropos modeling tool name TAOM4E [49], implementing the mapping of a Tropos Actor Diagram into an AORML agent Diagram.

Table 3-2: Mapping Tropos into AORML

Tropos Concepts	AORML Constructs
actor	agent
plan	interaction
resource	object
dependency	relationship
delegation	relationship and commitment
resource acquisition	relationship and commitment

This work is relevant for our approach because is proposed a methodology to map de Tropos into AORML thought ontologies and supporting by a set of mapping rules is useful this idea for the development of our research. In our case is proposed a guideline to annotate the organizational model using domain concepts establishing a set de suggestions.

3.4.3 Semantic annotation framework to manage semantic heterogeneity of process models

In this research [12] the authors describing a semantic annotation framework to manage the semantic heterogeneity of process models. In this work is presented the problem of semantic heterogeneity how a difficult to manipulate the distributed process models in a centralized manner. Ontology-based semantic annotation is the solution presented in this work. The process consists of a basic description of process models (profile annotation), process modeling languages (meta-model annotation), process models (model annotation) and the process models (goal annotation). This framework consists of extending and refining General Process Ontology (GPO).

There are some metadata elements from the Dublin Core metadata is used, and then is proposed to create also additional metadata with prefix “*profileAnno*” to describe the profile of a process model (Figure 3.3. This is used to align the heterogeneous meta-models of process models, a set mapping rules between process modeling language constructs or meta-model elements and GPO are proposed. The mapping rules consist of both one-to-one and one-to-many correspondences between GPO concepts and modeling language constructs or meta-model elements. A namespace "*metaAnno*" is used to encode meta-model annotation. In this work the domain ontology for model annotation and goal ontology for process goal annotation is used. The main contribution of this work is the formal process semantic annotation model (PSAM).

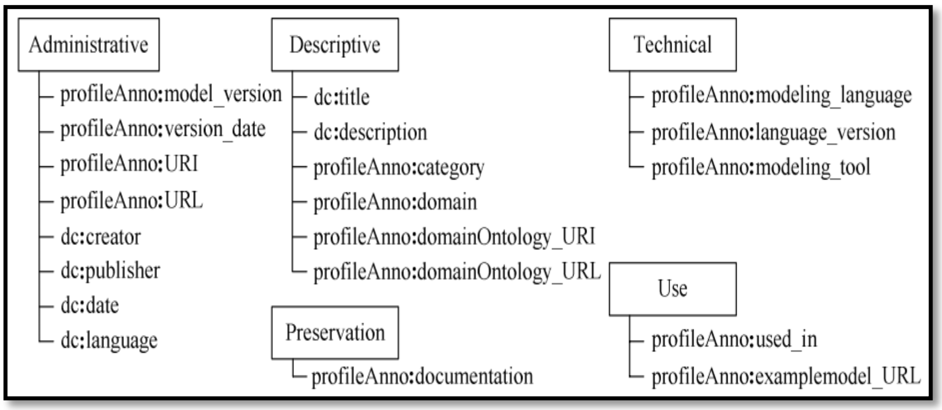


Figure 3.3: Profile annotation metadata elements

This research is relevant for this thesis because the methodology followed for the development of the mapping rules is useful for the development of our guideline to add semantic annotation of the organizational models.

3.4.4 SEAN: Multi-ontology semantic annotation for highly accurate closed domains

In this research [50] the authors propose SEAN a global framework for multi-ontology semantic annotation. This framework is based on the manual semantic annotation of documents associated with entities. This work is focused the notation of highly accurate close domains (HACD) as a set of domains with a minimal semantic model of concepts, that is a domain which can be very accurately defined by a set of concepts and can be very easily annotated manually. The annotation is based on a common vocabulary.

SEAN implements this common vocabulary as two groups of ontologies. On the one hand, an application ontology which describes the different products that can be associated with projects, while on the other hand, a domain ontology which relates the products with terms of the domain to which the project belongs. The domain ontology provides the common concepts which can be used to describe each of the elements generated.

The steps to annotation the process are: creation of a project, definition of products and related products and definition of the key words of the domain Figure 3.4). The layers SEAN architecture are: i) Annotation GUI using AJAX technology in Java environments, ii) Retrieval GUI provided by SPARQL and query RDF triples, iii) Reasoning engine using Renamed Abox and Concept Expression Reasoner (RACER) iv)Query engine used SPAQRL RDF, OWL DL and JENA.

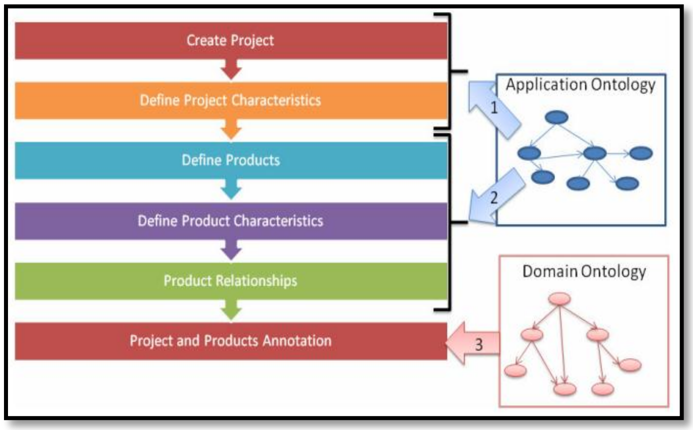


Figure 3.4: SEAN annotation process

This work is relevant for our approach because it is based on the potential for well-defined domains semantic annotation, consensus sharing and minimal semantic complexity applied to a given domain. This idea is useful due to we proposed to add domain concepts into organizational models using a set of semantic suggestions to clarify and to understand the models.

3.5 Semantic annotation of documents

Enrichment of text documents with semantic metadata reflecting its meaning facilitates document organization, indexing, retrieval, categorization, generation of more advanced metadata, smooth traversal between unstructured text and available relevant knowledge. The semantic annotation is applicable to any sort of text-web pages, regular (no-web) documents, text fields in databases, etc. In this section several proposal related with semantic annotation of textual, web document and web services are presented.

3.5.1 Cerno: Light-Weight Tool Support for Semantic Annotation of Textual Documents.

In this work [33] the authors describes a framework for semi-automatic semantic annotation of textual documents according to a domain-specific semantic model. This idea is founded on light-weight techniques and tools intended for legacy code analysis ad markup. In this framework the semantic model is defined in terms of UML class diagrams, and then this approach analyzes text to determine where to introduce annotation by exploiting software source code analysis tools and techniques from Reverse Engineering.

The Cerno framework consist of i) a semantic process for defining keyword and grammar-based rules for identifying instances of concepts in a text, and ii) an architecture based on software design recovery for applying the rules to mark up and extract identified instances in a document set. This work used TXL is a programming language for expressing structural source transformations from input to output text. The architecture of Cerno is: Parse, Markup and Mapping (Figure 3.5). This approach took advantage of WordNet and on-line Thesaurus, and the tool Protégé 3.0. Cerno was used to support requirements extraction from system descriptions in natural language.

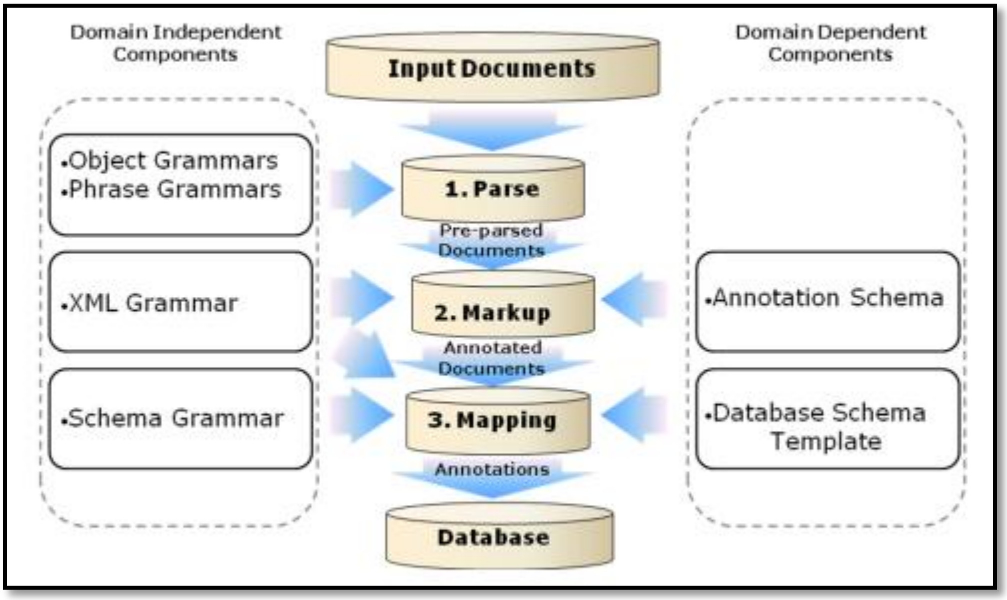


Figure 3.5: The semantic annotation architecture and process in Cerno

This work is relevant for our approach because the architecture of three layers of this approach is useful for the development of our research. We took this design to apply in our architecture of three levels.

3.5.2 From manual to semi-automatic semantic annotation: About ontology-based text annotation tools

In this work [51] the authors describes in ontology-based semantic annotation, which is embedded in a scenario of a knowledge portal application. This idea is founded in to conceive semantic annotation as a cyclic process between the actual task of annotation documents and the development and adaptation of domain ontology. The objective of this approach is to develop an ergonomic knowledge base-supported annotation tool, this is to support for the KA-initiative (Knowledge Annotation initiative of the Knowledge Acquisition community).

The idea behind this approach is to analyze the occurring words of a domain-specific corpus with its corresponding frequencies. In this work firstly is presented an approach of semantic annotation manual and the based on the experiences of the authors, proposed a semi-automatic annotation. The steps of this work, first the documents are processed using the information extraction system SMES (Saarbrücken Message Extraction System), this associates single words or complex expressions with a concept from the ontology, connected by means of domain lexicon linkage.

Then recognized concepts and dependency relations between concepts are highlighted as suggested annotations (Figure 3.6). In this approach was extended the engineering toolkit

OntoEdit by sem-automatic means for extracting and maintaining ontologies by analyzing existing data, this part is called "*ontology learning*". An important aspect of this work was that in parallel, linguistic resources are gathered, which connected the conceptual structures with the information extraction system. Thus, the ontology learning mechanisms support the engineering of evolving ontologies as well as the process incrementally improving the performance of the information extraction system for the semi-automatic annotation task.

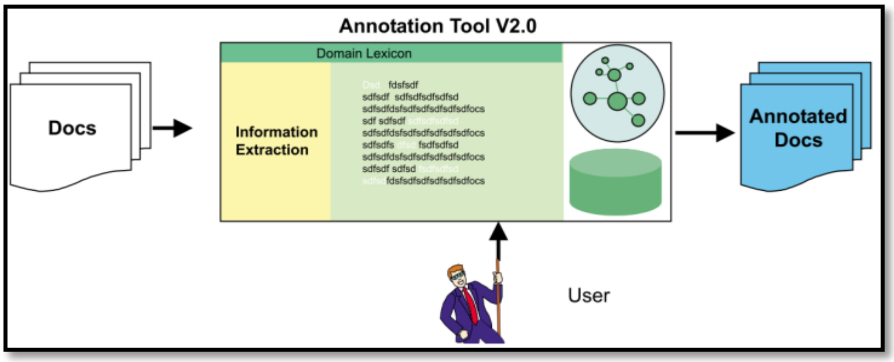


Figure 3.6: Semi-automatic annotation

This work is relevant for our approach because in this analyze the occurring words of a domain-specific corpus with its corresponding frequencies this is useful for the development of our research, we proposed to analyze the definition of each intentional element in the organizational model and then is suggested semantic annotation for general and specific ontology.

3.5.3 Semantic annotation platform

In this work [40] the authors describe a novel knowledge and information management infrastructure and services for automatic semantic annotation, indexing, and retrieval of documents. This approach uses an upper-level ontology and a knowledge base, these including RDF(S) repositories, ontology middleware and reasoning. This approach permits an automatic semantic annotation. KIM is based on GATE (General Architecture for Text Engineering) and SESAME.

The KIM platform consists of KIM Ontology (KIMO), knowledge base, KIM Server (with API for remote access, embedding, and integration), and fronts-ends (it is equipped with a plug-in for the Internet Explorer browser, KIM web user interface with various access methods, and knowledge Explorer for KB navigation). KIM ontologies and knowledge bases are kept in semantic repositories based on cutting edge Semantic Web technology and standards, including RDF(S) repositories (SESAME) and ontology middleware. Moreover, this approach used Lucene engine, the information retrieval for Lucene is used to index documents by entity types and measure relevance according to entities, along with tokens and stems. The semantic annotation in this research is based on the hypothesis that the named entities mentioned in the documents constitute important part of its semantics, this annotation consists of assigning to the entities in the text links to its semantic descriptions. The idea of this sort of metadata is to provide both class and instance information about the entities referred in the documents. In Figure 3.7 the sequential processing of content to the point where semantic annotations are produce over it is shown.

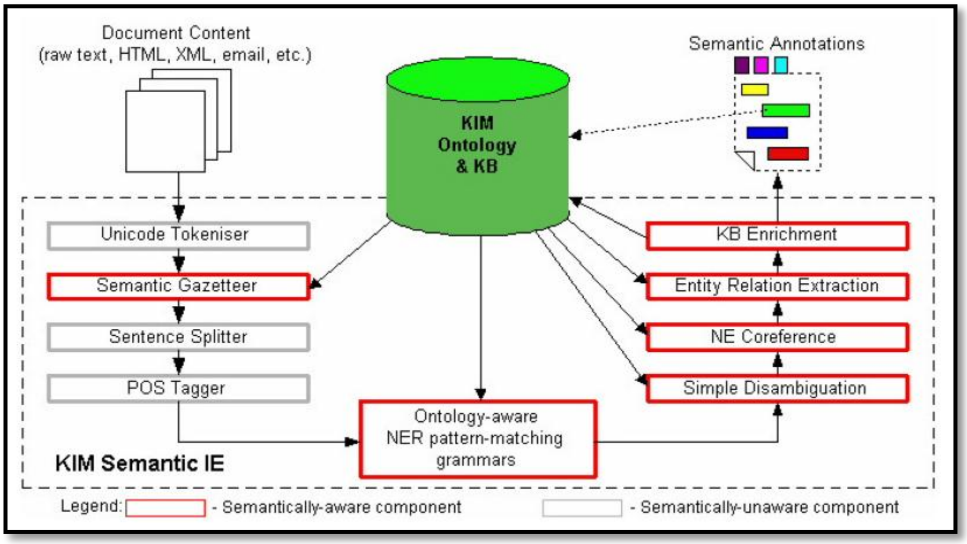


Figure 3.7: KIM Semantic IE flow diagram

This work is relevant for our approach because this research is based on the hypothesis that the named entities in the documents constitute part of its semantics. This idea is useful for our work due to the domain concepts should be related with the type and the intentional element name.

3.5.4 Semantic annotation of RESTful services using external resources

In this work [31] the authors describes an approach to tackle the problem of automating the semantic annotation of RESTful services using a cross-domain ontology, a semantic resource (DBpedia) and additional external resources (suggestions and synonyms services). The system in this work consists of three components: invocation and registration, repository and semantic annotation components. The semantic annotation follows a heuristic approach that combines a number of external services and semantic resources to propose annotations for the parameters (Figure 3.8).

The starting point of the semantic annotation process is a list of syntactic parameters, these parameters are used to query the DBpedia SPARQL Endpoint and retrieve the associated results to each parameter. In order to annotate semantically the parameters that did not match any DBpedia resource, it is add different external services to enrich the results: spelling suggestion and use of synonyms. In this approach is used the Yahoo Boss service, this is invocated for obtaining a list of suggestions to query DBpedia again. The use of synonyms is used to improve the semantic annotations process when our system does not offer results for the previous steps.

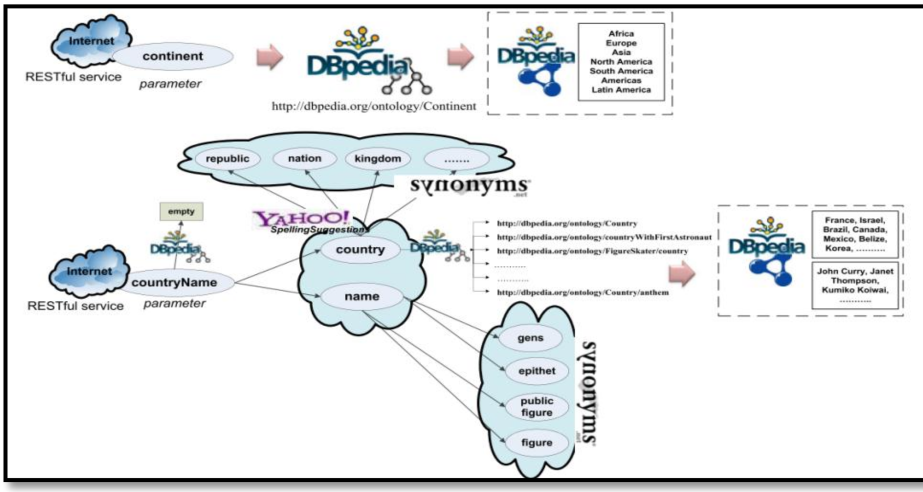


Figure 3.8: Semantic annotation process

This work is relevant for our approach because the idea to annotate RESTful services using cross-domain ontology is useful for carry on our specific and general semantic suggestions using domain concepts.

3.5.5 Ontology enrichment through automatic semantic annotation of On-lines glossaries

In this work [32] the authors provide a methodology for automatic ontology enrichment and for document annotation with the concepts and properties of a domain core ontology. The idea is to present methodology to automatically annotate a glossary *G* with the semantic relations of existing core ontology O. The process is from each gloss *G* of a term *t* in the glossary ***G***, is extracted one or more semantic relation instances R (C_t,C_w), where R is a relation in *O*, C_t and C_w are respectively the domain and range of R. The concept C_t corresponds to its lexical realization *t*, while C_w is the concept associated to a word w in *G*, captured by a regular expression.

The methodology is based on the use of regular expressions, to automatically annotate the glosses for the Architecture thesaurus (AAT), with the properties (conceptual relations) of a formal core ontology whose purpose is to facilitate the integration ad exchange of cultural heritage information between heterogeneous sources, the CIDOC-CRM. The annotated glosses are converted into OWL concept descriptions and used to enrich the CIDOC.

This ontology (CIDOC) includes 84 taxonomically structured concepts and a flat set of 141 semantic relations, called properties. In this approach is mapper manually the top CIDOC entities to ATT concepts (Figure 3.9). WordNet is used to verify that certain words in a gloss-string satisfy the range constraints in the CIDOC. For to annotate sentence of segments with CIDOC properties is proposed the property R: <R>f</R>. The selection of a fragment f to be included in the set F_r is based on different kind of constraints: a part-of-speech constraint, a lexical constraint, semantic constraints on domain and range.

AAT topmost	CIDOC entities
Top concept of AAT	CRM Entity (E1), Persistent Item (E77)
Styles and Periods	Period (E4)
Events	Event (E5)
Activities Facet	Activity (E7)
Processes/Techniques	Beginning of Existence (E63)
Objects Facet	Physical Stuff (E18), Physical Object (E19)
Artifacts	Physical Man-Made Stuff (E24)
Materials Facet	Material (E57)
Agents Facet	Actor (E39)
Time	Time-Span (E52)
Place	Place (E53)

Figure 3.9: Mappings between AAT and CIDOC

This work is relevant for our approach because the methodology followed to annotate documents with concepts and properties of domain core ontology is useful for our approach; we propose semantic suggestions using domain concepts to annotate the elements of organizational models.

3.6 Summary of related work

In this chapter, several related work in research fields closed to the research work developed in this thesis have been presented. A summary of related work is described in Table 3-3. The columns of the table contain the analysis criteria presented in Section 2.2 in which the description of each related work has been based. The rows of the table contain each related work.

Table 3-3: Summary of related work

Related work	Analysis criteria						
	Focus	Objective of the approach	Resources	Type of processing	Annotated resource	Technological Space	Proposed label
Decreus et al. 2009 [13]	Goal-Oriented Requirements Engineering methods	To translate semantically enriched Formal Tropos scripts into BPMO.	SBPM ontological, Formal Tropos, grammar BPMO	-	Formal Tropos script	WSML, WSMO	SBPM_annotation
Ghose et al. 2007 [46]	Agent oriented programming [actor eco-systems]	To propose Semantic annotation of abstract models of actor ecosystems to derive executable process models	i* Models, Formal Tropos, BPMN, CNF, CLN	Automatic	Abstracts models of actor ecosystems	-	-
Di Francesco marino 2011 [11]	Business Process Management	Semantic annotation of Business Process Models	BPMN Ontology, BPMN VQL, BPMN VRL, RDA, CDA, DBDA, WordNet	Semi-automatic	BPMN process elements	OWL, Clustering techniques, Tracer,	bpmn:TextAnnotation
Cuevas 2006 [47]	Ontology merging	To merge two ontologies and to create new ontology	OM Notation, COM algorithm, PLN, WordNet	Automatic	Ontologies	XML	<concept>c</concept>

Guizzardi et al. 2010 [48]	Design of conceptual modeling languages	To design an engineering language to the ARKnowD methodology	Tropos, AORML, Tefkat	Automatic	-	TAOM4E, OWL, MDA,	-
Lin et al. 2006 [12]	Semantic Heterogeneity of Process Models	To proposed a Semantic Annotation Framework	GPO, PSAM, Dublin Core metadata	-	Process models	OWL, Protégé	metaAnno
Gómez-Berbís et al. 2011 [50]	Highly accurate closed domain	To proposed a manual semantic annotation in a highly accurate closed domain	SPARQL RDF, RACER, OWL DL, Renamed Abox	Manual	HACD	AJAX, JENA	-
Kiyavitskaya et al. 2009 [33]	Use of light-weight techniques and tools	To propose a framework for semi-automatic semantic annotation of textual documents according to a domain-specific semantic model	RDF	Semi-automatic	Textual documents	Protégé 3.0, XML, UML, TXL, WordNet, on-line Thesaurus, OWL, HTML	<ad>text</ad>
Erdmann et al. 2001 [51]	Community web portal	To develop of an ergonomic knowledge base-supported annotation tool	RDF, OntoEdit, SMES, Inference engine,	Semi-automatic	Web Documents	XML, HTML-A	<A_onto:“O:C”></A> O=Instancies C=concept
Popov et al. 2004 [40]	Semantic Web	To develop an automatic semantic annotation, indexing and retrieval of documents.	GATE, upper-level ontology, middleware ontology, SESAME, KIMO	Automatic	Web Documents	RDF, Lucene, Internet Explorer Browser	-
Saquicela et al. 2010 [31]	Web services	To proposed a semantic annotation of RESTful services	DBpedia ontology, SPARQL Endpoint, external services (suggestions and synonymous)	Automatic	RESTful services	XML, RDF, Yahoo Boss services,	-
Navigli et al. 2006 [32]	Core Ontology	To provide a methodology for automatically annotate a glossary with semantic relations of a core ontology	CRM CIDOC, ATT	Automatic	On-line glossaries	OWL, WordNet,	<R>f</R>

Chapter 4 Organizational model semantic annotation

4.1 Introduction

The core of this thesis is the enrichment of organizational models with annotations, characterized by an explicitly semantics organized in a structured source of knowledge. The semantic annotations of organizational models, in fact, can be used to provide a precise meaning to elements of the model, thus making them more understandable to people and allowing further analysis. This annotation clarifies the label of the elements and its description by means of domain concepts. In this way, the standardization of elements by means of concepts improves the labeling activity, the process of analyzing and reuse of information.

This chapter describes all the process to carry out the enrichment of organizational models. In order to carry out the process of enriching of organizational models with annotations, our approach consists of two phases. **The first phase is the "*Organizational model semantic annotation*"**. The result of this phase is to represent an annotated model into iStarML format. **The second phase is "*Integrating organizational model ontology and domain ontology*"**. The result of this phase is the integration of an organizational model into domain ontology. An overview in the Figure 4.1 is shown.

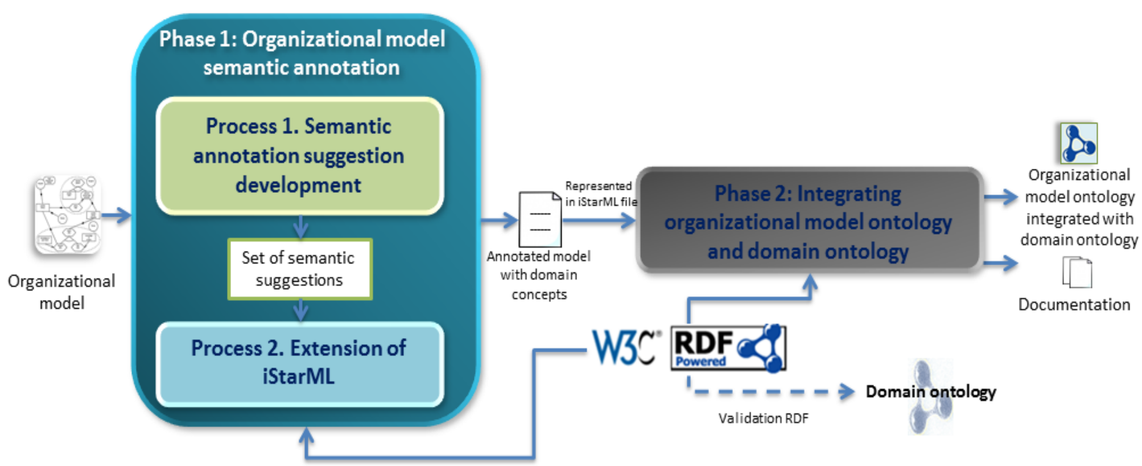


Figure 4.1 Overview of solution methodology

4.2 Phase 1: Organizational model semantic annotation

In this section, the first phase to annotate the organizational model and the development of semantic suggestions are presented. In order to carry out this result, this phase presents two processes.

Process 1 "*Semantic annotation suggestion development*" consists of developing a set of generals and specifics semantic annotation suggestions (Section 4.2.1) and Process 2 "*Extension of iStarML*" consists of representing the annotated model into iStarML format (Section 4.2.2). This iStarML file generated could be the input of some tools in order to represent the organizational model as ontology, or the iStarML could be useful to integrate the model into a domain ontology at instances level.

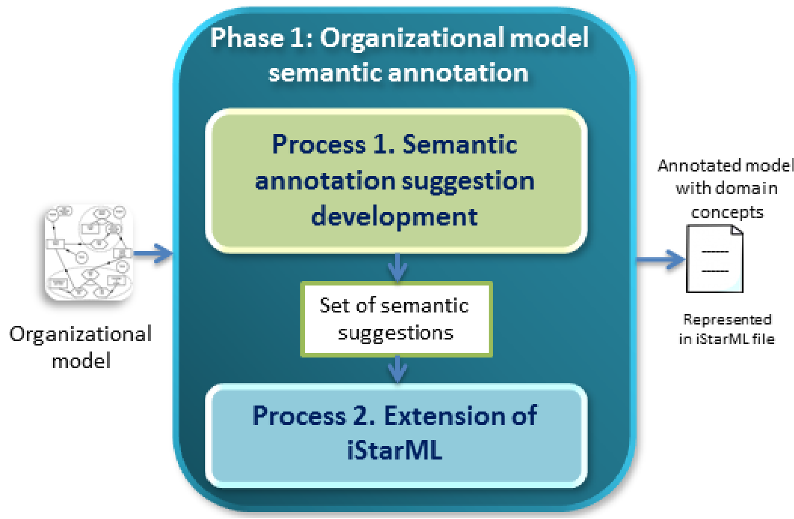


Figure 4.2 Phase 1: Organizational model semantic annotation

4.2.1 Process 1: Semantic annotation suggestions development

This first phase aims at the development of general semantic annotation suggestions that can be applied to any domain ontology and a set of specific semantic suggestions applied to a general ontology and its extension. Figure 4.3 the step in order to develop the suggestions is shown. The inputs in this phase are: i) the organizational model represented in the variants i*, Tropos and service-oriented i* and ii) the domain ontology. The output is the annotated model with domain concepts represented in an iStarML file.

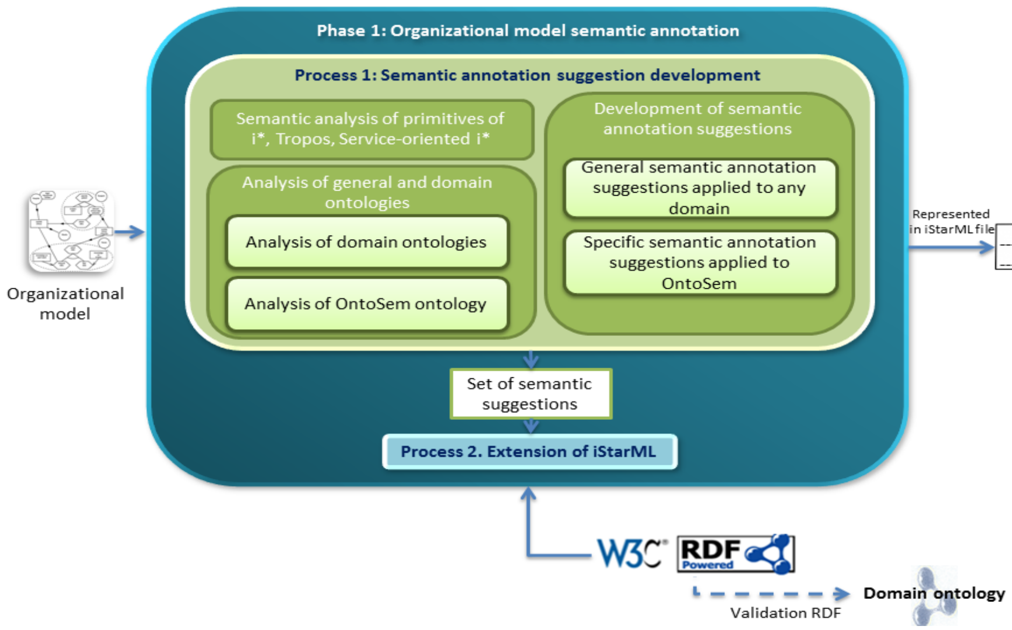


Figure 4.3 Process 1: Semantic annotation suggestions development

4.2.1.1 Step 1: Semantic analysis of primitives of i* variants

The first step consists of analyzing and comparing the primitives of each variant of i*, these are: actor, type actor (*agent, role, and position*), goal, softgoal, task, plan, resource, service and process. The aim of the analysis is to identify the differences and similarities among them. The result is to obtain a single definition for each one of the primitives. This step is explicated formally below. Supposing the sets defined as <V_1, V_2, V_3>, where V_1represents the first variant to analyze, the V_2 represented the second variant and V_3 represented the third variant. Given the following domain elements <p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9,p_{10}, p_{11}>. In Table 4-1 each domain element is defined.

Table 4-1 Describing the domain elements for each variant

Domain element	Representation
p_1	Actor primitive
p_2	Actor type "*agent*"
p_3	Actor type "*role*"
p_4	Actor type "*position*"
p_5	Goal primitive
p_6	Softgoal primitive
p_7	Task primitive
p_8	Plan primitive
p_9	Resource primitive
p_{10}	Service primitive
p_{11}	Process primitive

Now, we define each set with its respective domain elements. For example: $V_1=\{p_{1.1},p_{2.1},p_{3.1},p_{4.1},p_{5.1},p_{6.1},p_{7.1},p_{9.1}\}$, the second set $V_2=\{p_{1.2},p_{2.2},p_{3.2},p_{4.2},p_{5.2},p_{6.2},p_{8.2},p_{9.2}\}$, finally the set $V_3=\{p_{1.3},p_{5.3},p_{6.3},p_{7.3},p_{9.3},p_{10.3},p_{11.3}\}$. In this case, V_1 represents i*, V_2 represents Tropos and V_3 service-oriented i* with its respective primitives. The process consists of analyzing $p_{1.1}$ of set V_1, $p_{1.2}$ of set V_2 and $p_{1.3}$ of set V_3. The aim is to identify the differences and similitudes among them. The result is to obtain a single definition for each primitive that integrated similar features among variants. So, we obtain $\{D_1, D_2, D_3, D_4, D_5 ... D_{11}\}$, where D_n represents the definition integrated of each primitive, such as D_1 represents the definition integrated of actor, D_2represents the type actor "*agent*" and so on.

The comparative analysis of each variant is shown in Table 4-2. The first column presents the primitives (p_n), the next columns presented the definition of each primitive according to the variant. Finally, the last column presents the integrated definition (D_n). The symbol "-" indicates that the primitive is not presents in the variant.

Table 4-2 Comparative analysis among the variants i*, Tropos and Service-oriented i*

Primitive		Definition			Integrated definition
		i*	Tropos	Service-oriented i*	
Actor		An actor is an active entity that carries out actions to achieve goals by exercising its know-how. The term actor to refer generically to any unit to which intentional dependencies can be ascribed.	An actor is an entity that has strategic goals and intentionality.	An actor represents an autonomous and social entity that has strategic goals and intentionality.	The concept of **Actor** is an active entity that has strategic goals and intentionality. An actor can be specialized into agents, roles and positions.
	Agent	An agent is an actor with concrete, physical manifestations, such as a human individual. The term agent instead of person for generality, so that it can be used to refer to human as well as artificial (hardware / software agents).	The concept of agent is used to refer it a human and artificial agents (Hardware/software). An agent having properties such as autonomy, social ability, reactivity, proactivity, rationale.	-	The concept of **agent** has properties such as autonomy, social ability, and physical manifestations such as a human. It can be to refer a hardware and software.
	Role	A role is an abstract characterization of the behavior of a social actor within some specialized context or domain of endeavor.	A role is an abstract characterization of the behavior of an actor within some specialized context.	-	The concept of **role** is an abstract characterization of the behavior of an actor within some specialized context.
	Position	Intermediate abstraction that can be used between a role and an agent. It is a set of roles typically played by one agent. An agent occupies a position. A position is	A position represents a set of roles, typically played by one agent.	-	The concept of **position** represents a set of roles, typically played by one agent.

	said to cover a role.			
Goal	Represents and intentional desire of an actor.	A goal represents the strategic interests of actors.	A goal is a condition or state of affairs in the world that the stakeholders would like to achieve.	The concept of **hard goal** (or simply goal) describes a strategic interest or desire or condition.
Softgoal	Softgoals are similar to (hard) goals except that the criteria for the goal's satisfaction are not clear-cut, it is judged to be sufficiently satisfied from the point of view of the actor.	Softgoals are useful for modeling software qualities, such as security, performance and maintainability.	A softgoal represents a goal that has no clear-cut definition and/or criteria as to whether it is satisfied.	The concept of **softgoal** describes a strategic interest or desires equal a hard goal. . Softgoals are "*subjective to interpretation*" and "*context-specific*".
Task	The actor wants to accomplish some specific task, performed in a particular way. A description of the specifics of the task may be described by decomposing the task into further sub-elements.	-	A task specifies a particular way of doing something.	The concept of **task** describes a clear action or activity well-defined.
Plan	-	A plan represents a way of satisfying a goal.	-	The concept of **plan** describes a clear action or activity well-defined.
Resource	The actor desires the provision of some entity, physical or informational. This type of elements assumes there are no open issues or questions concerning how the entity will be produced or provided.	A resource represents a physical or an informational entity that one actor wants and another can deliver.	A resource represents a physical or an informational entity.	The concept of **resource** describes an entity physical or informational.
Service	-	-	A business service is a functionality that an organizational entity (an enterprise, functional area, department, or organizational actor) offers to other entities in order to fulfill its goals	The concept of **service** is a self-contained, stateless business functionality that is offered to potential customers by means a well-defined interface.
Process	-	-	A process business represents a set of structured activities for producing a specific business service for a particular customer	The concept of **process** represents a set of structured activities for producing a specific business service for a particular customer.

4.2.1.2 Step 2: Analysis of general and domain ontologies

This step consists of analyzing the hierarchy of concepts of general and domain ontologies. The analysis consists of exploring the hierarchy and relationships between concepts. The result of this analysis is to establish relationships between the definition of each primitive (Step 1) towards one or more concepts. Supposing the concepts $C_1$and C_2 are compared with the definition D_1. If C_1 and C_2helps to describe or defined a D_1, then all the instances of the primitive D_1 should be mapped with C_1 and C_2. In a general way, if C_n concept describes a D_n definition, so all the instances $i_1, i_2, i_3, \ldots i_n$ of D_n should be mapped to C_n. In this section, the analysis carried out of domain and general ontologies is presented.

Analysis of Domain Ontologies

This step consists of analyzing the hierarchy of domain ontologies. The result of this step is related the primitives with one or more domain concepts. An overview of domain ontologies analyzed to carry out the semantic annotation is presented below.

In the Figure 4.4 the domain ontology of travels [52] is shown. This ontology address the topic related a travel domain, examples of classes are: CarDomain, FlightDomain, HotelDomain, WeatherDomain and others, etc. This picture represents a general view of this ontology. It is observed different domain concepts, for example: "*Hotel*", "*Hotel Agency*" and "*Restaurant*". Now, we establish relationships between primitives and domain concepts. Supposing, a model could present a primitive of type "*actor*" with the label "*Italian restaurant*" or "*Mexican restaurant*" or contains similar labels.

Using the domain concept of the ontology analyzed, and then we labeled the actor "*Italian restaurant*" or "*Mexican restaurant*" with the domain concept "*Restaurant*". Other example, if a task element presents the label "*Do the reservation in the hotel*" or "*Reserve the room*" could be labeled with the domain concept "*RoomReservation*". It is important define that domain concept should be congruent with the description of the element of the model.

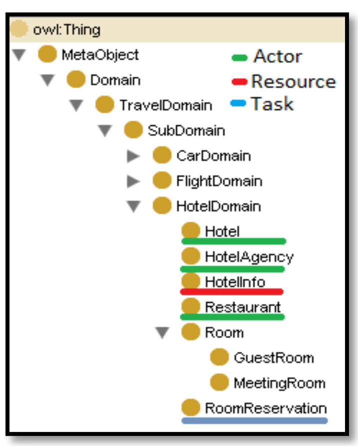


Figure 4.4 Travel ontology

Other ontology analyzed is the "*Urban ontology*" [53]. In Figure 4.5 an overview of this ontology is shown. It is observed domain concepts as "*student*", "*enrollment*", "*trainer*" and others. Supposing a model could present a goal element "*Register in master program*" or "*Register in courses*". Using this ontology the goal should be labeled with the domain concept "*enrollment*". Other case, it is the task element "*Training*" or "*Manage training*" could be labeled with "*professional_training*". The domain concept selected should be congruent with the description of the element of the model.

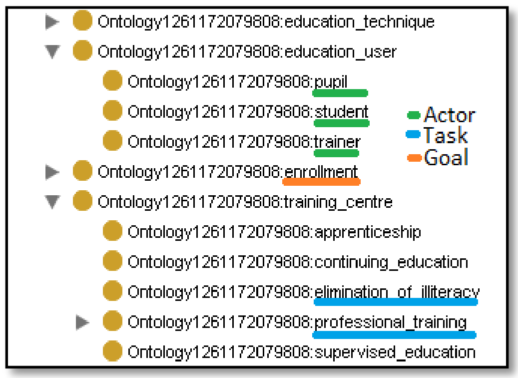


Figure 4.5 Urban ontology

For the development of semantic annotation suggestions different domain ontologies are analyzed. The aim is to examine several domain concepts and to establish relationships between the definition of the primitives and the domain concepts. The advantages of annotating a model element using domain concepts are the following:

- ✓ To clarify the elements of the model and its description by means of domain concepts.
- ✓ To give a formal and precise meaning to the elements of the model:
  - To be able to find and reuse parts of a model when creating new models.
  - To detect cross-item relationships
  - To simplify the management changes
  - To permit the interoperability among i* variants
- ✓ To resolve the ambiguity of natural language descriptions.

Domain ontologies

In Table 4-3 presents a list of the domain ontologies analyzed. The ontologies are classified according to topic, such as: educational, business and other topics. The metrics of these ontologies are described, too.

Table 4-3 Analysis of domain ontologies

Domain Ontology	Metrics	
Educational topic	**Total Class**	**Total properties**
University ontology for benchmark tests [54]	43	25
University ontology [55]	73	32
ScienceOWL ontology [56]	127	63

Portal ontology [56]	3844	108
Research ontology [58]	96	60
Business topic		
Travel ontology [59]	84	100
Organizational ontology [60]	97	76
Other topics		
Anatomy ontology [61]	6222	2
People ontology [62]	60	14
Wine ontology [63]	137	16
Robot ontology[64]	119	24

The OntoSem general ontology

OntoSem means "*Ontological semantics*". It is a theory of meaning in natural language and an approach to natural language processing which uses a constructed world model, or ontology, as the central resource for extracting and representing meaning of natural language texts, reasoning about knowledge derived from texts as well as generating natural language texts based on representations of its meaning [65].

In this way, the most important feature of OntoSem is to be a practical ontology. Research has been applied it in different topics, such as: word sense disambiguation [66] and semantic analysis [67]. It has been already successfully used for a number of non-English languages [68] and others projects. Other ontologies, such as the Dolce ontology [69] are upper level ontology but this ontology compare to OntoSem is not appropriate to practical concrete concepts.

The ontology is organized as a multiple-inheritance hierarchical collection of frames headed by concepts that are named using language-independent labels. It contains three types of concepts: **events, objects** and **properties** (see Figure 4.6). OntoSem containing about 9,000 concepts, that has a number of especially well developed domains that reflect past and ongoing application-specific knowledge acquisition.

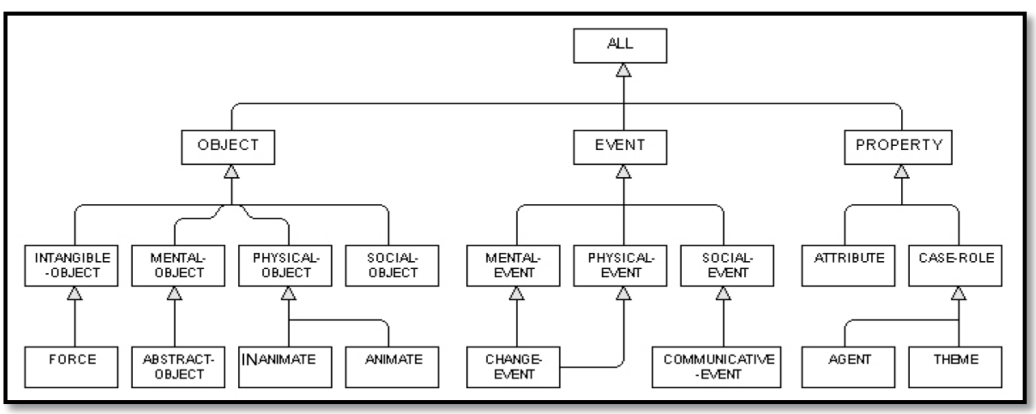


Figure 4.6: Fragment of OntoSem ontology, which tries to capture the most universal object, event and property (relation) concepts referred to by the natural language texts.[70]

Selection restrictions in the ontology are multivalued, with fillers being introduced by a facet. The value facet is rigid and is used less in the ontology than in its sister knowledge base of real-world assertions, the fact repository. The facets default (for strongly preferred constraints) and sem (for basic semantic constraints) are abductively overridable. The relaxable to facet indicates possible but atypical restrictions, and not blocks the given type of filler. The number of concepts in the ontology is far fewer than the number of words or phrases in any language due to the existence of synonyms in language; the possibility of describing lexical items using a combination of ontological and extra-ontological (e.g., temporal) descriptors; the use of a single concept for each scalar attribute that describes all words on that scale (e.g., gorgeous, pretty, ugly); and the decision not to include language-specific concepts in the ontology.

It is a **general ontology**, classifying terms into very high-level categories. The categories have a structural organization quite different from the one adopted in WordNet, in which the hierarchies of different grammatical categories are strictly separated. Other features of the OntoSem ontology that distinguish it from most other ontologies are present below:

- ✓ It is a general ontology, due to this one is a **high-level**, **domain-independent** ontology, providing a framework by which disparate systems may use a **common knowledge base** and from which **more domain-specific ontologies may be derived**.
- ✓ It is available ontology [71] compared with other, such as Dolce ontology [69].
- ✓ Describe an unambiguous model of the world.
- ✓ Provides a metalanguage for describing meaning.
- ✓ The concepts expressed in this ontology are intended to be basic and universal concepts to ensure generality and expressivity for a wide area of domains.

OntoSem concepts

For the development of specific semantic annotation suggestions, first the superconcepts of the OntoSem ontology are analyzed. The aim is to map each primitive of the model to one or more domain concepts of OntoSem. The main hierarchical of this ontology is presented below.

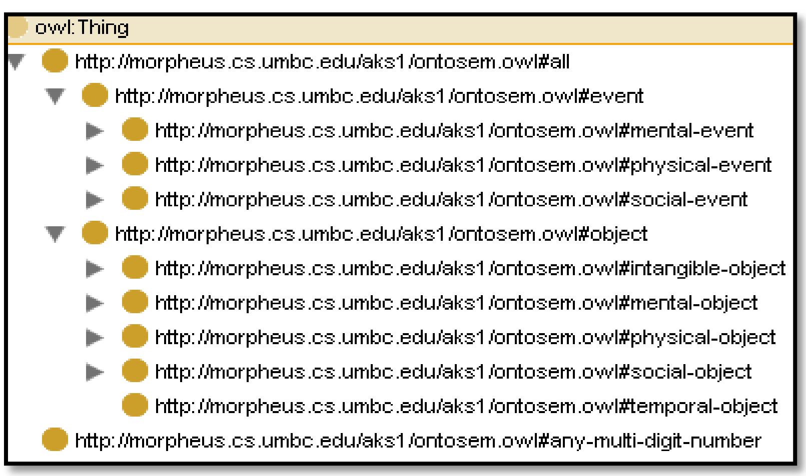


Figure 4.7 The main superconcepts of OntoSem ontology

Table 4-1 presents an overview of the main superconcepts of OntoSem. In the first column indicates the name of the main superconcepts of OntoSem, the second column presents a general definition of the classes that contained, the third column presents examples of subclasses and last column shows the total classes that contained each superconcept.

Table 4-4 Analysis of OntoSem ontology

Name superconcept		General definition	Examples of subclasses	Total subclasses
Event		Any activity, action, happening, or situation.		
	Mental-event	Events which involve mental processes, both active and passive.	active-cognitive-event, change-event, communicative-event, emotional-event.	278
	Physical-event	Events which involve mental processes, both active and passive.	Apply-force, artifact-event, change-location, natural-event, produce.	765
	Social-event	Events involving physical force.	Abstract-social-activity, academic-event, artistic-activity, communicative-event.	582
Object		Ontological concepts that are not actions, or properties; the static things that exist in the physical, mental, and social world.		
	Intangible-object	Objects that cannot be seen or touched but are evident in its influence on the physical world, such as momentum, energy, entrophy, etc.	Energy, entropy, force	14
	Mental-object	Objects that represent other things or ideas; products of mental activity; etc.	Abstract-object, abstract-idea representational-object, field-of-study.	804
	Physical-object	Object which is observable, has position, and has physical dimensions.	Animate, inanimate, physical-system, artifact, and place.	5210
	Social-object	Objects which exist only by the agreement of some people.	Family, organization, society, geopolitical-entity.	326

In general, OntoSem architecture [65] can be characterized by laying out its components:

- ✓ Static knowledge sources: the common sense ontology, fact repository language-specific lexicon, and onomasticon (lexicon of proper names).
- ✓ Formal languages for specifying knowledge representations.
- ✓ Dynamic knowledge sources and text processing.

OntoSem is an integrated, multilingual text processing environment. Multilingualism is supported by means of a language-specific lexicons. It is based on a language-independent ontology, a meta-language which ensures elimination of ambiguity and is able to capture detailed and precise meaning. Due to these features, we propose to apply this ontology in our research work.

4.2.1.3 Step 3: Development of semantic annotation suggestions

This step consists of formally establishing each primitive into one or more domain concepts. The result of this step is a set of general semantic annotation suggestions and a set of specific semantic annotation suggestions. The first suggestions are applied to any domain ontologies. The second are applied to the OntoSem ontology and its extensions of this ontology.

The general suggestions have certain freedom to relate each primitive with domain concepts. For example, the primitive “*goal*” should be mapped into domain concepts that describe a clear and precise condition, interest or desire (Table 4-5). While, the specific semantic annotation suggestions present the relationships of each primitive with one or more domain concepts from OntoSem. For example, the primitive “*goal*” should be mapped to the concepts “*mental-event, social-event and mental-object*” (Table 4-6).

This means that all the instances of a primitive of type “*goal*” should map into one of these concepts, independently of the model domain.

General semantic annotation suggestions

The result of this step is to develop a set of semantic annotation suggestions to guide the process annotation to organizational models. The General Semantic Annotation Suggestions (**GSAS**), applied to any domain ontology are described below.

GSAS 1- The concept of **Actor** is an active entity that has strategic goals and intentionality. We propose that an actor should be mapped into a domain concept that describes an organization, agent, or entity tangible or intangible. For example: if the actor primitive is “*Student Control Department*” then domain concept should be “*academic-department*”.

GSAS1.1- The concept of **agent** has properties such as autonomy, social ability, and physical manifestations such as a human. It also can be to refer a hardware and software. Due to we propose that an agent actor type should be mapped into a domain concept that is refereed to an individual people, or a specific hardware or software.

GSAS1.2- The concept of **role** is an abstract characterization of the behavior of an actor within some specialized context. We propose that a role actor type should be mapped into a domain concept that describes roles an individual person that may have in a society.

GSAS1.3- The concept of **position** represents a set of roles, typically played by one agent. We propose that a position actor type should be mapped into a domain concept that describes a set of roles even a position could be to refer to a human or non-human role.

GSAS 2- The concept of **hard goal** (or simply goal) describes a strategic interest or desire. The goals are concepts well-defined and always possible to identify if these have been fulfilled or not [72]. Guizzardi et.al in [48] mentioned that a set of situations should satisfy a goal and it need be shared by rational agents. As consequence we propose that a goal should be mapped into domain concepts that describe and **clear and precise condition, interest or desire**. For example: if the goal concept is “*Registration in course*” then domain concept should be “*enrollment*”.

GSAS 3- The concept of **softgoal** describes a strategic interest or desire equal a hard goal, but the difference is this element is related with aspects of quality, such as security, performance and maintainability. Sometimes, softgoals are used to represent non-functional requirements.

Softgoals are "*subjective to interpretation*" and "*context-specific*" [48]. Due to this, we propose that a softgoal should be mapped into domain concepts that describe **an interest or desires not clear-cut satisfaction criteria**. For example, if the softgoal concept is "*Better quality papers*" then domain concept should be "*improvement*".

GSAS 4- The concept of **task or plan** describes an action or activity well-defined. Due to this, is proposed that a task or plan should be mapped into domain concepts that describe a **clear action or activity**. For example, if the task concept is "*Capturing student data*" then the domain concept should be "*take-census" or "review" or "information-obtain*".

GSAS 5- The concept of **resources** describes an entity physical or informational. We propose that a resource should be mapped into domain concepts that represent an **object physical or informational entity**. For example, if the resource concept is "*Agri statistical data*" then the domain concept should be "*statistical-number" or "information*".

GSAS 6- The concept of **services** is a self-contained, stateless business functionality that is offered to potential customers through a well-defined interface. Due this, we propose that a service should be mapped into domain concept that represents a functionality or specification of services. For example, if the service concept is "*Flight reservation*" then the domain concept should be "*travel-agency-service*".

GSAS 7- The concept of process represents a set of structured activities for producing a specific business service for a particular customer. We propose that a process should be mapped into domain concepts that describe a **clear action or activity**. For example, if the process concept is "*Request control number*" then the domain concept should be "*information-obtain" or "identify*". An overview of the general semantic annotation suggestions (GSAS) defined in this section and applied to any domain is presented below.

Table 4-5: General semantic annotation suggestions

Primitives	Domain Concepts
GSAS 1- Actor	An actor should be mapped into domain concepts that describe an organization, agent, or entity tangible or intangible.
GSAS1.2 - Agent	An agent should be mapped into domain concepts that are referred to an individual people, or a specific hardware or software.
GSAS1.3- Role	A role should be mapped into domain concepts that describe roles an individual person that may have in a society.
GSAS1.4 Position	A position should be mapped into domain concepts that describe a set of roles even a position could be to refer to a human or non-human role.
GSAS 2- Goal	A goal should be mapped into domain concepts that describe and clear and precise condition, interest or desire.
GSAS 3- Softgoal	A softgoal should be mapped into domain concepts that describe an interest or desires not clear-cut satisfaction criteria.

GSAS 4- Task/plan	A task or plan should be mapped into domain concepts that describe a clear action or activity.
GSAS 5- Resource	A resource should be mapped into domain concepts that represent an object physical or informational entity.
GSAS 6- Service	A service should be mapped into domain concept that represents a functionality or specification of services.
GSAS 7- Process	A process should be mapped into domain concepts that describe a clear action or activity.

Specific semantic annotation suggestions

The set of specific semantic annotation suggestions consist of mapping the definition integrated of each primitive (Section 4.2.1.1) with the concepts of OntoSem (Section 4.2.1.2). The specific suggestions are applied to OntoSem and its extensions. This Specific Semantic Annotation Suggestions (**SSAS**) are presented below.

SSAS 1- The concept of **Actor** is an active entity that has strategic goals and intentionality. An actor can be specialized into agents, roles and positions. We propose that an actor should be mapped into the superconcept "***all:object***" in OntoSem. This concept describes ontological concepts that are not actions or properties; present static things that exist in the physical, mental, and social world. Several of the subclasses of this concept are: intangible-object, mental-object, social-object, etc. This superconcept is composed by 6358 subclasses. For example: if the actor concept is "*Vigilance agent*" then domain concept should be "*watchman" or "police-officer*". A fragment of this superconcept is presented below.

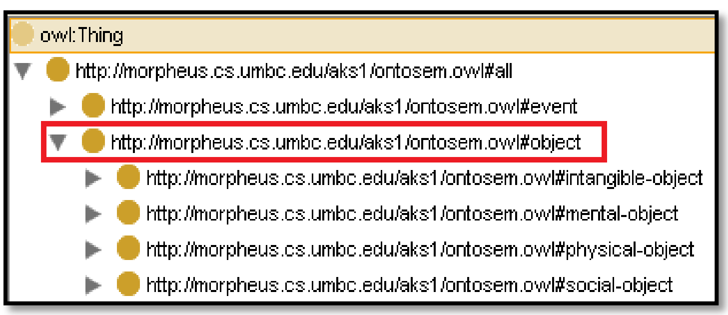


Figure 4.8Hierarchy of superconcept "object".

SSAS1.2- The concept of **agent, role and description** describe an abstract characterization of the behavior of an actor within some specialized context. We propose that these concept should be mapped into the superconcept"***all:object:animate:animal:vertebrate:mammal:primate:human:social-role***". This superconcept describes the roles an individual person may have in a society. Several of the subclasses of this concept are: academic-role, business-role, service-role and others. This superconcept is composed by 373 subclasses. A fragment of this superconcept is presented below.

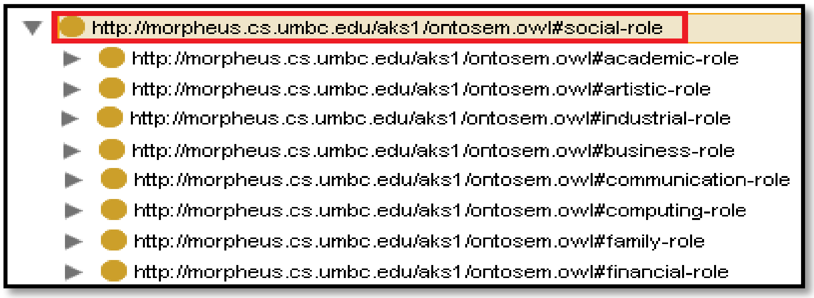


Figure 4.9 Hierarchy of the superconcept "social-role".

SSAS 2-The concept of **hard goal** (or simply goal) describes a strategic interest or desire or condition. The goals are concepts well-defined and always possible to identify if these have been fulfilled or not [72]. As consequence we propose that a goal should be mapped into the superconcepts "***all:event:mental-event***", "***all:event:social-event***" and "***all:object:mental-object***" in OntoSem.

The concepts describe a cognitive action in which analysis and study are involved. Several of the subclasses of these concepts are: analytic-cognitive-event, creative-cognitive-event, demonstrate, etc. There are 1666 subclasses among the concepts. For example: if the goal concept is "*Registration in course*" then domain concept should be "*enrollment*". A fragment of these superconcepts are presented below.

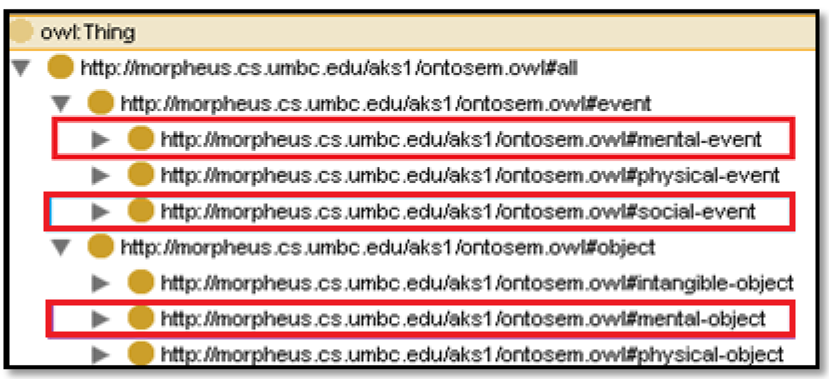


Figure 4.10 Hierarchy of superconcepts "mental-event", "social-event" and "mental-object".

SSAS 3- The concept of **softgoal** describes a strategic interest or desires equal a hard goal, but the difference is this element is related with aspects of quality, such as security, performance and maintainability. Softgoals are "*subjective to interpretation*" and "*context-specific*" [48]. We propose that a softgoal should be mapped into the superconcepts "***all:event:mental-event:active-cognitive-event***" and "***all:object:mental-object***". These superconcepts describe mental objects that are not inherently representational in nature, such as ideas, beliefs and information. Several of the subclasses of this concept are: abstract-idea, classification, conscience, etc. These superconcepts are composed by 871 subclasses. For example, if the softgoal concept is

"*Correctness*" then domain concept should be "*characteristic*". A fragment of these superconcepts are presented below.

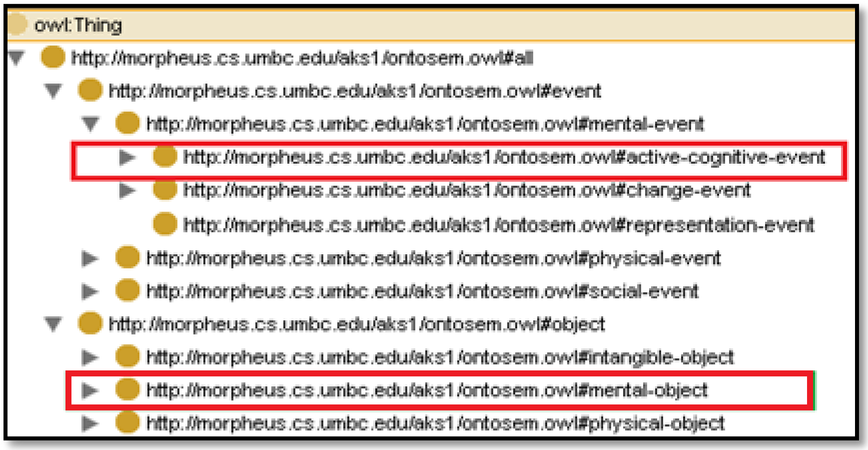


Figure 4.11 Hierarchy of superconcept "active-cognitive-event" and "mental-object".

SSAS4- The concept of **task or plan** describes a clear action or activity well-defined. Due to this, is propose that a task or plan should be mapped into the superconcepts "***all:event:mental-event:active-cognitive-event***", "***all:event:social-event***" and "***all:event:physical-event***". These superconcepts describe actions among peoples and business. Several of the subclasses of this concept are: academic-event, work-activity, abstract-social-activity, etc. There are 1416 subclasses among the superconcepts. For example, if the task concept is "*Register entrance*" then domain concept should be "*register*". A fragment of these concepts are presented below.

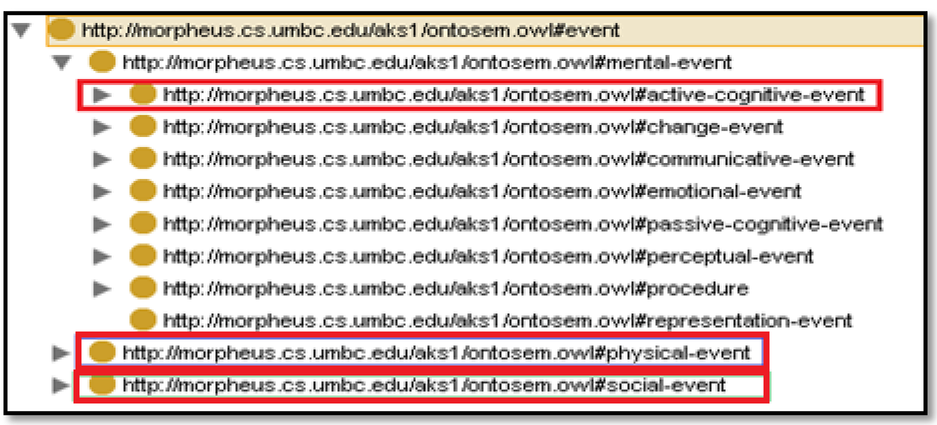


Figure 4.12: Hierarchy of superconcepts "active-cognitive-event" and "social-event".

SSAS5- The concept of **resource** describes an entity physical or informational. We propose that a resource should be mapped into superconcepts "***all:object:mental-object***" and "***all:object:physical-object***". The first superconcept describes an object which is observable, has position, and has physical dimensions. The concept describes objects that represent other things or ideas; products of mental activity. Several of the subclasses of these concepts are: animate, physical-system, abstract-idea, etc. There are 6011 subclasses between both concepts. For example, if the resource concept is "*Information about identify*" then the domain concept should be "*information*". A fragment of these concepts are presented below.

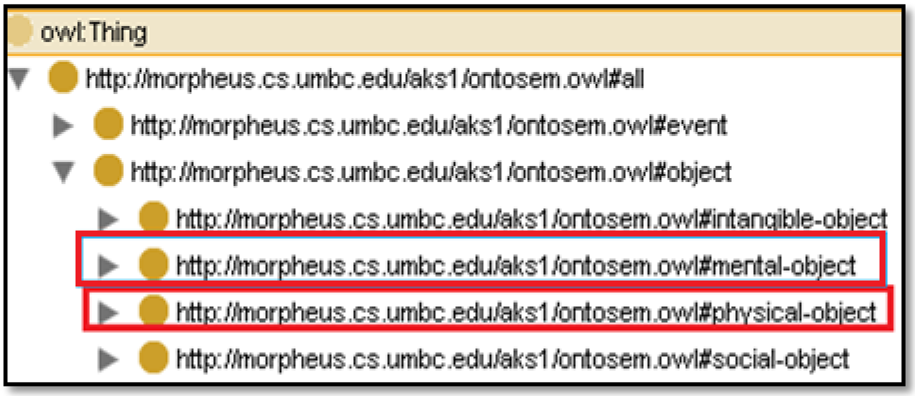


Figure 4.13 Fragments of superconcepts "mental-object" and "physical-object" .

SSAS6- The concept of **service** is a self-contained, stateless business functionality that is offered to potential customers through a well-defined interface. Due this, we propose that a service should be mapped into superconcept "***all:event:social-event***". This superconcept describes events having to do with providing and getting services. Several of the subclasses of this concept are: commonplace-service-event, professional-service-event, computing serve, etc. This superconcept is composed by 583 subclasses. For example, if the service concept is "*Flight reservation*" then the domain concept should be "*travel-agency-service*". A fragment of this concept is presented below.

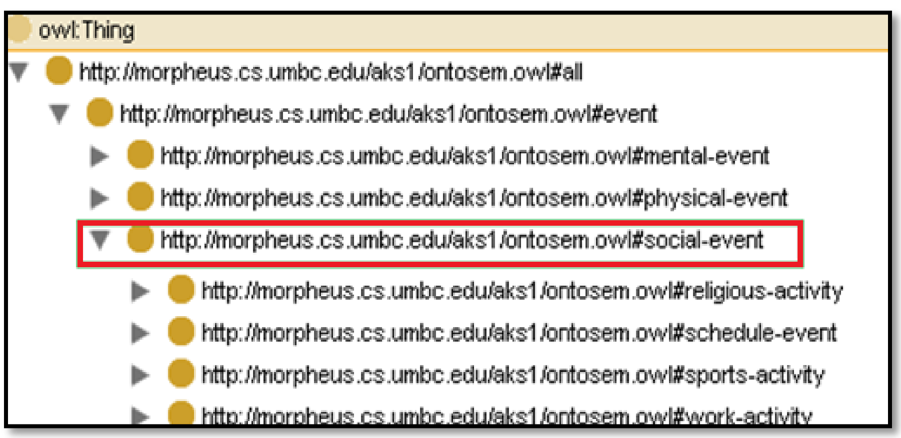


Figure 4.14: Fragments of superconcept "work-activity".

SSAS7-The concept of **process** represents a set of structured activities for producing a specific business service for a particular customer. We propose that a process should be mapped into the superconcepts "***all:event:mental-event:active-cognitive-event***" and "***all:event:social-event***". These superconcepts describe actions among peoples and business. Several of the subclasses of this concept are: academic-event, work-activity, abstract-social-activity, etc. There are 649 subclasses between both concepts. For example, if the process concept is "*Attend class*" then domain concept should be "*attend-academic-institution*". A fragment of these concepts in Table 4-6 is shown.

The merging axioms of the specific semantic annotation suggestions in the Table 4-6 are presented. The suggestions are applied to OntoSem ontology and its extensions.

Table 4-6 Specific semantic annotation suggestions between elements of the models (abb., EM) and OntoSem concepts (abb., OC)

Merging axioms	Domain Concepts
EM: ***Actor*** $\xrightarrow{AB}$ *OC:object*	A type actor element of the model can be annotated only with (can represent only) the superconcept **object**.
EM: ***Agent*** $\xrightarrow{AB}$ *OC:social-role*	A type agent element of the model can be annotated only with (can represent only) the superconcept **social-role**.
EM: ***Role*** $\xrightarrow{AB}$ *OC:social-role*	A type role element of the model can be annotated only with (can represent only) the superconcept **social-role**.
EM: ***Position*** $\xrightarrow{AB}$ *OC:social-role*	A type position element of the model can be annotated only with (can represent only) the superconcept **social-role**.
EM: ***Goal*** $\xrightarrow{AB}$ *OC:mental-event v OC:social-event v OC:mental:object*	A type goal element of the model can be annotated only with (can represent only) the superconcepts **mental-event** or **social-event** or **mental-object**.
EM: ***Softgoal*** $\xrightarrow{AB}$ *OC:abstract-object*	A type softgoal element of the model can be annotated only with (can represent only) the superconcept **abstract-object**.
EM: ***Task*** $\xrightarrow{AB}$ *OC:active-cognitive-event v OC:social-event v physical-event*	A type task element of the model can be annotated only with (can represent only) the superconcepts **active-cognitive-event** or **social-event** or **physical-event.**
EM: ***Plan*** $\xrightarrow{AB}$ *OC:active-cognitive-event v OC:social-event v physical-event*	A type plan element of the model can be can be annotated only with (can represent only) the superconcepts **active-cognitive-event** or **social-event** or **physical-event.**
EM: ***Resource*** $\xrightarrow{AB}$ *OC:physical-object v OC:mental-object*	A type resource of the model can be annotated only with (can represent only) the superconcepts **physical-object** or **mental-object.**
EM: ***Service*** $\xrightarrow{AB}$ *OC:social-event*	A type service of the model can be annotated only with (can represent only) the superconcepts **social-event.**
EM: ***Process*** $\xrightarrow{AB}$ *OC:active-cognitive-event v OC:social-eventv physical-event*	A type process of the model can be annotated only with (can represent only) the superconcepts **active-cognitive-event** or **social-event** or **physical-event.**

4.2.2 Process 2: Extension of iStarML

This section describes the second part of the process of our methodology, and consists in extending the iStarML interchange format. The extension consists of exporting an annotated model to iStarML format adding a new XML attribute (we call this attribute of semantic annotation "sannotation"). This second process consist of three steps: i) Analysis of iStarML format described in Section 4.2.2.1, ii) Extension of iStarML format presented in 4.2.2.2 and iii) Generation of iStarML plug-in for JUCMNav described in 4.2.2.3. The overview of this phase is shown in the

Figure 4.15. The inputs in this phase are: i) the set of semantic annotation suggestions (Section 4.2.1.3), ii) the organizational model represents in the variants i*, Tropos and Service-oriented and iii) the domain ontology. The output is the annotated model represented in an iStarML file.

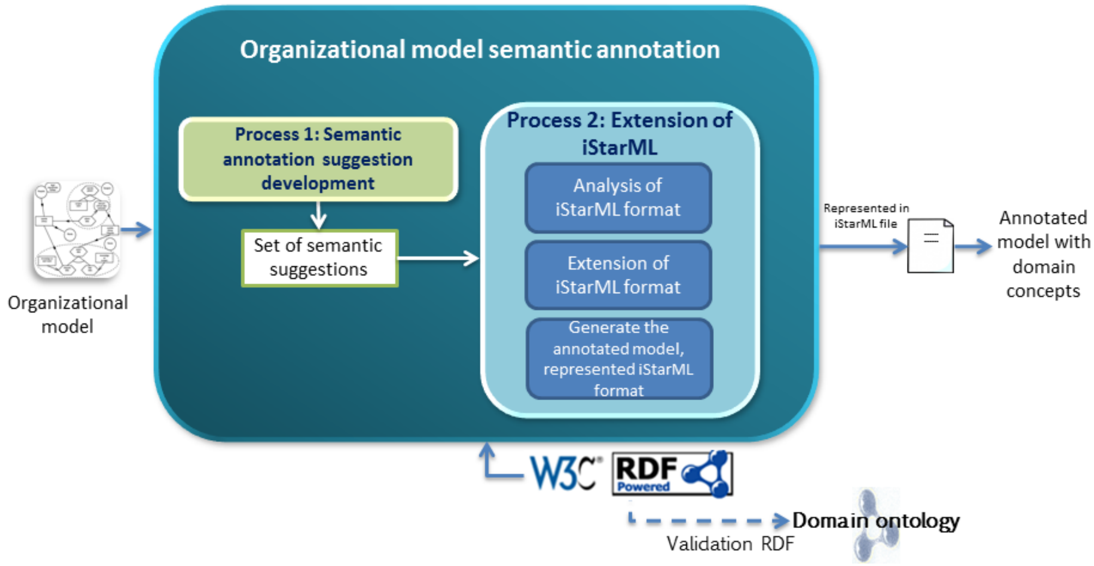


Figure 4.15 Process 2: Extension of iStarML

4.2.2.1 Step 1: Analysis of iStarML format

According to [73] different methodologies have been created based on i* concepts and modeling techniques. In particular the i* framework has been exploited in different areas such as organizational modeling, business process reengineering and requirements engineering. Moreover, some proposals have been made that incorporate i* modeling concepts to deal with software systems requirements representation and design. The goal of iStarML according to [74] is to have a common format where the common conceptual framework of the main i* language variations is made explicit and, in addition, the differences could be expressed using open options using the same specification.

In this way, a common representation allows i) To have an interchange format among i* variants, ii) The representation of differences and similarities among variants, iii) To have a repository common of i* concepts and iv) To represent the interchange format by means of the XML format for Internet communication. The most important features of iStarML format is that the different i* variants can eventually be translated into iStarML [21]. Therefore iStarML allows a textual representation of domain models, requirements, actor relationships and a wide set of the different uses that i* has covered as modeling language, particularly GORE and AORE aspects. In Table 4-7 is shown the core concepts and its corresponding iStarML tags. Also it includes some of the main options in order to illustrate how particular i* constructs can be represented.

Basic Structure of the iStarML format

The tag <istarml> is the main tag in iStarML. It can content only the <diagram> tag. In Table 4-8 the options of this tag are shown. Under this structure it is possible to store on the same file a set of different i* diagrams. The derivation of iStarML tags from the i* core concepts has allowed keeping the language simple and, at the same time, to consider different language variations using the same language constructs. The extensibility of iStarML is provided by allowing additional XML attributes on the static set of iStarML tags [74]. This option seems to be the best one in order to keep a closed core set of fundamental concepts, which would allow the manager of the attribute-based extensionality because the corresponding semantic is mainly associated to the core concept in place of its attributes.

Table 4-7Core concepts of i*-based modeling languages and proposed XML tags for iStarML [75]

*i** core concept	iStarML Tag	Main attributes or subtags
Actor	<actor>	*type* attribute to specify different types of actors (e.g. agent)
Intentional element	<ielement>	*type* attribute to specify different kind of intentional elements (e.g. goal)
Dependency	<dependency>	Can contains two subtags: <dependee> and <depender>
Boundary	<boundary>	*type* attribute for representing future variations on boundary conceptualizations
Intentional element link	<ielementLink>	*type* attribute to specify types of intentional relationships (e.g. contribution) *value* attribute to specify values related to the relationship (e.g. +,++,-,++)
Actor association link	<actorLink>	*type* attribute to specify different types of actors' associations (e.g. is_part_of)

Table 4-8 iStarML syntax [74]

istarmlFile ::=	<istarml version="1.0"> *diagramTag* {*diagramTag*} </istarml>
diagramTag ::=	<diagram *basicAtts* [author=*string*] {*extraAtt*} > [*graphic-diagram*] { [*actorTag*] | [*ielementExTag*]} </diagram>
extraAtt ::=	*attributeName* = *attributeValue*
basicAtts ::=	[id="*string*"] name="*string*" | id="*string*" [name=" *string*"]

4.2.2.2 Step 2: Extension of iStarML format

The extensibility of iStarML interchange format is the main features of this language. Our goal is to extend the iStarML format adding a XML that stores the domain concepts for each element of the model; this label is called "*sannotation*". The syntax of this tag is "*sannotation=concept*$_1$*concept*$_2$ *concept*$_3$*….concept*$_n$".

An element of the model could be an annotation with one or more domain concepts; the goal is to clarify the elements with domain concepts achieving the standardization of concepts by means of similar descriptions. In Figure 4.16 the extension of iStarML format is shown. The tag "*sannotation*" contains the concepts "i*dentify authenticate negotiate-transaction*" from a domain ontology.

```
<ielement id="100" name="Present card for transaction" type="goal"
sannotation="identify authenticate negotiate-transaction">
```

Figure 4.16 Extension of iStarML interchange format

4.2.2.3 Step 3: Generate the annotated model, represented iStarML format

In this section we propose to represent the model in the iStarML format. This step consists of automating the process to generate the model annotated represented in iStarML format. In order to carry out the automation, we propose to extend an existing plug-in for the JUCMNav[1] tool. In the next section an overview of this tool is presented.

JUCMNav tool

JUCMNav [76] is a graphical editor and an analysis and transformation tool for the User Requirements Notation (URN). URN is composed of two complementary notations: the Use Case Map (UCM) scenario notation and the Goal-oriented Requirement Language (GRL).

GRL is based on the i* and NFR frameworks. JUCMNav is an Eclipse plug-in (Figure 4.17) that provides editors for both notations, links between both views, analysis capabilities (including GRL model evaluations), and the Import/Export extension brings the user the possibility to overcome the difficulties and exploit the benefits of i* model interchanging by using the iStarML model interchange format.

The last version 4.0 supports: UCM and GRL editing, user-defined traceability links between GRL elements and UCM elements, UCM analysis (traversal mechanism) based on UCM scenario definitions (initial context), six GRL analysis algorithms: quantitative, qualitative, two hybrid ones, quantitative with KPI functions/aggregation, and constraint-oriented. Integrated UCM/GRL analysis (GRL evaluations affect scenario traversal, and vice-versa), verification of user-defined semantic constraints (in OCL) on URN models and predefined OCL constraints to support a GRL profile for i*, Computation of user-defined metrics (in OCL) on URN models, Structuring of relating GRL and UCM diagrams in "*concerns*".

Supporting of Key Performance Indicators combined with GRL for business process modeling and monitoring, support for Z.151 standard XML file format (import and export), Report generation in PDF, RTF and HTML, export of GRL/UCM models in various bitmap formats, export of strategy results to .csv files, Import/Export of GRL catalogues, integration with Telelogic DOORS 7 and above (for full requirements management).

[1] JUCMNav is a graphical editor and an analysis and transformation tool for the User Requirements Notation.

This tool has been used in the follows projects: "*Healthcare business process, secure electronic access, teaching assistant allocation system, wireless Intelligent Network features*", and others.

The main view of jUCMNav is divided in the follows sections (Figure 4.17).

In the left side the elements browser are presented, in the right side the elements palette are presented, in the bottom side the element properties is shown and the center side the edition of models is presented.

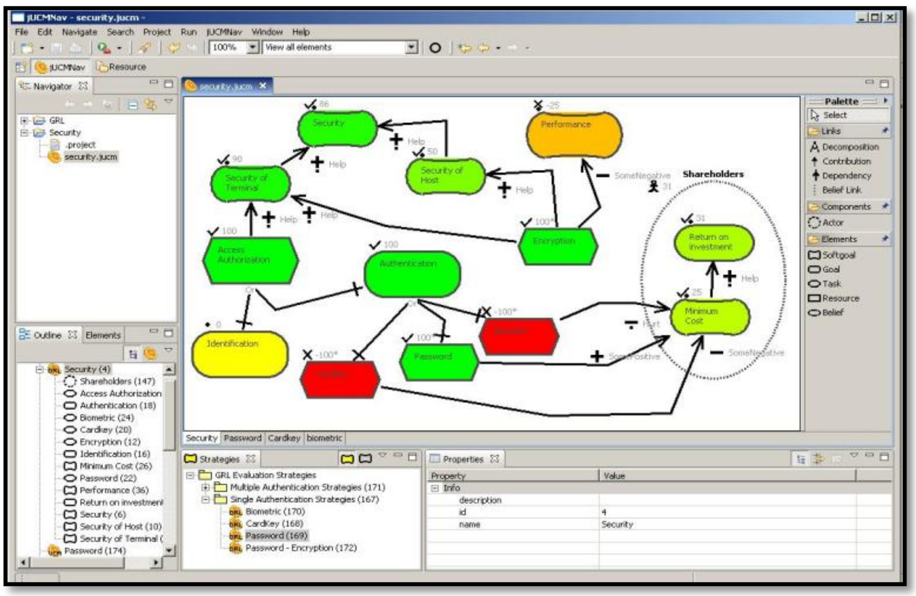


Figure 4.17 jUCMNav tool

Extension of the iStarML plug-in to JUCMNav

We extend an existing the plug-in to export models to an iStarML file. The aim is export an annotated model to iStarML format. It is important to mention that the semantic annotation is added in each element of the model using the *"@"*symbol. In this section, fragments of code will be presented to show the extension of this plug-in.

The files extended are: `ccistarmlContent.java` and `ExportIStarml.java`. The first file encapsulates an abstract functionality for handling iStarML files. It allows XML parsing and iStarML parsing separately. This file provides the basic functionality to create an iStarML file.

In the Figure 4.18 the tags that composed an iStarML file, such as "*id, name, type, sannotation, iref, aref, value, content, xpos, ypos, width, and height*" are shown. In the tag "*sannotation*" all the domain concepts of each element will be stored.

```
public String buildXMLatt() {
    LinkedList t = new LinkedList(Arrays.asList(new String[]{"id","name","type","sannotation",
                  "iref","aref","value","content","xpos","ypos","width","height"}));
    Set key = new HashSet(this.attribute.keySet());
    String att="";
    String val="";
    String ret="";
    while (!t.isEmpty()){
        att = (String)t.poll();
        if (key.contains(att)){
            val=(String)this.attribute.get(att);
                ret+=" "+att+"=\""+val+"\"";
                key.remove(att);
        }
    }
    Iterator it = key.iterator();
    while (it.hasNext()){
        att=(String)it.next();
        val=(String)this.attribute.get(att);
        ret+=" "+att+"=\""+val+"\"";
    }
    return ret;
}
```

Figure 4.18 Extension of ccistarmlContent.java

In order to assign values to each iStarML tag, in Figure 4.19 a fragment of assigning of values for each label is shown. Let see that the tag "*name*" will contain the value of variable "*iname*" this will store the name of the element, the tag "*type*" (that indicates whether the element is goal, softgoal, task, etc.) and the tag "*sannotation*" contain the value of the variable "*sannotation*", this will store the semantic annotation for each element of the model.

```
x.set_attribute("name",iname);
x.set_attribute("type",itype);
x.set_attribute("sannotation", sannotation);
```

Figure 4.19 Assignment of values for each label

Before exporting the model, it is necessary to obtain the value of description property of each element. In order to obtain this value should use the function "*.getDescription*" (Figure 4.20). Then is parsed the property "*.getDescription*" to identify the semantic annotation. The symbol "@" allow us to identify the suggestions for each element of the model. Each word after "@" symbol will be considering the annotation of this element.

The `ccistarmlContent` file analyzes this **description property**, identifies and parser the "@" symbol. Finally when the organizational model is export automatically is generated the new label "*sannotation*" with the value obtained. The Export iStarML file exports the model in iStarML format including the new label.

```
IntentionalElementType ieType = ielement.getType();
if (ieType.equals(IntentionalElementType.GOAL_LITERAL)) {
    return b.add_goal(ielement.getName(),ielement.getDescription());
} else if (ieType.equals(IntentionalElementType.SOFTGOAL_LITERAL)) {
    return b.add_softgoal(ielement.getName(),ielement.getDescription());
} else if (ieType.equals(IntentionalElementType.TASK_LITERAL)) {
    return b.add_task(ielement.getName(),ielement.getDescription());
} else if (ieType.equals(IntentionalElementType.RESSOURCE_LITERAL)) {
    return b.add_resource(ielement.getName(),ielement.getDescription());
} else {
    digest.add("ERROR: intentional Element with no type.");
    return null;
}
```

Figure 4.20 Obtaining the value of description property

In our methodology, we propose to use the **description property** of each element of the model (actor, goal, softgoal, task, resources) to add the semantic annotation. All elements present this description property. In order to identify the domain concept applied, we proposed to use the "@" symbol. All the elements of the model could have one or more domain concepts the objective is clarify the semantic hidden and to improve the activity of labeling.

In the Figure 4.21 the element task "*Analyze materials to propose*" and the description property "*The professor should @analyze of materials to propose*" is shown. Our methodology proposes that each word after of*"@"* symbol is the semantic annotation. The Figure 4.22 a fragment of the representation in iStarML format of this element is shown. The label "*sannotation*" contain the value "*analyze*", and this indicates that the extension of the plug-in is correct.

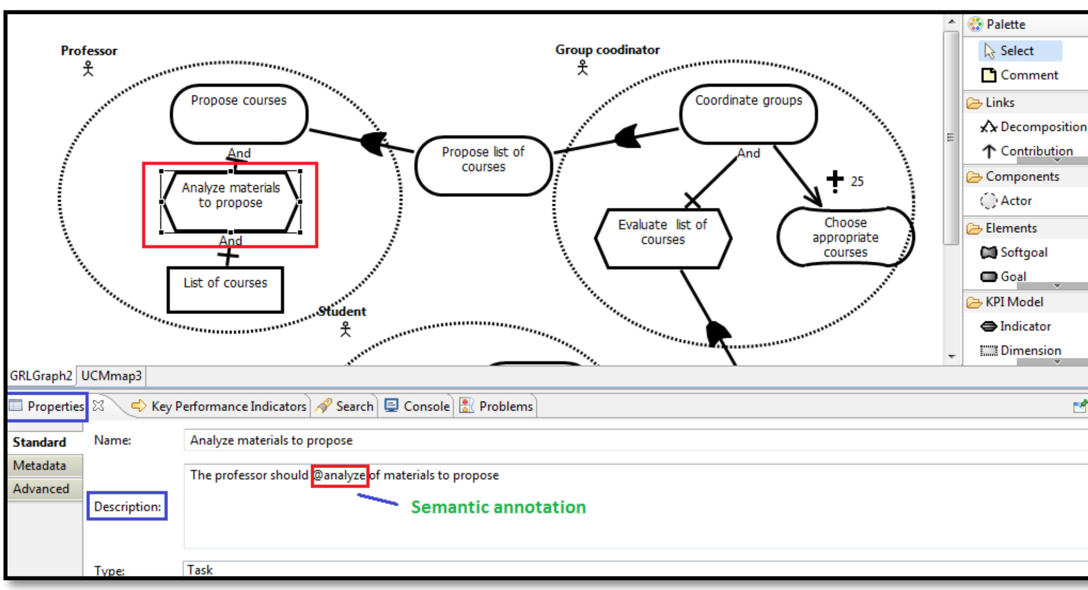


Figure 4.21 Adding the semantic annotation using the property "Description" into the element "Analyze materials to propose".

```xml
<ielement id="AA29" name="Analyze materials to propose" type="task" sannotation="analyze">
    <graphic content="basic" xpos="146" ypos="151" width="0" height="0"/>
    <ielementLink type="decomposition" iref="AA31" aref="AA30" value="and"/>
</ielement>
```

Figure 4.22: Fragment of an exported iStarML file using istarml plug-in extended

4.3 Phase 2: Integration organizational model ontology and domain ontology

The core of this thesis is the presentation of an approach to enrich the organizational model with semantic annotation, and in order to improve the activity of labeling and to avoid inconsistent. In this way, the support of the semantic annotation is achieved the standardization of concepts, sharing a common understanding inside the community and the annotation is useful to discover and to implement services futures. A specific objective of our research indicates "*The development of an approach for building of ontologies integrated with an organizational model ontology*". In order to carry out the phase, we propose the development of TAGOOn+. The next sections the development is described.

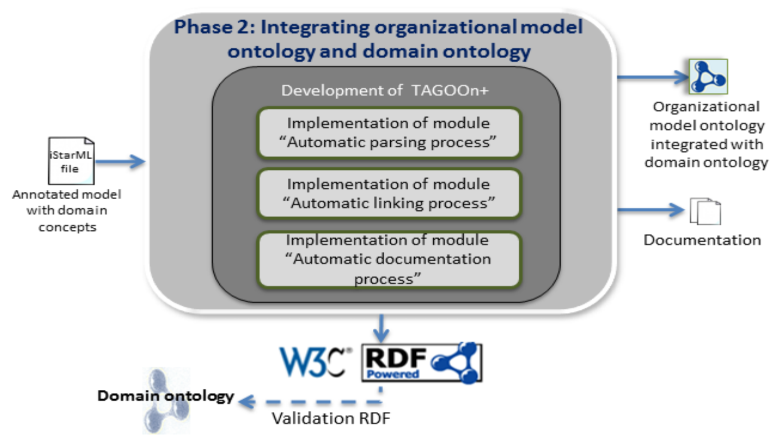


Figure 4.23 Phase 2: Integrating organizational model ontology and domain ontology

4.3.1 Process 1: Development of TAGOOn+

OntoiStar+ [15][77] corresponds to the ontology integrated with the variants: i*, Tropos and Service-oriented i*. This ontology can be used for take advantage of the ontologies services, such as ontology linking service, querying, automated reasoning and others. The objective of this ontology is to represent in terms of ontologies the organizational models generated with i* variants. Our goal is to integrate the organizational model into domain ontology. First the model is represented into an organizational ontology.

In order to carry out this process is proposed used the OntoiStar+ ontology to support this transformation. In the Section 4.2.2.3 the iStarML file that represented the model annotated semantically adding the XML attribute "*sannotation*" was presented. The extension of OntoiStar+

consists of adding the data property "*Node_sannotation*". The Node_sannotation is included into OntoiStar+ as attributes of Node class. Where its domain corresponds to the owner class and its range corresponds to the primitive data type. In Table 4-9 on the left side the data property added in OntoiStar+ is presented.

Table 4-9 Adding dataproperty in OntoiStar+

DataProperty	Domain	Range
Node_sannotation	Node	String

4.3.2 Architecture of TAGOOn+

In order to carry out the integration of the organizational model into a domain ontology we propose the extension of the tool called TAGOOn. This tool can transform i* based models into ontologies, with the variants: i*, Tropos and Service-oriented i*. The extension consists of supporting the automatic transformation and integrating from an i* base model represented in the variants: i*, Tropos and Service-oriented i*, into a domain ontology.

The extended tool is called **TAGOOn+** (**T**ool for the **A**utomatic **G**eneration of **O**rganizational **M**odel **O**ntologies and **I**ntegration). The input of TAGOOn+ is the model represented in iStarML format (Section 4.3.5) and the domain ontology represented in OWL. A previous step is the OWL file should be validates using the "*RDF Validation Service*" [78] proposed by W3C, the aim is to validate the domain ontology and to avoid inconsistent. The output of our tool proposed is an OWL file with a knowledge base which contains as Tbox the concept of the ontology OntoiStar+ and the domain ontology; and as Abox the instances of the elements of OntoiStar+ which represent the organizational knowledge linked with the domain concepts. The architecture of this tool is shown in the Figure 4.24. The architectural is integrated for the modules: "*Automatic Parsing Process*", "*Automatic Mapping Process*", "*Automatic Linking Process*" and "*Automatic Document Process*".

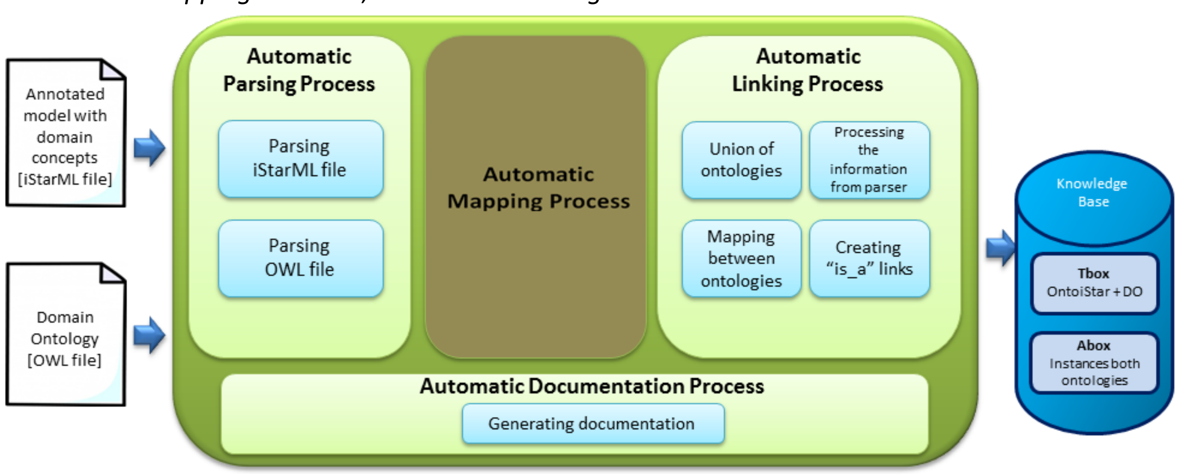


Figure 4.24 Architecture of TAGOOn+

The first module "***Automatic Parsing process***" contains the submodules "*Parsing iStarML file*" and "*Parsing OWL file*". The first sub module reads the iStarML file and then obtained of each element of model its id, name, type, semantic annotation and its relationships with other elements. This information is stored in array for further analysis. The second submodule "*Parsing OWL file*" reads the OWL file and obtained the all domain concepts and its respective label or comment. This information is stored in array for further analysis

The second module "***Automatic Mapping process***" performs the automatic transformation from an i* based model into an ontology derived from the concepts of the OntoiStar+ontology described by Najera in [5]. In this module the XML attribute "*sannotation*" is represented as data property of each element called "*Node_Sannotation*". The output of this module is the model represented as organizational ontology.

The third module "***Automatic Linking Process***" describes the submodules follow: i) "*Union of ontologies*", ii) "*Processing the information from parser*", iii) "*Mapping between ontology*", and iv) "*Creating is a links*". The submodule "*Union of ontologies*" integrated the ontology into the domain ontology. All concepts of both ontologies are integrated in one. The submodule "*Processing the information from parser*" reads the information stored in the two arrays obtained in the module "*Automatic Parsing process".* On the one hand, we obtain the name of each element and its semantic annotation and the other hand we obtain the domain concepts.

Each term is converted from uppercase to lowercase, eliminating white spaces, slash and other information not necessary. This information without additional elements is the input in the next submodule. The submodule "*Mapping between ontologies*" compares each semantic annotation of the elements of the model with domain concepts whether the annotation is equal to concepts then is saved the domain and range of these elements. If both terms are different, then follows searching in all the domain concepts. The searching finish when all the semantic annotation has been evaluate. From of the domain and range stored in the previous step, the submodule "*Creating is a links*" integrates each element of the model with its respective domain concept. The output in this module is an organizational model ontology integrated with a domain ontology represented in OWL.

The four module "***Automatic Document Process***" is related with the generation of the documentation of the model with its domain ontology. The array generated in the first module stored each domain concept with its label or comments. Sometimes, the general ontology provides a metalanguage to describe each concept. In the case of OntoSem each concept is related with its own description. When each element is integrated with one or more domain concepts is searched the description of each domain concept. The output of this module is a text document that represents each element of the model with its semantic annotation and the description of each of the annotated concepts in the ontology. In Figure 4.25 the structure of this output is shown.

```
NAME: "Name of the element" ANNOTATION: "Domain concept" DESCRIPTION = "Brief description of each concept"
```

Figure 4.25 Structure of the documentation generated for TAGOOn+

4.3.3 Description of documentation generated in TAGOOn+

The documentation generated for TAGOOn+ is useful for the technical and analyst people in order to achieve a better understanding of the organizational model. Each element of model is grouped according to type of element. This first group is the actors, then the intentional element and finally the dependencies. The goal of our research is the standardization of elements by means of concepts improves the labeling activity, the analysis process and allows information reuse

In the Figure 4.26 an example of the documentation generated is shown. Let see how the dependum of type "*goal*" called "*Present card for transaction*" was annotated with three domain concepts "*negotiate-transaction, identify and authenticate*". The description of each concept also is visualized. Others dependums are presented, such as new account and account be managed.

```
        ------------------------>> Intentional element:Dependum <<-------------------------------
        ------------------------>> Intentional element:Dependum  Type: Goal <<--------------------
NAME: Present card for transaction ANNOTATION: "negotiate-transaction" DESCRIPTION =  "to work out the terms of a transaction in order to reach an agreement"
NAME: Present card for transaction ANNOTATION: "identify" DESCRIPTION =  "to fix the identity of something or someone"
NAME: Present card for transaction ANNOTATION: "authenticate" DESCRIPTION =  "to verify the identity of someone or something in order to grant access privileges"
NAME: new account be created ANNOTATION: "open-account" DESCRIPTION =  "The event of opening a bank account."
NAME: new account be created ANNOTATION: "bank-account" DESCRIPTION =  "money deposited in a bank and credited to the depositor"
NAME: Account be managed ANNOTATION: "finance" DESCRIPTION =  "The field of study that is concerned with the science of managing money."
NAME: Account be managed ANNOTATION: "money-managing-activity" DESCRIPTION =  "Business-activity that involves the managing of money and other financial-objects.
NAME: Account be managed ANNOTATION: "accounting" DESCRIPTION =  "to make a record of financial transactions in a business"
```

Figure 4.26 Example of the documentation generated for TAGOOn+

4.4 Guideline to annotate the organizational models through semantic annotation

In this section, we describe the guidelines to annotate the organizational model applying to any domain ontology and to a generic ontology. This guideline also considers the process to export the model using the extension of iStarML interchange format and the plug-in to export the model, and finally the integration of the model with the domain ontology is presented too. In the section 4.4.1 step by step we describe how to annotate the models using our guideline.

4.4.1 Description of the annotation process

The goal in this research work is to enrich the organizational model defining the elements description by means of generic concepts. The guidelines to carry out the annotation process of organizational models described in i*, Tropos and Service-oriented i* are the following:

1. **To obtain the domain ontology**. Some scenarios to obtain the ontologies are presented below.

   a. Existing different repositories of ontologies to obtain domain ontology, such as Swoogle[2], BioPortal[3], Protégé Ontology Library[4] and others.
   b. Sometimes, when an organization is modeled the analysts create ontologies to describe the entities and its relationships.
2. **To validate the domain ontology**. In order to avoid inconsistencies and missing of information is proponed to validate the domain ontology with an online tool proposed by the World Wide Web Consortium (W3C) called "*Validation Service*[5]". This tool validated the syntaxes and visualizes RDF documents. In order to validate is needed to enter a URI or paste an RDF/XML document into of a text field. A 3-tuple (triple) representation of the corresponding data model as well as an optional graphical visualization of the data model will be displayed if the document is correct.
3. **To obtain an organizational model to be annotated**. The organizational model can be represented in the variants i*, Tropos and service-oriented i*.
4. **To define the type of semantic suggestions to apply**. The type of domain ontology defines the semantic suggestions to use.
   a. If the ontology is about a specific problem or generic domain, so the analyst should use the general semantic annotation suggestions (Section 4.2.1.3).
   b. If the ontology is a general ontology such as OntoSem or extend this ontology, so the analyst should use the specific semantic annotation suggestions (Section 4.2.1.3)
5. **To select an element of model to annotate.** Each variant presented different primitives described in the section 2.1. Due to the definition of each element is the type of suggestions to apply.
6. **To annotate each element of the model**. In order to carry out the semantic annotation, it is necessary to attend the semantic annotation (general o specific) suggestion, then it going in-depth of the ontology and to find out the more appropriate domain concept for the model element. In order to annotate each element with domain concept from OntoSem, the process should be for example: if the suggestions indicate that the element "*task*" should annotate with superconcept "*Social-event*" then going in-depth of the superconcept and to find out the more appropriate domain concept for the task element. This concept should be congruent with the description of the element. The idea is to annotate all the elements of the model with one or more domain concepts, such annotation provide enrichment and formal meaning to the element description and allow the implementation of services
7. **To export the organizational model**. When all the elements of the model have been annotated should export it into iStarML interchange format.

[2] http://swoogle.umbc.edu/
[3] http://bioportal.bioontology.org/
[4] http://protegewiki.stanford.edu/wiki/Protege_Ontology_Library
[5] http://www.w3.org/RDF/Validator/

  a. The model could be modeled using JUCMNav tool and to add the semantic suggestions in the description property of each element of the model. To export the model adds the extended plug-in proposed in this research work.
  b. The model could be modeled using other graphic tool and to export the model to iStarML format adding manually the label "*sannotation*" with its respective domain concepts.

8. **To integrate the model with the domain ontology.** In order to integrate the model with a domain ontology, the model should be represented in iStarML format and the domain ontology in OWL file. Both files are the input of TAGOOn+. During the execution of TAGOOn+ the analyst should indicate the URI to save the organizational model ontology integrated with a domain ontology represented in OWL file, and the documentation of this integration represented in a text document.
9. **To analyze the joined ontology.** In order to analyze the integrated ontology is propose to open the OWL file generated by TAGOOn+ using Protégé 4.1. In this way, the new ontology can be analyzed graphically and applied reasoning about this.
10. **To analyze the documentation generated.** In order to analyze the documentation generated could open the text file using any text editor.

The novel of our research emerges to annotate all the elements of the model with one or more domain concepts. The objective is to provide a precise and clear meaning to the elements of model, achieving the standardization of concepts and common understating within a community.

4.5 Summary

This chapter described an approach proposed to annotate the elements of the model with semantic annotation from generic concepts. In Section 4.2 the development of the semantic annotation suggestions were presented. First a semantic analysis of the primitives of i* variants were done (Section 4.2.1.1), the aim was examined how each variant define its primitives semantically and to compare among variants to obtain a single definition of each primitive. Then an analysis of general and domain ontologies were presented (Section 4.2.1.2).

Several domain ontologies were presented analyzing its concepts and the taxonomy of a generic ontology was described. This analysis of a generic ontology consisted of developing the specific semantic suggestions where each primitive was mapped towards one or more domain concepts. In Section 4.2.1.3 the general semantic annotation suggestions apply to any domain ontologies were presented. In the same section the specific semantic annotation suggestions apply to OntoSem ontology and its extension were presented. In Section 4.2.2 the extension of iStarML interchange format to represent the annotated model was presented.

An analysis of iStarML format has been carried out to extend this format. The extension of iStarML interchange format was presented; the new the XML attribute called "*sannotation*" stored the semantic annotation for each element of the model in section 4.2.2.2 was described. Finally, the analysis and extension of an existing plug-in for the JUCMNav tool was presented. We proposed the use “@”symbol, this annotation allow us to categorize the element of the model, by unifying labels. The iStarML format was selected to export the model because the extensibility to add XML attributes on the static set of iStarML tags.

Section 4.3 the process of integrating an organizational model into a domain ontology was presented. In order to carry out the integration, on the one hand the extension of the OntoiStar+ ontology adding the data property “*Node_Sannotation*” for each concept of OntoiStar+ was presented. On the other hand, we proposed a tool extended called “*TAGOOn+*” allowed us the integration of the model organizational represented in iStarML format and the domain ontology, the architecture of our tool proposed in Section 4.3.2 was presented. The output of the tool is an OWL file with a knowledge base which contains as Tbox the ontology OntoiStar+ and the domain ontology; and as Abox the instances of the elements of OntoiStar+ which represent the organizational knowledge linked with the domain concepts. In Section 4.3.3 we presented the description of the documentation generated in the tool proposed. Finally, Section 4.4.1 described the complete process to apply our guideline.

Chapter 5 Tool for the Automatic Generation of Organizational Ontologies and Integration

5.1 Introduction

In this chapter our tool extended called TAGOOn+ is presented. TAGOOn+ (Tool for the Automatic Generation of Organizational Ontologies and Integration) supports the automatic transformation and integrating from an i* base model represented with the variants: i*, Tropos and Service-oriented i* into a domain ontology or generic ontology. The architecture and the modules to develop this tool are presented in this section.

5.2 Development of TAGOOn+

TAGOOn+ has been developed in order to integrate an organizational model represented in the variants i*, Tropos or Service-oriented i* into a domain ontology or generic ontology. The result is the generalization of the elements of the model by means of concepts improves the labeling activity, the analysis process and allows information reuse. The categorization of elements with the same annotations could be useful to implement futures services. The components to develop TAGOOn+ are presented below.

Eclipse Modeling Tools: TAGOOn+ has been developed using the environment of Eclipse project and the programming language Java. The version used is "*Indigo Service Release 1*"and the version of JAVA is the 1.6.0_24. The JDK is available in http://www.oracle.com/technetwork/java/javase/downloads/index.html and the version Indigo for Eclipse is available in http://www.eclipse.org/downloads/packages/release/indigo/r.

JENA: It is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine. The version of JENA used in TAGOOn+ is the 2.6.4. This version is available in http://jena.apache.org/download/index.html

Protégé: Protégé is a free, open source ontology editor and knowledge-based framework. The Protégé platform supports two main ways of modeling ontologies via the Protégé-Frames and Protégé-OWL editors. Protégé ontologies can be exported into a variety of formats including RDF(S), OWL, and XML Schema. The version used to open the files generated by TAGOOn+ is the 4.1. This version is available in http://protege.stanford.edu/download/download.html

JDOM: It is a unique Java toolkit for working with XML, engineered to enable rapid development of XML applications. Its design embraces the Java language from syntax to semantics. The version used is 1.1.2. This version is available in http://www.jdom.org/downloads/index.html

5.2.1 Modules of TAGOOn+

TAGOOn+ requires two files as input: i) an OWL file describing the general or domain ontology and ii) an iStarML file describing an annotated organizational model. The output of TAGOOn+ are two files: i) an OWL file with a knowledge base which contains as Tbox the ontology OntoiStar+ and the concepts of the domain ontology; and as Abox the instances of the elements of OntoiStar+ which represent the organizational knowledge linked with the domain concepts and ii) a text document describing each element of the model with its respective domain concepts.

The modules of TAGOOn+ are: i) "*OWL file manager*", ii) "*Mapping process*", iii) "*Linking process OntoiStar+ Domain ontology*" and iv) "*Documentation process*". The module "*Linking process OntoiStar+ Domain ontology*" is the most important module of TAGOOn+. In this module each element of the model is related with its corresponding concept in the domain ontology by means of "*is a*" links.

Module "OWL file manager"

The module "*OWL file manager*" loads, reads and analyzes the domain ontology stored in the OWL file. The result of this module is to obtain all the concepts and its descriptions of the domain ontology (Figure 5.1). All the information obtained is stored in an array in order to future analysis.

Module "Mapping process"

The module "*Mapping process*" analyzed the array that is obtained in the first module. Also, the iStarML file is stored in an array to be analyzed. Both analyses consisted of converting from uppercase to lowercase the elements of the arrays. Moreover the spaces-white and slash are deleted. This process is to avoid wrongs during the process of integration.

During the analysis of iStarML the main tag is "*sannotation*" because stored the domain concepts for each element. So, when this analysis finalized, the value of "*sannotation*" is compared with the domain concepts exiting in the ontology. If both terms are equals then is saved the domain (URI of the element of the model) and the range (URI of domain concept). The result of this module is to store the domain and the range of each element related (Figure 5.1).

Module "Linking process OntoiStar + Domain Ontology"

The module "*Linking process OntoiStar + Domain Ontology*" is the central module of TAGOOn+. This module interacted with the user to indicate the URI to store the organizational model ontology integrated with a domain ontology. This module consists of the union of the model represented as ontology and the domain ontology.

From the domain and the range stored in the previous module is establish a relationship "*is a*" between the individuals the model and the concepts of the domain ontology. If an individual not contains semantic annotation will not related with any concept of the ontology. The result of this module is to generate the OWL file in order to represent the integration between ontologies (Figure 5.1). We propose to use Protégé 4.1 to visualize and to reason about this integrated ontology.

Module "Documentation process"

The module "*Documentation process*" consists of storing the individual of the model related with the domain concept and its description. The result is to generate a text document that classified each element of the model according the type of the element. Then each element should relate with it domain concept and the description of its (Figure 5.1).

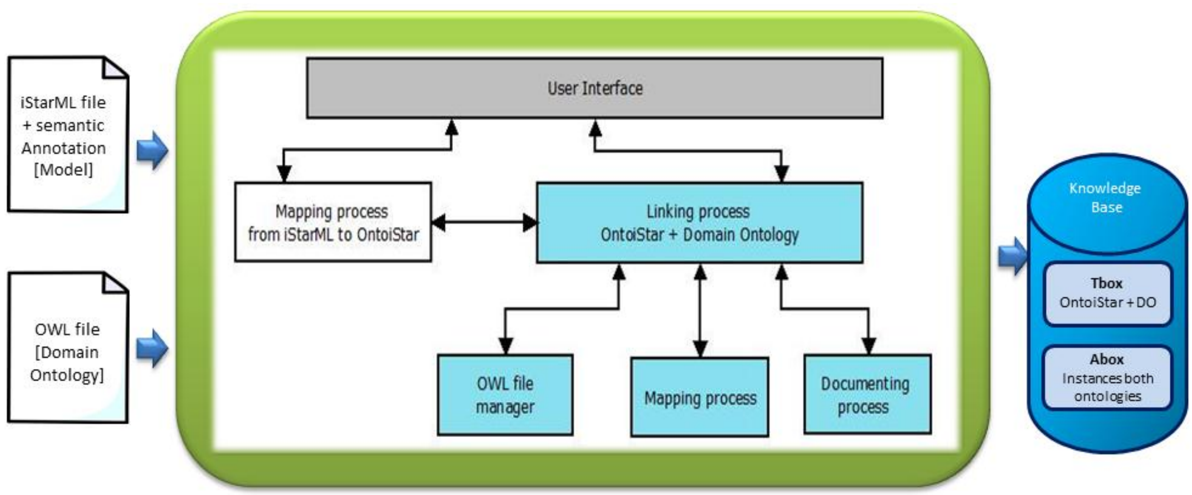


Figure 5.1Modules of TAGOOn+

5.2.2 User interface of TAGOOn+

The interface of TAGOOn+ has a simple Graphic User Interface (GUI) with five menus (Figure 5.2): "*File*", "*Options*", "*Documentation*", "*View*" and "*Help*". The menu "File" contains the options to open an iStarML file and to close the application. The menu "*Options*" allows us to generate an OWL file to order to represent the model into an organizational ontology. The menu "*Documentation*" generates a text document. In order to document the integration between annotated model and domain ontology.

TAGOOn+ presents two views: i) The basic view in Figure 5.2 is shown and ii) the detail view in Figure 5.3 is shown. The difference between views is the detail level that show for the user. For example, in the basic view presents a center panel, in this panel the user observed the systems messages, such as "*1.Load Domain Ontology*". While, in the detail view presents four panels.

The first left-panel the domain ontology loaded is shown. In the second center-panel the iStarML file is shown. In the third right-panel the organizational model integrated with a domain ontology are shown, finally in four bottom panel the systems messages is shown. The menu "*View*" presents the options to select the type of view. The menu is "*Help*" contains the options "*Help contents*". This menu introduce bibliographic as reference of the tool. The options "*About*" describe the copyright of the tool.

Execution TAGOOn+

TAGOOn+ requires two files as input: i) an OWL file describing the general or domain ontology and ii) an iStarML file describing an annotated organizational model.

In order to load the domain ontology the options are: ii) the button "*1. Load Domain Ontology*" or ii) the option "*Load Domain Ontology*" from menu "*Options*". If the view selected is "*detail view*" (Figure 5.3), so in the left panel is shown the domain ontology. All the time, the bottom panel shows system messages, such as: "*Loaded domain ontology: D:\Case study\Specific suggestions\ontosem.owl*".

Then, in order to load the iStarML file the options are: i) the button "*2. Open an iStarML file*" or ii) the option "*Open iStarML file*" from menu "*File*". The message that should see the user is: "*Opened file: D:\Case study\Specific suggestions\iStarML files\Test1.xml*".

To generate the integration between ontologies, the user should create the ontology that represented the annotated model. In order to represent the annotated model into ontology the options are: i) the button "*3.Generate OWL file*" or ii) the option "*Generate OWL file*" from menu "*Options*". Then, a dialog box is shown; the user should indicate the place to save the OWL file. The systems messages in this option are: "*To save the organizational model ontology click in the bottom: Save OWL file as*" and "*To create the integration between ontologies click in the bottom: Create Links &Save*".

Then, in order to integrate the organizational model with a domain ontology the option is: i) Button "*Create Links &Save*". Again a dialog box is shown; the user should indicate the place to save the OWL file. When the integration has finalized, the system message is "*Integrating the organizational ontology with a domain ontology*". When the integration has been carried out, then it should generate the documentation. The option to save the documentation is: the option "*Generate Documentation*" from menu "*Documentation*". The last systems messages should be: "*Successfully saved file: /Case study/Specific suggestions/Test1.txt*".

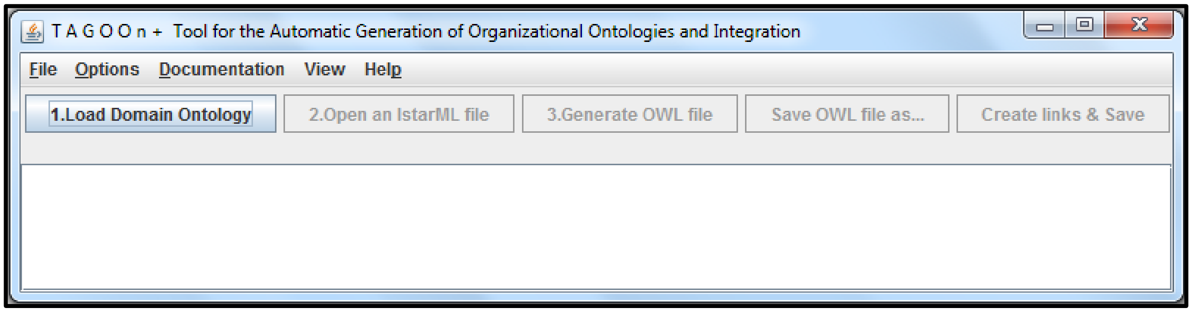


Figure 5.2 Basic view of TAGOOn+

5.2.3 Interactions among modules

In this section the interactions among the modules and the user interface of TAGOOn+ is presented.

The modules of TAGOOn+ were presented in Figure 5.1. Then button "*1. Load Domain Ontology*" active the module "*OWL file manager*"; it allows us to load, to read and to analyzes the domain ontology. The button "*2. Open an iStarML file*" active the module "*Mapping process from iStarML to OntoiStar*"; it allows us to load, to read and to analyzes the iStarML file. The button "*3.Generate OWL file*" used the module "*Mapping process from iStarML to OntoiStar*". Each element of the model is represented as an individual in the ontology. The button "*Create Links &Save*" active the module "*Linking process OntoiStar + Domain Ontology*"; it allows us to create the links "*is a*" between the organizational ontology and the domain ontology. The button "*Generate Documentation*" active the module "*Documentation process*"; it allow us to create an overview of each element of the model with its concepts and the description of them.

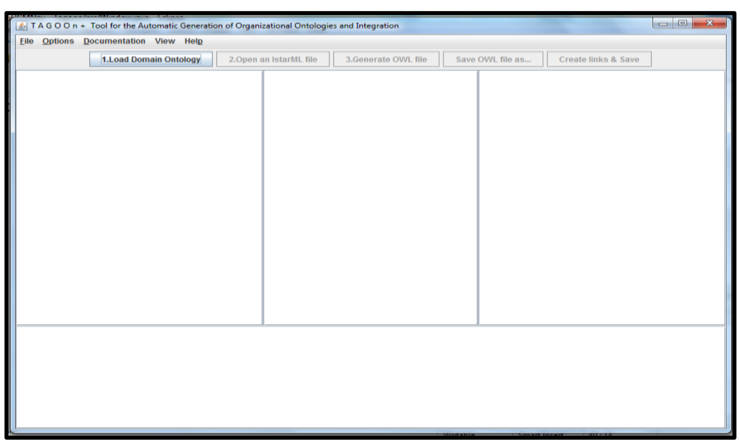


Figure 5.3 Detail View of TAGOOn+

5.3 Summary

In this section, the development of a tool to integrate the organizational model ontology with a domain ontology called TAGOOn+ has been presented. The components and the versions in order to develop TAGOOn+ such as Eclipse, Protégé, JENA and JDOM were described.

TAGOOn+ consists of four modules: i) "*OWL file manager*" reads and to analyze the domain ontology, the result is to generate an array, ii) "*Mapping process*" cleans and to compare each element of the model with domain concepts. The result is to store the URI of each element of the model related with domain concepts, iii) "*Linking process OntoiStar + Domain Ontology*" liked each element of the model with one or more domain concepts. The result is an OWL that integrated the annotated model with a domain ontology and iv) "*Documentation process*" generates a text document representing the integration between ontologies. The user interface of TAGOOn+ was presented. The basic and the detail view were described. A guideline to execute our tool in order to integrate an organizational model with a domain ontology was described. Finally, the interaction among modules and the user interface to clear the behavior of our tool were presented.

Chapter 6 Case Study

6.1 Introduction

The main objective of this thesis is to propose an approach to enrich organizational model with annotations characterized by a semantics explicitly organized in a structured source of knowledge, in order to improve the labeling activity, the process analysis and the reuse of information. In this chapter, the application of the proposed solution has been validated with real case studies. First, a briefly introduction of case studies are presented. Then the organizational models are visualized. In order to validate our approach, the models are annotated using domain ontologies and general semantic annotation suggestions. Later, the models are annotated using OntoSem ontology and specific semantic annotation suggestions.

6.2 Description of the cases studies

Three real case studies have been taken to enrich the models and to integrate with a domain ontology.

6.2.1 i* models

A real case study taken of [79] is presented. The case study describes a generic smart card-based payment system. The goal is to illustrate the analysis of trust-related issued within the full operational and social context of the involver actors. A cardholder depends on a card issuer to be allocated a smart card, for the terminal depends on him to present his card for each transaction. The card issuer in turn depends on the card manufacturer and software manufacturer to provide cards, devices, and software.

The data owner is the one who has control of the data within the card. He depends on the terminal to submit transaction information to the central database. In Figure 6.1 the strategic rational model is presented. The model presents six actors: card holder, terminal owner, card issuer, data owner, card manufacturer, card software.

In Figure 6.1, the goal dependency "**new account be created**" of the card issuer to the data owner means that it is up to the data owner to decide how to create a new account. The card issuer does not care how a new account is created, what matters is that, for each card, an account should be created. An example de task dependency is the card issuer depends on the cardholder to apply for a card via a task dependency by specifying standard application procedures. If the card issuer were to indicate the steps for the data owner to create a new account, then the data owner would be related to the card issuer by a task dependency. In Figure 6.1, the card issuer's dependencies on the card manufacturer for cards and devices, the manufacturers' dependencies on card issuer for payment are modeled as resource dependencies.

The strategic rationale model in Figure 6.2 elaborates on the relationships among cardholder, card issuer, data owner, terminal owner, card manufacturer, and software manufacturer as depicted in the SD model of Figure 6.1. For example, each cardholder has an internal goal of "*Buy Goods with Smart Card*". When an element is expressed as a goal, it means there might be several alternatives to accomplish this, i.e., the cardholder can either "*Buy Goods with Credit Card*", or "*Buy Goods with Stored Value Card*".

6.2.2 Tropos models

In order to validate our approach using the Tropos models, we taken of [80] a case study. This case study is related with the designing distributed agricultural information services for developing countries. A farmer depends on a credit agent to get a credit service, while the credit agent depends on the department agent to check the credit availability. The farmer depends on department agent to advisory services, to get information and logistics arrangement. In Figure 6.3 the actor diagram is presented. The diagram presents five actors: farmer, farm supplier, department assistant, credit agent and regional office.

In the diagram of Figure 6.3 the department agent depends on a farmer to agriculture statistical data is represented as resource dependency. A softgoal dependency is the department agent depends on farmer to increase trust. The department agent depends on a farm supplier to check farm inputs availability, represented as goal dependency. The farm supplier depends on a regional office to consult catalogue.

The goal diagram in Figure 6.4 elaborates on the relationships between the farmer and department agent as depicted in the actor diagram of Figure 6.3. For example the goal Increase productivity have several alternatives to accomplish this, i.e., the farmer can either "*Find credit*", "*Fund farm inputs*", "*Get know-how*" and "*Reduce cost*".

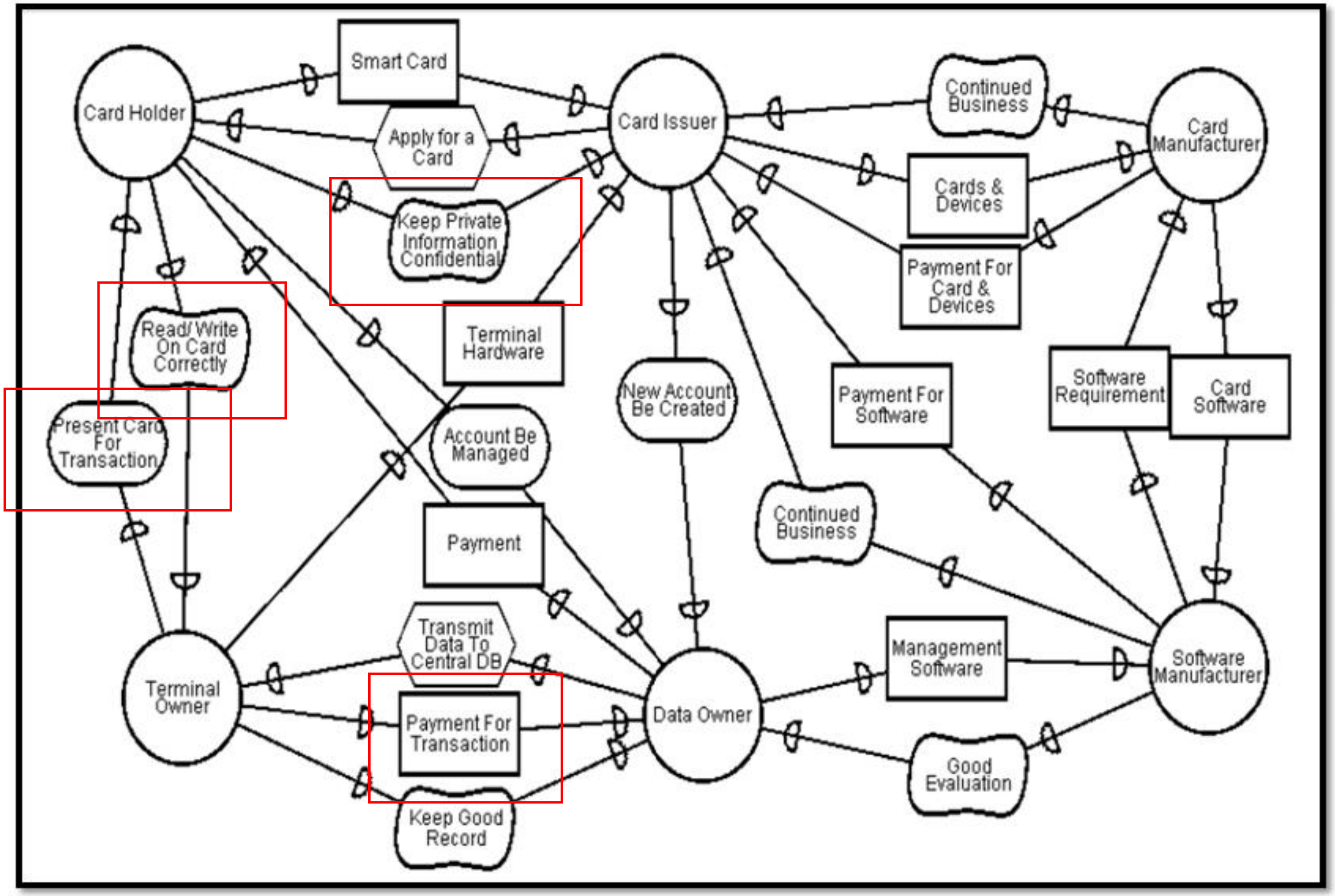
6.2.3 Service-oriented models

In order to validate our approach using the service-oriented models, we taken of [4] a case study. The case study presents the processes to register students at a postgraduate institution (www.cenidet.edu.mx). The institution offers Master and PhD programs in the following areas: computer science, mechanics and electronics. The objective of the case study was to model the specific process to register students in the academic semesters of the postgraduates programs. In Figure 6.6 presents the actors: Bank, vigilance agent, student, finance department, thesis advisor, finance system, group coordination, department chair, student control department, direction, student control system, organization and tracking, professor and planning department. The student control department depends on a vigilance agent to support for the registration process. The thesis advisor presents the services of analyze courses, authorize schedule and propose courses. The student actor depends on a department chair to authorize schedule. The student control system presents the service student registration and subject registration.

The protocol model is presented in Figure 6.6. The goal "*Authorize schedule*" presents the process receive signed schedule, seal schedule and deliver final schedule. The goal "*manage active students*" has several alternatives to accomplish this, i.e. register students and manage school register.

The protocol model in Figure 6.7 elaborates on the relationships among Bank, vigilance agent, student, finance department, thesis advisor, finance system, group coordination, department chair, student control department, direction, student control system, organization and tracking, professor and planning department as depicted in the global model of Figure 6.5.

It is important mention that the seven models mentioned previously are applied the general and specific semantic annotation suggestions. On the one hand, all the models are presented using the general semantic annotation suggestions, the domain ontologies applied in these models are presented too. On the other hand, the same models are presented using the specific semantic annotation suggestions into OntoSem ontology.



Model elements

6 actors
Dependencies
- 3 goals
- 6 softgoals
- 2 task
- 10 resources

Figure 6.1 Strategic dependency model for the case study

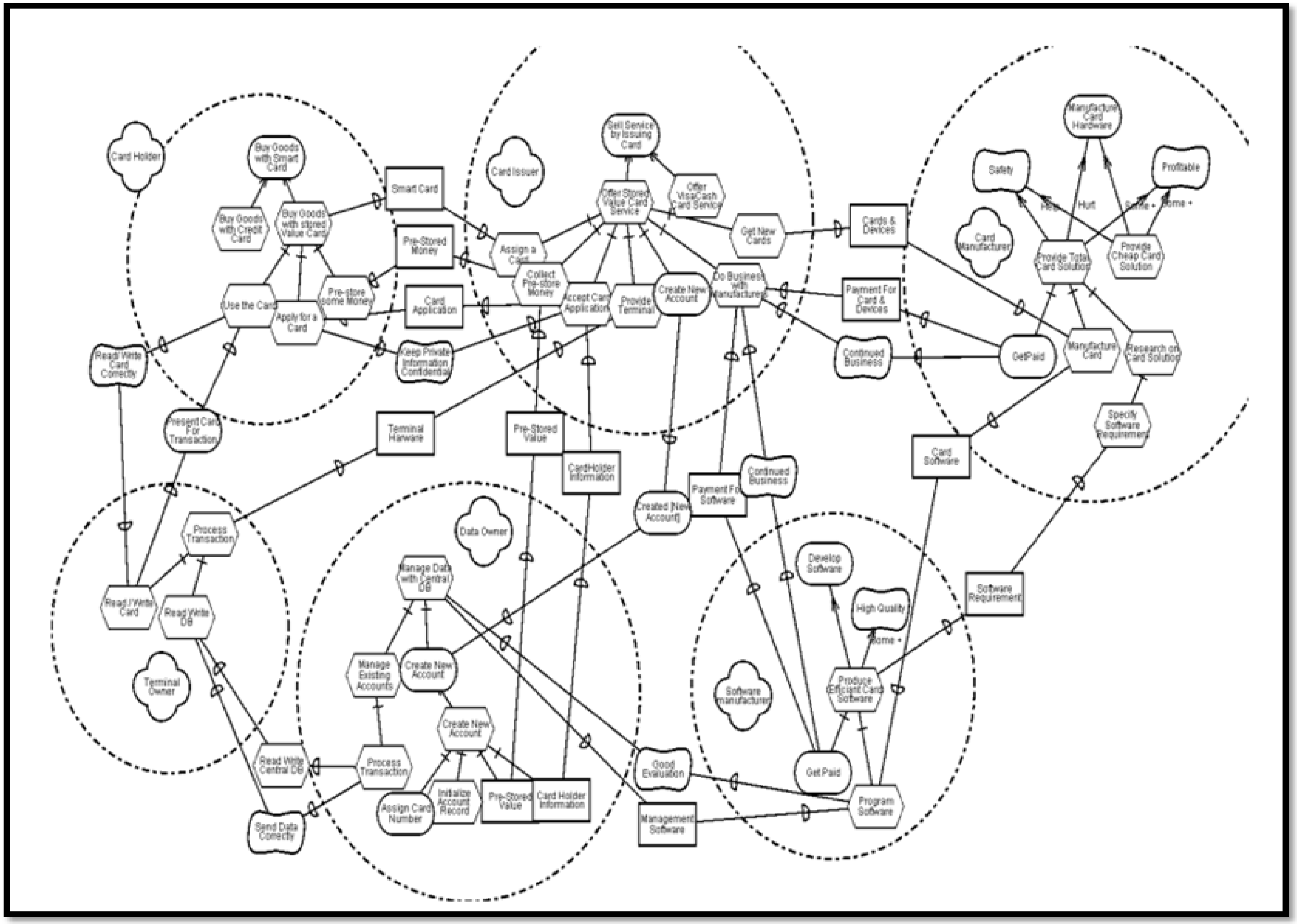


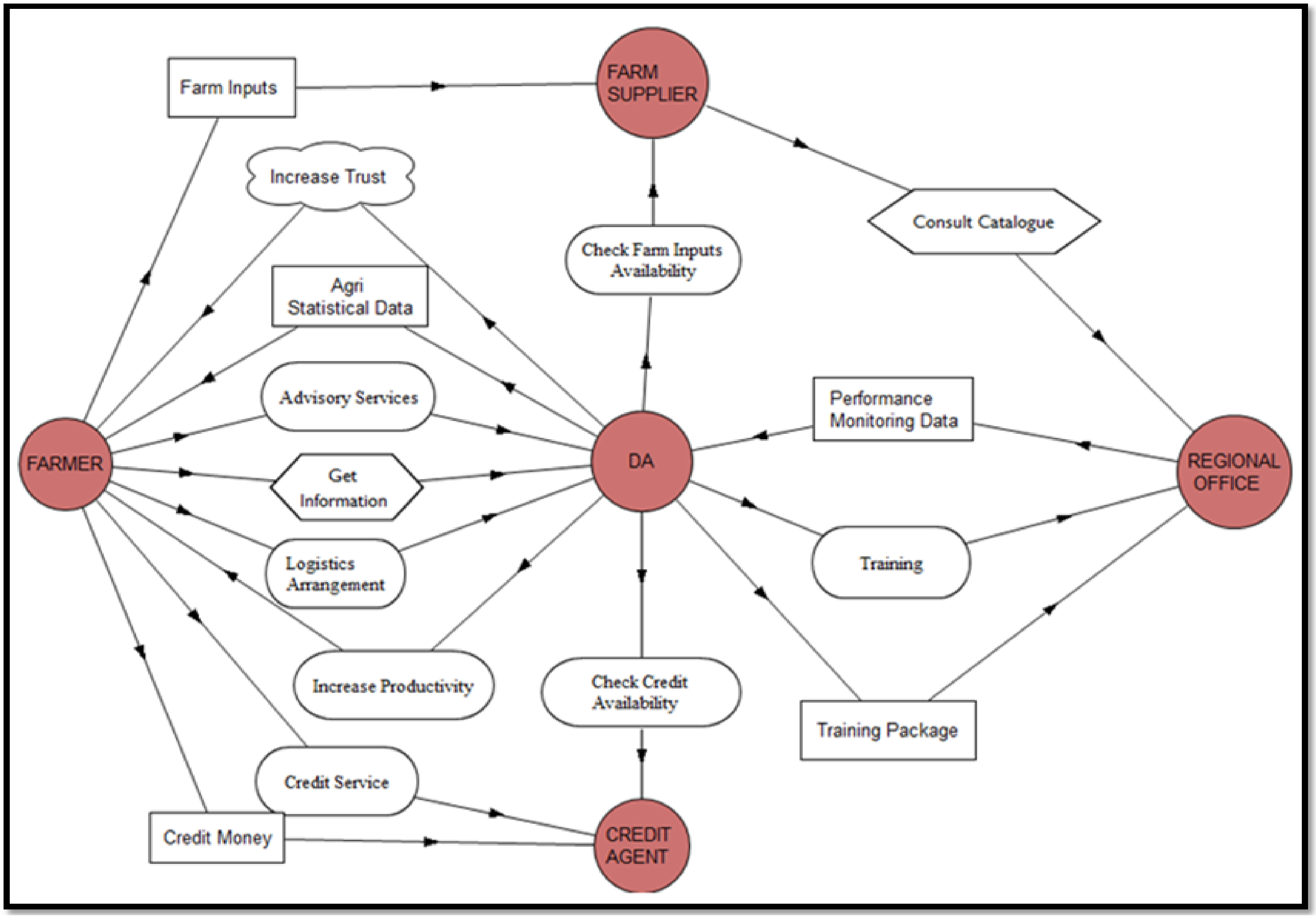
Figure 6.2 Strategic rationale model for the case study

Model elements

6 actors type "position"
Intentional elements
- 9 goals
- 3 softgoals
- 28 tasks
- 2 resources

Dependencies
- 3 goals
- 6 softgoals
- 2 task
- 10 resources

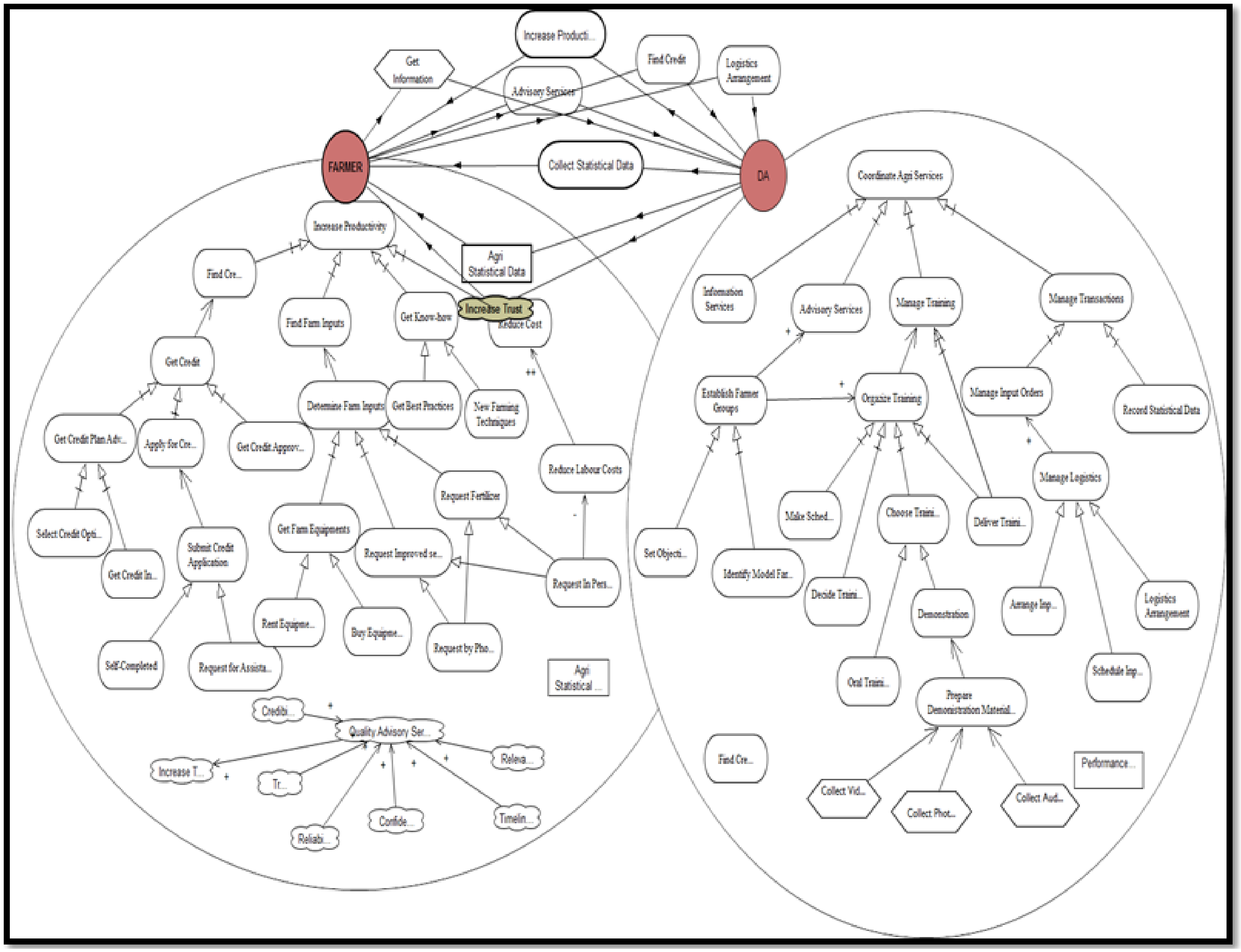
4 Means-end
10 And Decomposition
2 Contribution



Model elements

5 actors
Dependencies
- 7 goals
- 1 softgoals
- 2 plans
- 5 resources

Figure 6.3 Actor diagram for the case study



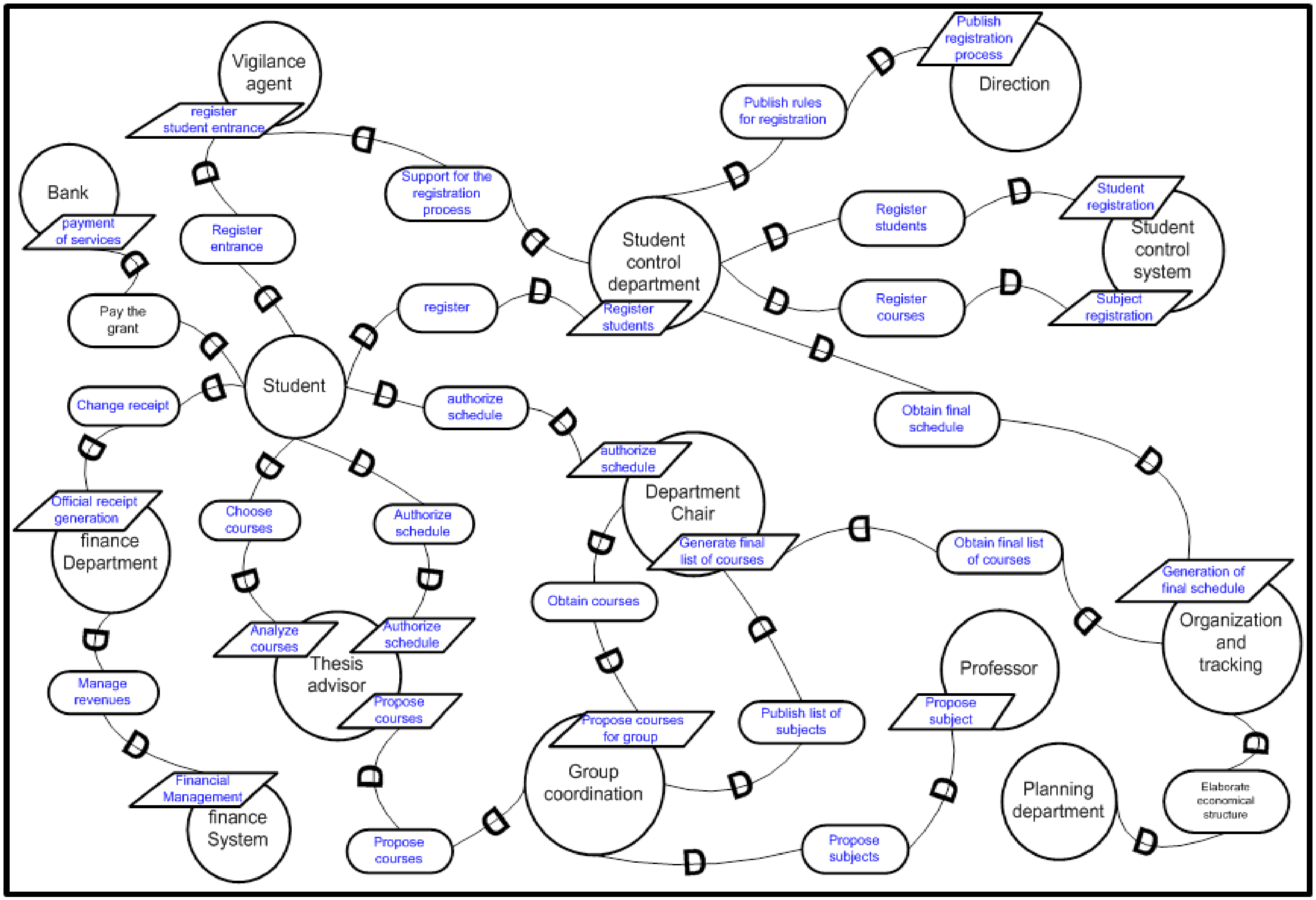
Model elements

2 actors
Intentional elements
- 22 goals
- 8 softgoals
- 3 plans

Dependencies
- 5 goals
- 1 softgoals
- 1 task
- 1 resources

1 Means-end
9 And Decomposition
14 Or Decomposition
7 Contribution

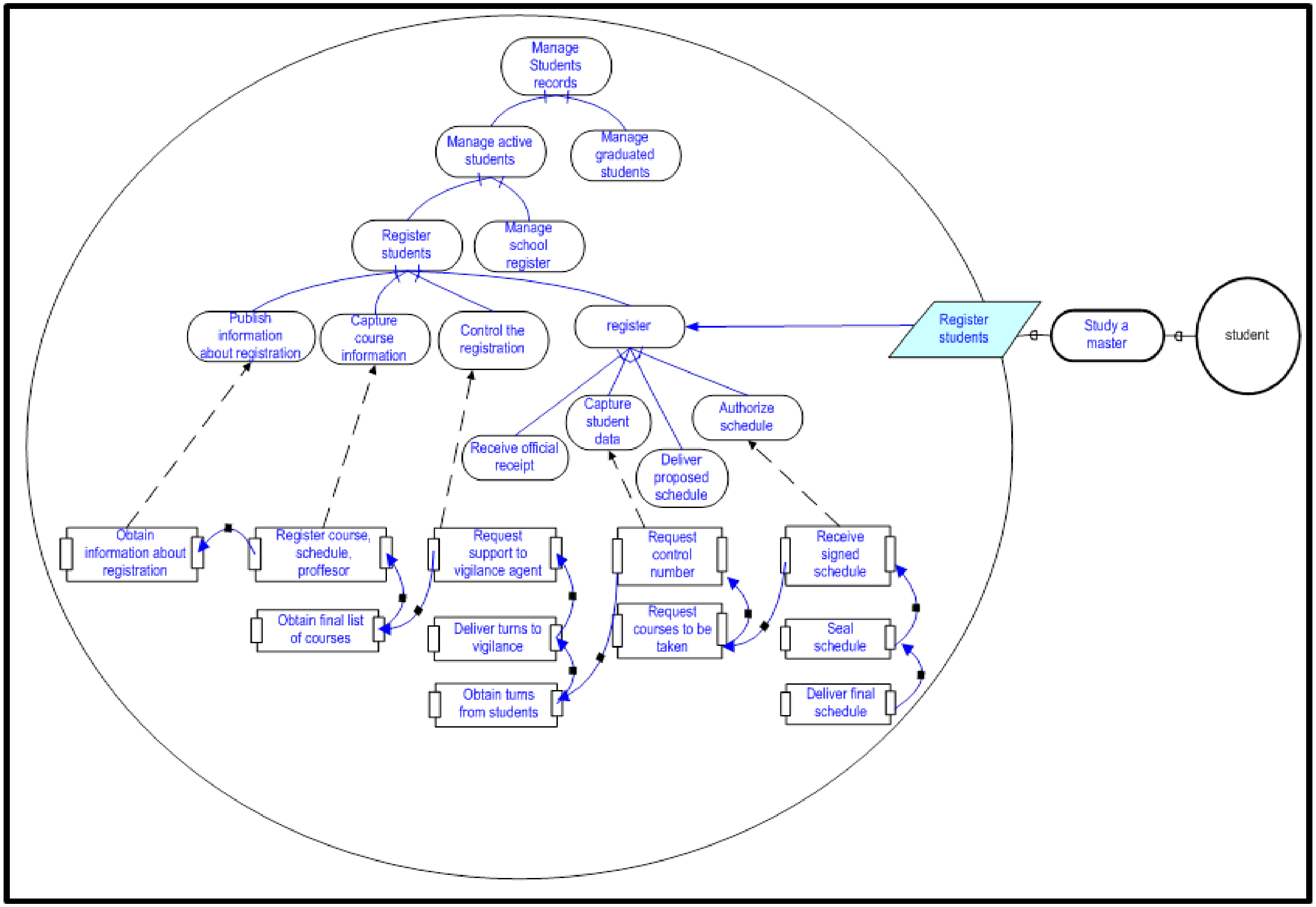
Figure 6.4 Goal diagram for the case study



Model elements

14 actors
Dependencies
- 19 goals
- 16 services

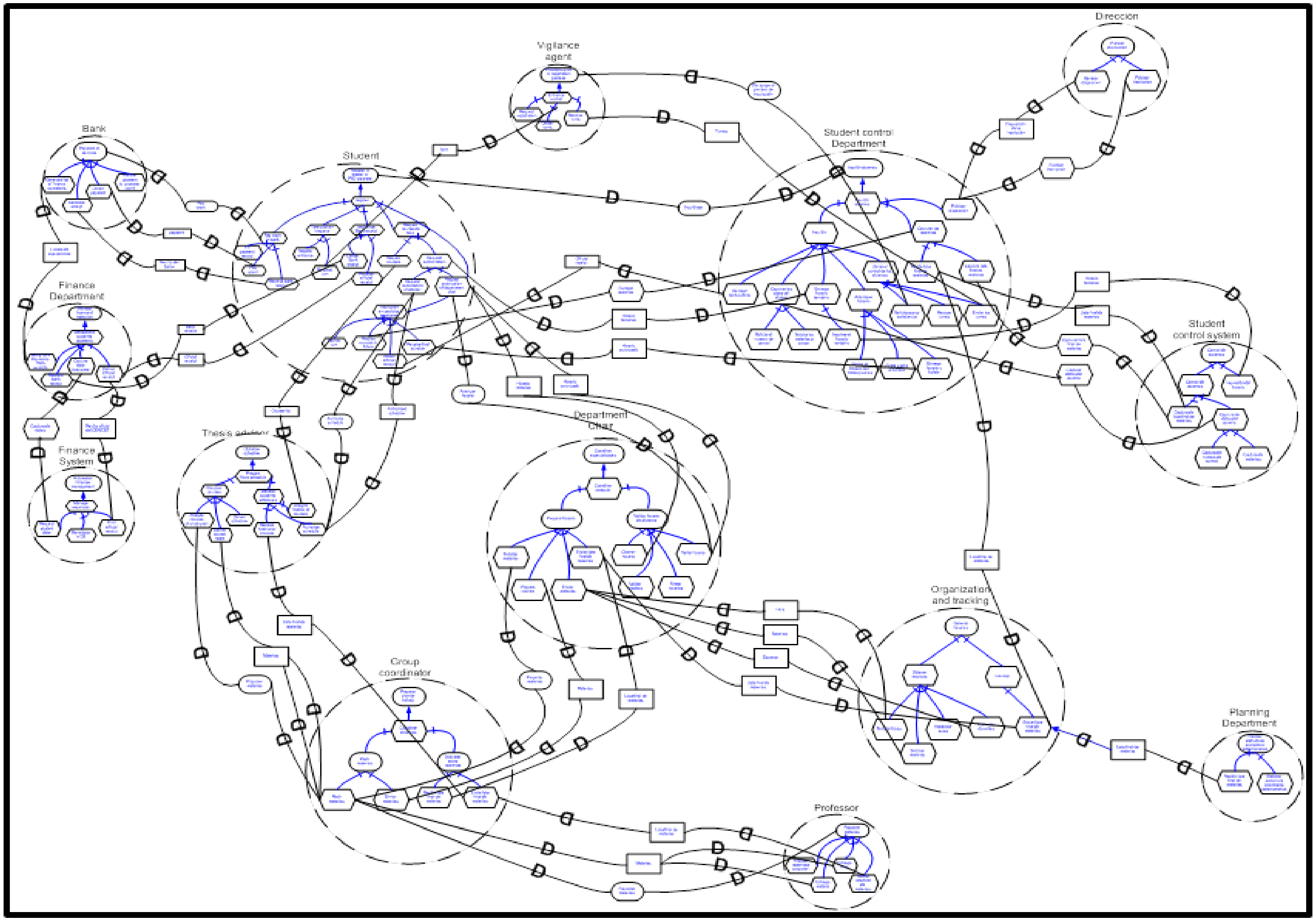
Figure 6.5 Global model for the case study



Model elements

6 actors
Intentional elements
- 13 goals
- 11 Process
Dependencies
- 1 goal
- 1 services
4 And Decomposition
1 Services goal-relationship
10 Process-relationship
5 Process dependency
2 Contribution

Figure 6.6 Process model for the case study



Model elements

14 actors
Intentional elements
- 17 goals
- 100 task
Dependencies
- 8 goal
- 15 Tasks
- 30 resources
34 And Decomposition
10 Means-end

Figure 6.7 Protocol model for the case study

6.3 Following the annotation process flow

The inputs of our approach are: i) An organizational model represented in the variants i*, Tropos and Service-oriented i*, ii) the set of semantic annotation suggestions and iii) the domain ontology previously validated with online tool RDF Validation Service (http://www.w3.org/RDF/Validator/).

The processes flows in order to integrate an organizational model with a domain ontology are: Step 1 Enrich organizational model with semantic annotation, Step 2 Export the model into iStarML format and Step 3 Integrate the model with a general or domain ontology. In Figure 6.8 Annotation process flow is presented. In the next sections each step are described.

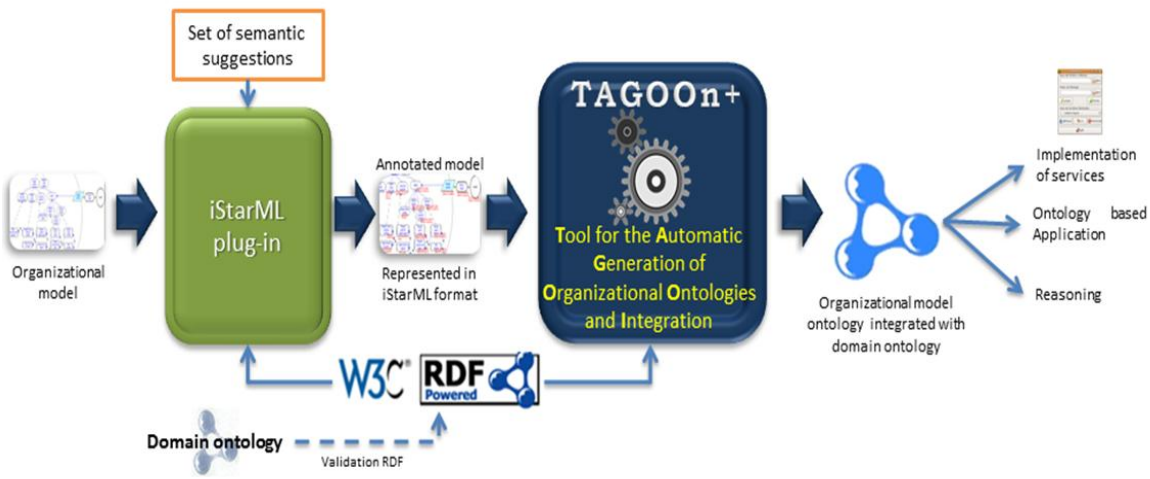


Figure 6.8 Annotation process flow

6.3.1 Step 1: Enrich organizational models with semantic annotations

In order to annotate the organizational models through of semantic annotation is necessary to attend first the general or specific suggestions (section 4.2.1.3). The suggestions are the guideline to annotate the models. The idea is to annotate all the elements with one or more domain concepts. This concept should be congruent with the description of the element. In the case of specific suggestions the element should mapped inside of the superconcepts described in Table 4-6. We propose to add the semantic annotation for each element of the model using the *"@"* symbol. In order to demonstrate the feasibility and performance of our approach the case study was tested with the general semantic annotation (section 6.3.1.1) and the specific semantic suggestions (6.3.1.2).

Obtaining a domain ontology

A previous step to annotate the model is to select a domain ontology. Some scenarios to obtain the ontologies are presented below.

a. Existing different repositories of ontologies to obtain domain ontology, such as Swoogle[6], BioPortal[7], Protégé Ontology Library[8] and others.
b. Sometimes, when an organization is modeled the analysts create ontologies to describe the entities and its relationships.

In both cases the ontology should be verified with the tool http://www.w3.org/RDF/Validator/. This online tool validates and checks the consistent of the ontology. In this way, it is assurance that the ontology will not generate problems to integrate into organizational model. Moreover, it is important mention that all the online domain ontologies not cover the social and structure processes of the modeled organization. In order to resolve this situation is possible to add domain concepts, however these additional concepts should not repeat with the exist concepts, and should add only when the ontology selected present missing information. In order to carry out the annotation process should follow the semantic annotation suggestions proposed in the chapter 4.

6.3.1.1 Annotating models with general semantic annotation suggestions

Annotating i* model

The domain ontology to annotate the i* models is available in (http://lfe.uni-muenster.de/Products/DictOnt/Data/Ontologies/lfe_2007.owl). This domain ontology describes electronics advices, software, roles and design system. It is composing by 222 classes. A fragment of this ontology in Figure 6.9 is shown.

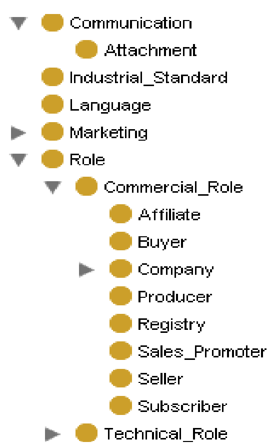


Figure 6.9 Fragment of domain ontology applied to i* models

The i* strategic dependency model in Figure 6.1 was shown. In this model exists two actors "*Card Holder*" and "*Card Issuer*". The GSAS No. 1 (General Semantic Annotation Suggestions No.1) suggests that "*An actor or actor types should be mapped into domain concept that describes an organization, agent, or entity tangible or intangible*." In order to annotate the actor element, the

[6] http://swoogle.umbc.edu/
[7] http://bioportal.bioontology.org/
[8] http://protegewiki.stanford.edu/wiki/Protege_Ontology_Library

analyst should going in-deph of domain ontology and to find-out the more appropriate domain concepts. In this model the actor "*Card Holder*" was annotated with the domain concepts "*@user*" and *"@affiliate*". The other actor "*Card Issuer*" was annotated with the concepts "*Sales_Promoter*". These domain concepts meet with the suggestions, due to a "*user*" or "*affiliate*" or "*Sales_Promoter*" are agents and entities tangible. A fragment of annotated model in Figure 6.10 Fragment of annotated strategic dependency model is presented.

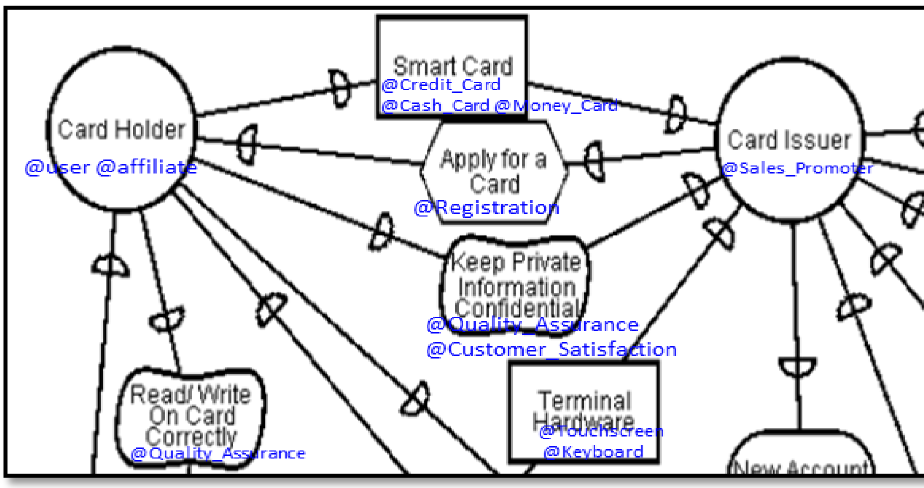


Figure 6.10 Fragment of annotated strategic dependency model

In the case of resource element the GSAS 5 (General Semantic Annotation Suggestion) suggests that "*A resource should be mapped into domain concepts that represent an object physical or informational entity.*" The resource "*Smart Card*" was annotated with the domain concepts "*@Credit_Card*", "*@Cash_Card*" and "*Money_Card*". These concepts meet with the general suggestions, because the three domains concepts are informational entity.

In order to annotate the i* strategic rationale model (Figure 6.2) the domain ontology used is the same in the previous model. The GSAS No.2 (General Semantic Annotation Suggestions No.2) suggests that "*A goal should be mapped into domain concepts that describe and clear and precise condition, interest or desire***.**" In the model, the goal element "*Create new Account*" was annotated with the domain concept "*Registration*". This concept meets with the general suggestion; due to registration is a precise condition or desire of a stakeholder.

The GSAS No.3 (General Semantic Annotation Suggestions No.3) indicates that "*A softgoal should be mapped into domain concepts that describe an interest or desires not clear-cut satisfaction criteria*". The softgoal "*Send Data Correctly*" was annotated with the concepts "*Information_Retrieval*" and "*Quality_Assurance*". In this case both domain concepts meet the suggestions due to the concepts are desires not-clear criteria. A fragment of this model annotated using the general semantic annotation suggestions in Figure 6.11 are presented.

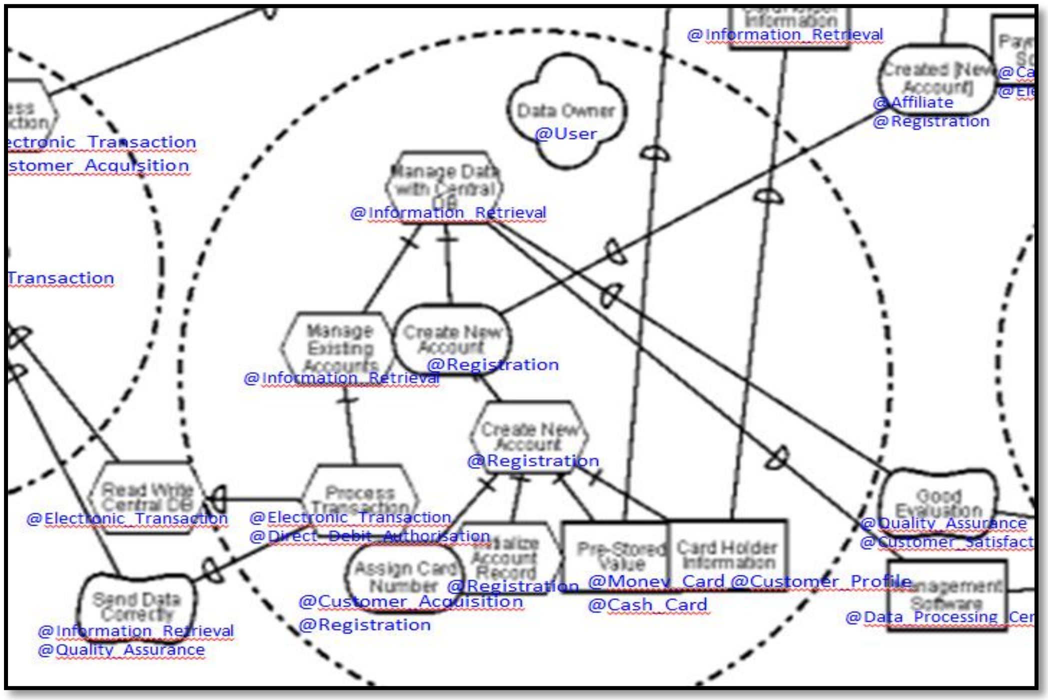


Figure 6.11 Fragment of annotated strategic rationale model

Annotating Tropos models

The domain ontology to annotate the Tropos models was done by us to guide the annotation process. This domain ontology describes activities, object and roles related with farms. It is composing by 76 classes. A fragment of this ontology in Figure 6.12 is shown.

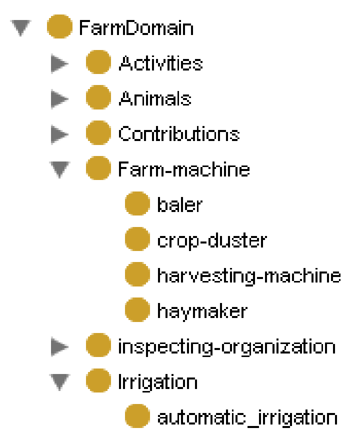


Figure 6.12 Fragment of domain ontology applied to Tropos models

The actor diagram in Figure 6.3 was shown. In this model is shown the task "*Get information*". The GSAS 4 (General Semantic Annotation Suggestions) suggests that "*A task or plan should be mapped into domain concepts that describe a clear action or activity*". The task "*Get information*" was annotated with domain concepts *"@obtain-information*". In this case "*obtain-information*" is an activity or clear action, then the concept meet with the previous suggestions. A fragment of this model annotated using the general semantic annotation suggestions in Figure 6.13 are presented. The goal element "*Logistics arrangement*" was annotated with domain concepts *"@schedule_input*", "*@logistic*" and "*@schedule_crops*". The three concepts meet with the GSAS No.2 "*A goal should be mapped into domain concepts that describe and clear and precise condition, interest or desire."* (see Figure 6.13).

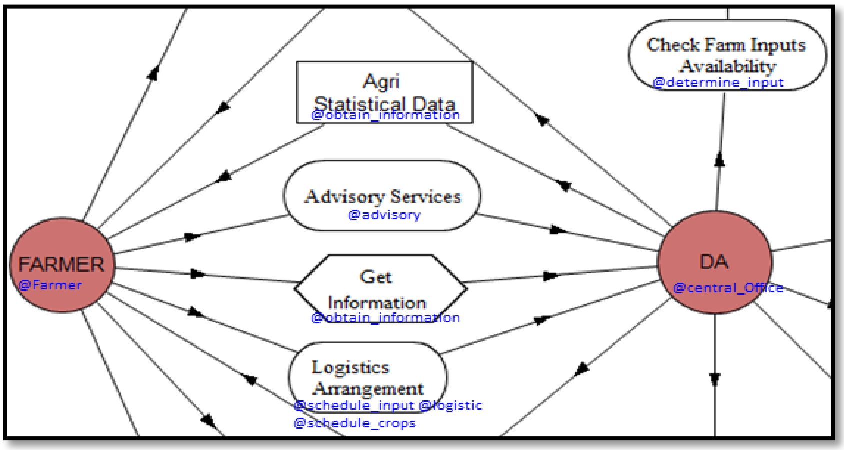


Figure 6.13 Fragment of annotated actor diagram

In the Figure 6.4 the Tropos goal diagram is shown. Following the guideline, the goal "*Get Credit Plan Advise*" was annotated with the domain concepts *"@obtain_information*", "*@credit*" and "*obtain_resources*". It is important mention that an element of the model could be annotated with one or more domain concepts the aim is to clarify the hidden semantic, categorize and unified the elements and can implement future services. Other annotated elements in the goal diagram in Figure 6.14 are observed.

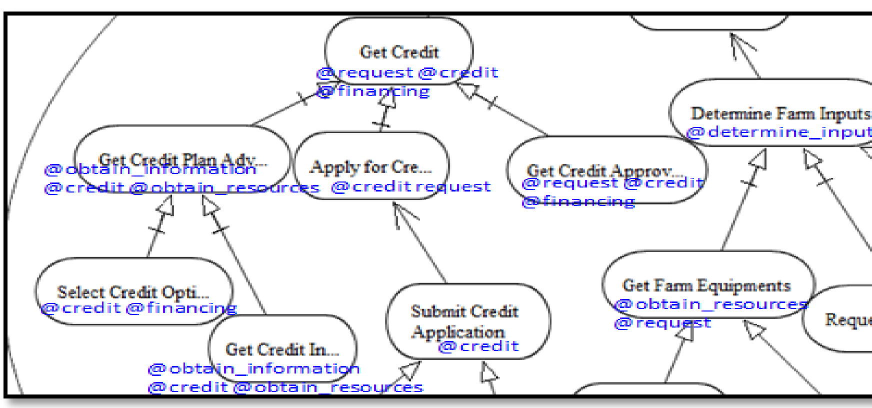


Figure 6.14 Fragment of annotated goal diagram

Annotating service-oriented i*

The domain ontology to annotate the service-oriented models is available in (http://www.webkursi.lv/luweb05fall/resources/university.owl). This domain ontology describes departments, processes, roles, courses, programs related with academic processes. It is composing by 79 classes. A fragment of this ontology in Figure 6.15 is shown.

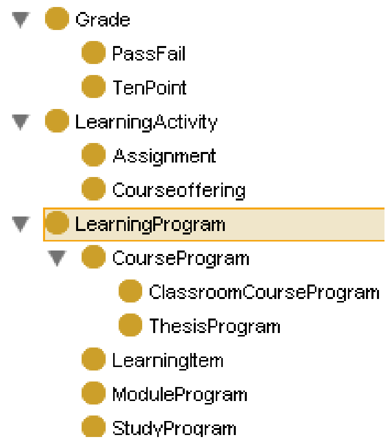


Figure 6.15 Fragment of University ontology applied to services models

The service-oriented global model in Figure 6.5 is shown. The GSAS No.6 suggestions: "*A service should be mapped into domain concept that represents a functionality or specification of services*". In the global model is shown the service "*Official receipt generation*". In this case was annotated with the domain concepts "*@Payment*" and "*@Enrollment*". Both concepts describe a functionality of department of finance. A fragment of annotated global model in Figure 6.16 is shown.

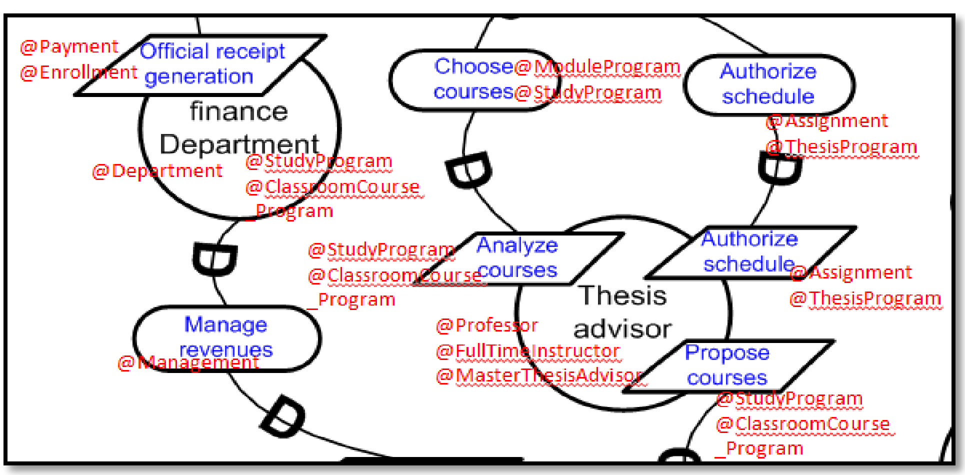


Figure 6.16 Fragment of annotated global model

The service-oriented protocol model in Figure 6.6 was shown. The domain ontology used to annotate this model in Figure 6.15 is shown. The GSAS No. 7 suggests "*A process should be* *mapped into domain concepts that describe a clear action or activity*". In the protocol model the process "*Request support to vigilance agent*" was annotated with domain concepts *"@Supporting*" and "*@Assignment*". In this case "*supporting*" and "*assignment*" are related to actions or activities. So, both domain concepts meet with the suggestion. A fragment of the process model in Figure 6.17 is shown.

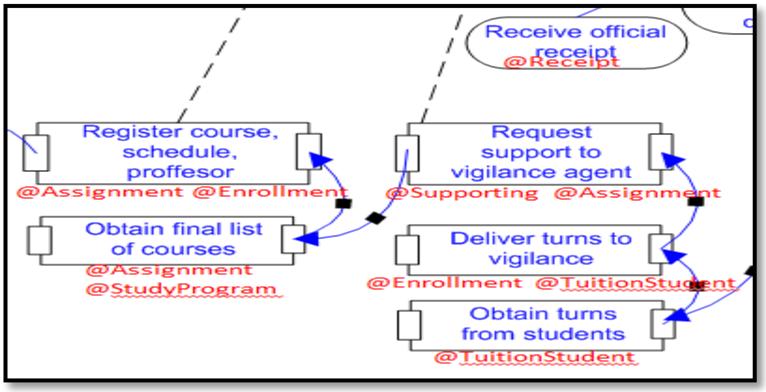


Figure 6.17 Fragment of annotated process model

The service-oriented protocol model in Figure 6.7 was shown. The domain ontology used to annotate this model in Figure 6.15 is shown. In this model the goal "*Register in master or PhD program*" was annotated the domain concept "*@Enrollment*".

In this example is clear our guideline. The aim of this thesis is the enrichment of organizational models by means of domain concepts. The advantages of the semantic annotation are a way of linking domain ontology and data to align the semantics defined heterogeneously.

Hence, semantic annotation helps to be more precise and efficient information retrieval, achieving to share common understanding within a community. A fragment of annotated global model in Figure 6.18 is shown.

The enrichment of organizational models with domain concepts allows us to group the elements of the model according to similar situations and description. In this way, the domain concepts permit the standardization of elements by means of common denominators. Moreover, the grouped elements with the same annotation could be useful to implement services.

In the Figure 6.18 the goal "*Register in master or PhD program*", the tasks "*register*", "*Take position in queue*", "*Request turn*", and the resource "*turn*" were annotated with the domain concept "*@Enrollment*". So, all these element of the model could implement the service "*Enroll*" or "*Enrollment*" grouping semantically different elements.

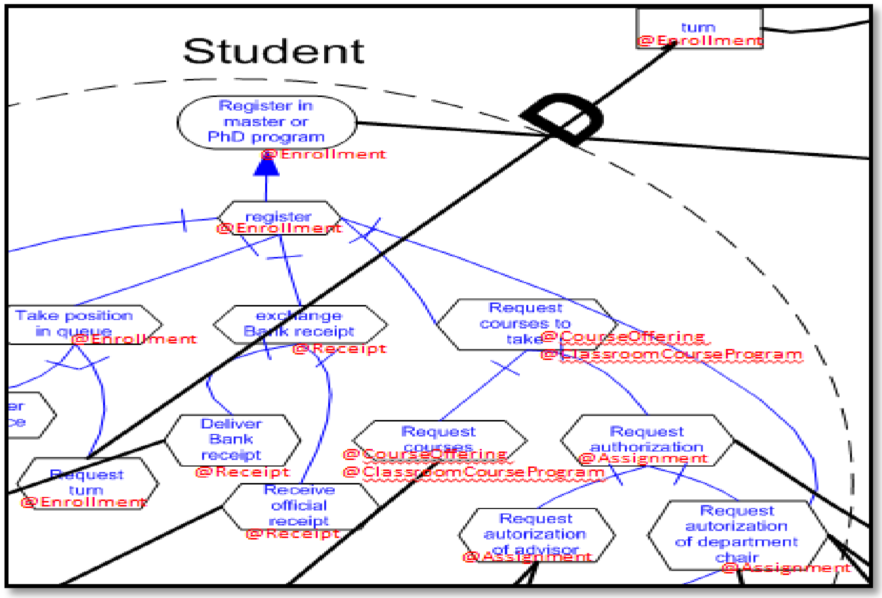


Figure 6.18 Fragment of annotated protocol model

6.3.1.2 Annotating models with specific semantic annotation suggestions

In order to carry out the semantic annotation of elements of the models is used the OntoSem ontology and the suggestions describes in Chapter 4. The elements of the model could be annotated with one or more concepts. In order to annotate each element with domain concept from OntoSem, the process is for example: if the suggestions indicate that the element "*task*" should annotated with superconcept "*Social-event*" then going in-depth of the superconcept and to find out the more appropriate domain concept for the task element. This concept should be congruent with the description of the element. The idea is to annotate all the element of the model with one or more domain concepts, such annotation provide enrichment to the element description and allow the implementation of services.

Annotating i* models

In the Figure 6.1 the strategic dependency model was shown. In this model the goal element "*Present Card for Transaction*" is shown. The merge axioms suggests that "*ME:* ***Goal*** $\xrightarrow{AB}$ *OC:mental-event v OC:social-event v OC:mental:object*". The element goal can be annotated with the superconcepts *mental-event, social-event or mental:object.* Now, the process to annotate is going in-deep in these superconcepts and to found out the more appropriate concepts that describe the goal element. In this way, the concept "*identify*" from "*mental-event*" superconcept describes "*to fix the identity of something or someone*". Moreover, the concept "*authenticate*" from "*social-event*" superconcept describes "*to verify the identity of someone or something in order to grant access privileges*". Finally the concept "*negotiate-transaction*" from "*social-event*" describes "to work out the terms of a transaction in order to reach an agreement". In this case the transaction in the goal element requires that customer is authenticated to pay, and to start a transaction is should identify the customer, so these concept describe the element analyzed. In this way, the goal element *"Present Card for Transaction"* was annotated the concepts: *"@identify"*, *"@authenticate"* and "@negotiate-transaction". In the Figure 6.19 the hierarchical of these concepts is shown. In Figure 6.20 a fragment of annotated strategic dependency model is shown.

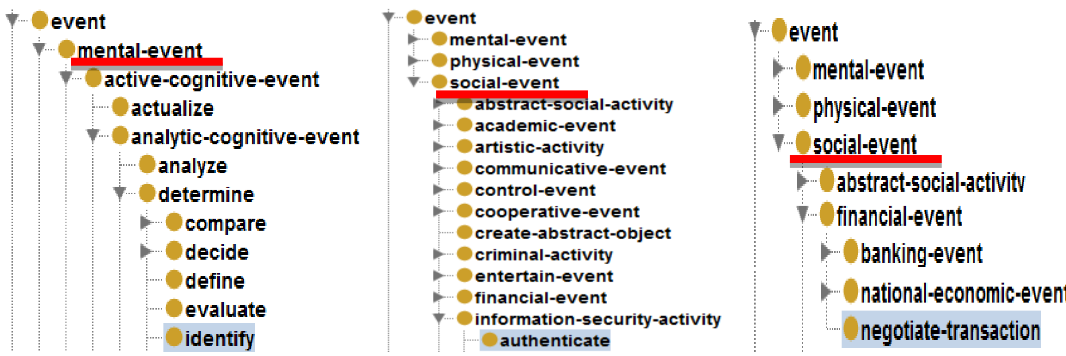


Figure 6.19 Hierarchical of domain concepts "*identify*" (left), "*authenticate*" (center) and "*negotiate-transaction*" (right) to annotate the goal element "*Present Card for Transaction*"

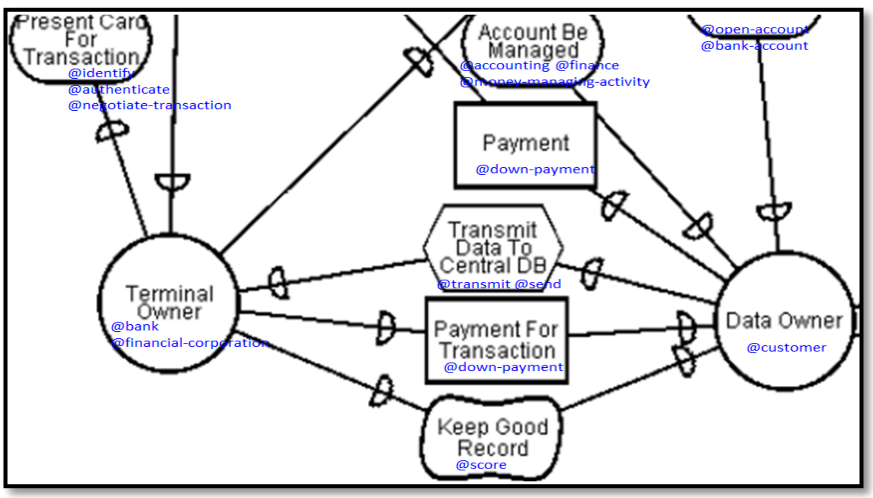


Figure 6.20 Fragment of annotated strategic dependency model

In the Figure 6.2 the strategic rationale dependency model was shown. In this model the task element "*Create New Account*" and the goal element "*Create New Account*" are shown. In the case of element task the suggestions indicates "*ME: **Task** $\xrightarrow{AB}$ OC:active-cognitive-event v OC:social-event v OC:physical-event".* Both elements were annotated with the domain concepts *"@open-account"* and *"@bank-account"*. So, the concept "*open-account*" from "*social-event*" superconcept describes "*The event of opening a bank account.*" The hierarchical of this concept in Figure 6.21 is present. Other concept is *"bank-account"* means "*money deposited in a bank and credited to the depositor*". Following the suggestions both concepts added description the goal and task elements, and clarify the semantic of these elements. Moreover, the semantic annotation could help to implement a service called "*Open-account*" that integrates two different elements semantically. A fragment of annotated strategic rational model in the Figure 6.22is shown.

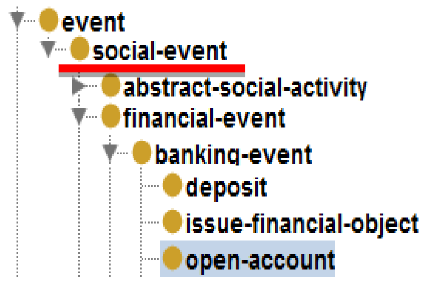


Figure 6.21 Hierarchical of "open-account" concept

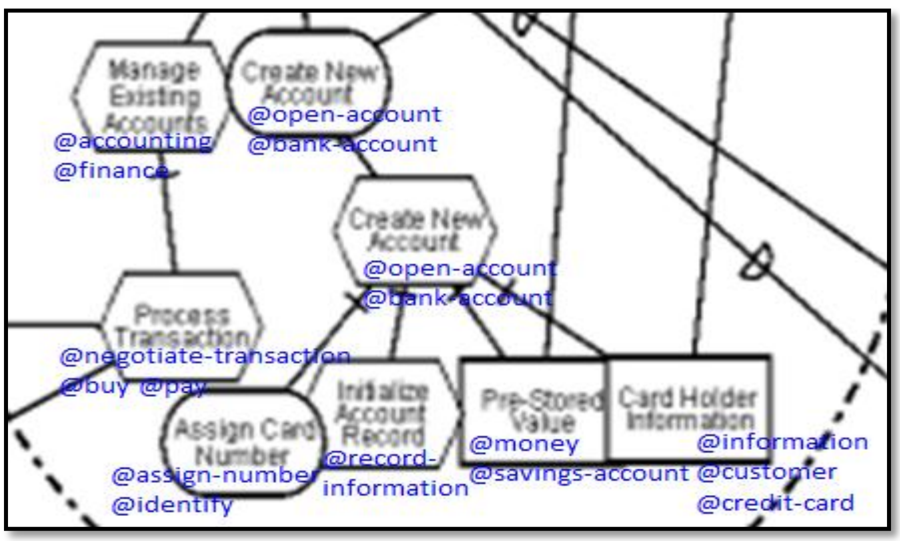


Figure 6.22 Fragment of annotated strategic rationale model

Annotating Tropos models

In Figure 6.3 the actor diagram was shown. This model presents the resource element "*AgriStatistical Data*". The suggestions describes "*ME:* ***Resource*** $\xrightarrow{AB}$ *OC:physical-object v OC:mental-object". A resource can be annotated with the super concepts "physical-object" and "mental-object".* Now, the process is going in-deep in the superconcepts physical-object and mental object and to found out the more appropriate concepts that describe the resource element.

The concept "*statistical-number*" that describes "*A number that represents certain data assembled in such a way that it presents significant information*", in this case this concept describe our resource analyzed. Other domain concept is "*information*" from "*mental-object*" superconcept. This concept describes "*Anything having mental content that can be perceived by an individual or transferred from one individual to another to create new ideas, etc.*" The hierarchical of these domain concepts in Figure 6.23 are presented. A fragment of annotated actor diagram in Figure 6.24 is shown.

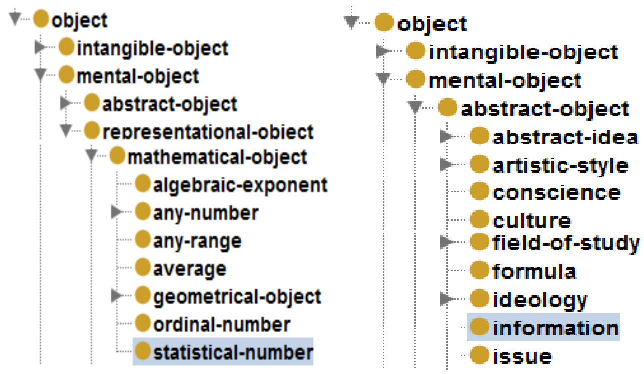


Figure 6.23 Hierarchical of domain concepts "*statistical-number*" (left), "*information*" (right)

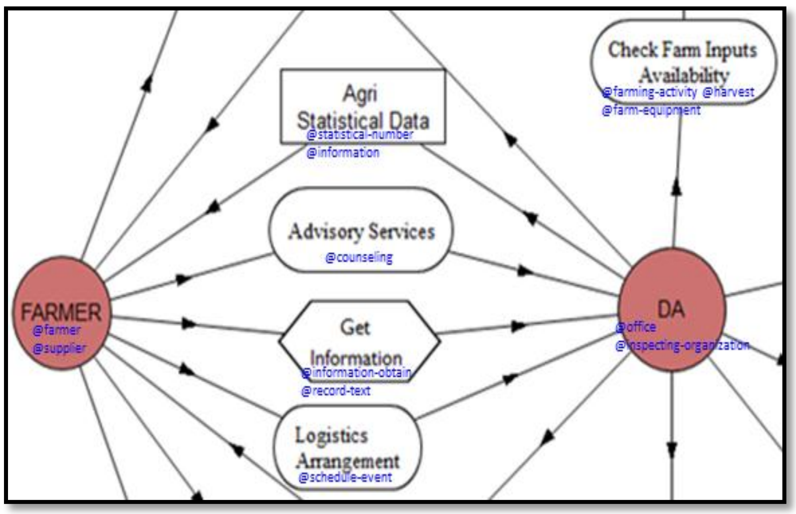


Figure 6.24 Fragment of annotated actor diagram

In Figure 6.4 the goal diagram is shown. The goal element "*Decide training contents*" is shown in this model. This element should be annotated with the superconcepts ***"mental-event***", "***social-event***" or "***mental-object***". The concepts selected to annotate this element are: *"@communicate-content*" from "*mental-object*", this concept describes "*information conveyed through communication*".

Other concept annotated is *"@example*" and "*@fact*" from mental-object means "*A step-by-step problem-solving procedure, especially an established, recursive computational procedure for solving a problem in a finite number of steps*". A fragment of the annotated diagram in Figure 6.25 is shown.

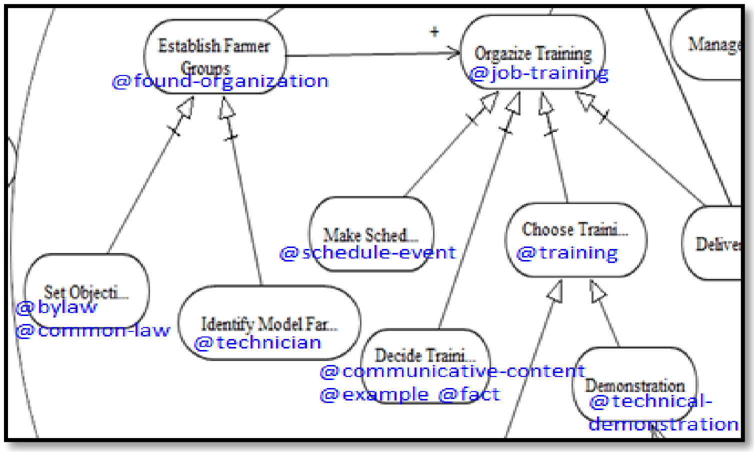


Figure 6.25 Fragment of annotated goal diagram

Annotating service-oriented i*

In Figure 6.5 the service-oriented global model is shown. In this model the goal "*authorize schedule*" were annotated with the concepts *"@approve"*, *"@confirm"* and "*@accept*" all these concepts from "*mental-event*". Other specific suggestion is about the service element; it should be annotated with the superconcepts "*social-event*". In the model the "*authorize schedule*" service was annotated with domain concepts "*@publish*" and *"@record-text*" from social-event superconcept.

The concept "*publish*" means "*To disseminate results and findings by writing them up and making copies available to a select or general audience*." This concept clarifies the semantic and the description of elements of the model. Other elements annotated are presented below.

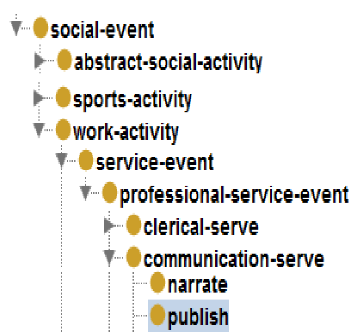


Figure 6.26 Hierarchical of domain concepts "*publish*"

The hierarchical of concept "*publish*" in Figure 6.26 is shown. A fragment of annotated service-oriented global in Figure 6.27 is shown.

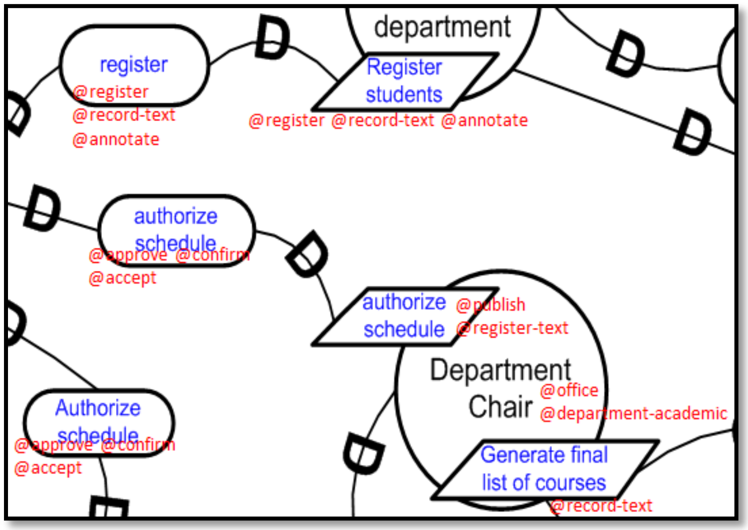


Figure 6.27 Fragment of annotated global model

In Figure 6.17 the service-oriented process model is shown. In this model the element process "*Register course, schedule, professor*" is shown. The process should be annotated with the superconcepts "*active-cognitive-event*", "*social-event*" or "*physical-event*". This element was annotated with concepts "*@register*" from "*social-event*" and "*record-text*" from "*physical-event*". The concept "*register*" means "*to enter in a list such as to enroll, sign up, admit someone (as to a hospital), etc.*" and the concept "*record-text*" means "*record events on paper or on-line*". So, both concepts describe the process element. In Figure 6.28 a fragment of annotated process model is shown.

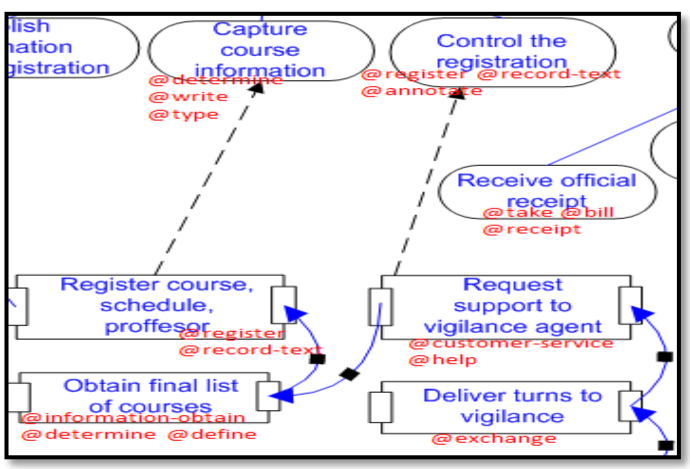


Figure 6.28 Fragment of annotated process model

In Figure 6.7 the service-oriented protocol model is shown. The task "*Analyze courses of study plan*" and "*Analyze final list of courses*" is shown in this model. Both task elements should annotate with the superconcepts "*active-cognitive-event*", "*social-event*" or "*physical-event*". In this case both elements were annotated with concepts "*@revise*" and "*@coordinate*" from social-event superconcept. When an element is annotated using domain concept allow us to categorize different elements, typically, a same domain concept can refer to different referents in different models. Hence, our research improves the interoperability semantic among different variants of modeled. In Figure 6.29 a fragment of annotated process model is shown.

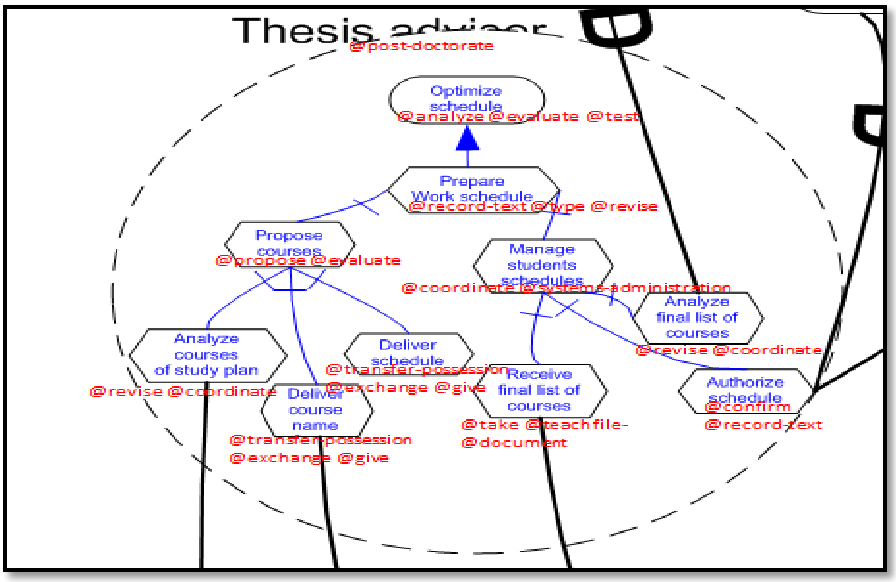


Figure 6.29 Fragment of annotated process model

Scoping of this research is enriching the organizational models by means of semantic annotation. The heterogeneity of modeling techniques makes it difficult to manipulate the distributed process models in a centralized manner. Ontologies and semantic annotation provide a means to tackle this problem. It is important mentioned that when a model is enriched with concept from general ontology like "*OntoSem*" is more enriching that with a domain ontology; this different is due to that the general ontology are often characterized as representing common sense concepts, i.e. those that are basic for human understanding of the world. The general ontology is also sometimes referred to as foundational ontologies or universal ontologies. The general ontology allows the **reasoning** because all the concepts are **inside a hierarchical,** such as it shown in the models of case study.

6.3.2 Step 2: Export the model into iStarML format

This step consists of exporting the organizational models analyzed previously into iStarML interchange format. In the annotation process flow in Figure 6.8 was shown. The input in this phase is the organizational model and the semantic annotation and the output is the model represented in iStarML file. This file should contain the XML attribute "*sannotation*". This tag stores the domain concepts that describing the element of the model. In order to generate this file there are two options. On the one hand, the model can be designed using the JUCMNav tool [76] and to add the extended plug-in to export the iStarML file. This plug-in extended is a contribution of this thesis; the model is exported with the new XML tag. On the other hand the model can be designed using any tool of modeling, for example "*OME*" [81], "*TAOM4E*" [82] and other, finally export the model using any external tool and to add manually the tag "*sannotation*" with its respectively domain concepts. A fragment of each models exported into iStarML is shown below.

In Figure 6.20 a fragment of the annotated strategic dependency model was shown. This fragment exported to iStarML file in Figure 6.30 is shown. In this model the domain ontology to card system was applied (Figure 6.9). The actor "*Software manufacturer*" and "*Card manufacturer*" are shown. The softgoal "*Read/Write on cards correctly*" and goal "*Presented card for transaction*" is shown too. Each element of the model presents the sannotation attribute with its respective semantic annotation. In this model the general semantic annotation are applied.

In Figure 6.22 a fragment of the annotated strategic rationale model was shown. This fragment exported to iStarML file in Figure 6.31 is shown. In this model the concepts of the OntoSem ontology has been applied. The actor "*Card Holder*" and the task elements "*Use the card*", "*Pre-store some Money*" and "*Buy goods with smart card*" are presented with its respective semantic annotation. In this model the specific semantic annotation are applied.

In Figure 6.13 a fragment of the annotated Tropos actor diagram was shown. This fragment exported to iStarML file in Figure 6.32 is shown. In this model the Farm ontology was applied (Figure 6.12). The actors "*Farmer*", "*Credit agent*" and "*Regional office*" are presented with its semantic annotation. The resource "*Credit money*", the goals "*Credit service*" and "*Increase productivity*" are presented with its respective semantic annotation too. In this model the general semantic annotation are applied.

In Figure 6.25 a fragment of the Tropos goal diagram was shown. This fragment exported to iStarML file in Figure 6.33 is shown. In this model the concepts of the OntoSem ontology has been applied. The actor "*Farmer*" and the goals "*Select credit option*", "*Get credit info*" are presented. The plan elements "*Request for Assistance*" and "*Rent equipment*" are shown with its respective semantic annotation too. In this model the specific semantic annotation suggestions are applied.

In Figure 6.16 a fragment of the annotated service-oriented global model was shown. This fragment exported to iStarML file in Figure 6.34 is shown. In this model the University Ontology was applied (Figure 6.15). The actor "*Bank*", "*Finance Department*", "*Finance system*" and the service elements "*payment of services*", "*Official receipt generation*" and "*Propose courses*" are presented with its respective semantic annotation. In this model the general semantic annotation has been applied.

In Figure 6.28 a fragment of the service-oriented process model was shown. This fragment exported to iStarML file in Figure 6.35 is shown. In this model the OntoSem ontology has been applied. The actor "*Student Control Department*" and the process elements "*Obtain information about registration*", "*Register course, schedule, professor*" and "*Request support to vigilance* *agent*" are presented with its respective semantic annotation. In this model the specific semantic annotation was applied.

In Figure 6.29 a fragment of the service-oriented protocol model was shown. This fragment exported to iStarML file in Figure 6.36 is shown. In this model the University Ontology has been applied. The actor "*Finance Department*" and the task elements "*Verify bank payments with receipts*", "*Request bank receipt*"; the goal element "*Manage finances of institution*" and resource element "*List of Finances operation*" are presented with its respective semantic annotation. In this model the general semantic annotation has been applied.


```xml
<istarml version="1.0">
  <diagram name="Strategic Dependency Model" author="Blanca Vazquez" id="iStar-SD">
    <actor id="01" name="Card holder" sannotation="User Affiliate"/>
    <actor id="02" name="Terminal Owner" sannotation="Sales_Promoter"/>
    <actor id="03" name="Data Owner" sannotation="User"/>
    <actor id="04" name="Software manufacturer" sannotation="Developer Webmaster"/>
    <actor id="05" name="Card manufacturer" sannotation="Producer"/>
    <actor id="06" name="Card issuer" sannotation="Sales_Promoter"/>
    <ielement id="300" name="Read/write on card correctly" type="softgoal" sannotation="Quality_Assurance">
        <dependency>
            <depender aref="01"/>
            <dependee aref="02"/>
        </dependency>
    </ielement>
    <ielement id="100" name="Present card for transaction" type="goal" sannotation="Electronic_Transaction Customer_Acquisition">
        <dependecy>
            <depender aref="02"/>
            <dependee aref="01"/>
        </dependecy>
    </ielement>
    <ielement id="400" name="Terminal hardware" type="resource" sannotation="Touchscreen Keyboard Point_of_Sale">
        <dependency>
            <depender aref="02"/>
            <dependee aref="06"/>
        </dependency>
    </ielement>
    <ielement id="101" name="Account be managed" type="goal" sannotation="Account_Payable">
        <dependency>
            <depender aref="01"/>
            <dependee aref="03"/>
        </dependency>
    </ielement>
```


Figure 6.30 The actor "Software Manufacturer" with its semantic annotation from Strategic dependency model

```xml
<istarml version="1.0">
  <diagram name="Strategic Rationale Model" author="Blanca Vazquez" id="iStar-SR">
   <actor id="01" name="Card holder" type="position" sannotation="customer credit-card">
       <boundary>
           <ielement id="200" name="Use the card" type="task" sannotation="pay invest"/>
           <ielement id="201" name="Apply for a Card" type="task" sannotation="apply-for register subscribe-to"/>
           <ielement id="202" name="Pre-store some Money" type="task" sannotation="invest savings-account"/>
           <ielement id="203" name="Buy goods with stored value card" type="task" sannotation="buy invest">
               <ielementLink type="decomposition" value="and">
                   <ielement iref="200"/>
                   <ielement iref="201"/>
                   <ielement iref="202"/>
               </ielementLink>
           </ielement>
           <ielement id="204" name="Buy goods with credit card" type="task" sannotation="buy invest"/>
           <ielement id="100" name="Buy goods with smart card" type="goal" sannotation="buy invest">
               <ielementLink type="means-end">
               <ielement iref="204"/>
               <ielement iref="203"/>
               </ielementLink>
           </ielement>
       </boundary>
   </actor>
   <actor id="02" name="Terminal Owner" type="position" sannotation="bank">
       <boundary>
           <ielement id="206" name="Read/write card" type="task" sannotation="read write actualize"/>
           <ielement id="207" name="Read/write DB" type="task" sannotation="record-text information-obtain actualize"/>
           <ielement id="208" name="Process transaction" type="task" sannotation="negotiate-transaction buy pay">
               <ielementLink type="decomposition" value="and">
                   <ielement iref="206"/>
                   <ielement iref="207"/>
               </ielementLink>
           </ielement>
```

Figure 6.31 The task “Use the card” with its semantic annotation from Strategic rationale model

```xml
<istarml version="1.0">
    <diagram name="Actor Model" author="Blanca Vazquez" id="Tropos-Actor">
        <actor id="01" name="Farmer" sannotation="Farmer"/>
        <actor id="02" name="DA" sannotation="central_Office"/>
        <actor id="03" name="Farm supplier" sannotation="supplier"/>
        <actor id="04" name="Credit agent" sannotation="agent"/>
        <actor id="05" name="Regional office" sannotation="central_Office regulation"/>
        <!--    dependency -->
        <ielement id="501" name="Credit money" type="resource" sannotation="credit financing">
            <dependency>
                <depender aref="01"/>
                <dependee aref="04"/>
            </dependency>
        </ielement>
        <ielement id="502" name="Credit service" type="goal" sannotation="credit financing">
            <dependency>
                <depender aref="01"/>
                <dependee aref="04"/>
            </dependency>
        </ielement>
        <ielement id="503" name="Increase productivity" type="goal" sannotation="productive improve_crop">
            <dependency>
                <depender aref="02"/>
                <dependee aref="01"/>
            </dependency>
        </ielement>
        <ielement id="504" name="Logistics arrangement" type="goal" sannotation="schedule_input logistic schedule_crops">
            <dependency>
                <depender aref="01"/>
                <dependee aref="02"/>
            </dependency>
        </ielement>
```

Figure 6.32 The resource “Credit money” with its semantic annotation from actor diagram

```xml
<?xml version="1.0"?>
<istarml version="1.0">
    <diagram name="GRLGraph2">
        <actor id="AA02" name="Farmer" sannotation="farmer">
            <graphic content="basic" xpos="80" ypos="242" width="2060" height="1139"/>
            <boundary>
                <ielement id="AA01" name="Select credit option" type="goal" sannotation="credit financing counseling">
                    <graphic content="basic" xpos="55" ypos="514" width="0" height="0"/>
                </ielement>
                <ielement id="AA03" name="Get credit info" type="goal" sannotation="counseling">
                    <graphic content="basic" xpos="189" ypos="447" width="0" height="0"/>
                </ielement>
                <ielement id="AA11" name="Get credit plan advice" type="goal" sannotation="counseling">
                    <graphic content="basic" xpos="70" ypos="321" width="0" height="0"/>
                    <ielementLink type="decomposition" iref="AA01" aref="AA02" value="and"/>
                    <ielementLink type="decomposition" iref="AA03" aref="AA02" value="and"/>
                </ielement>
                <ielement id="AA04" name="Self-completed" type="plan" sannotation="close">
                    <graphic content="basic" xpos="213" ypos="648" width="0" height="0"/>
                </ielement>
                <ielement id="AA05" name="Request for Assistance" type="plan" sannotation="request-info consult">
                    <graphic content="basic" xpos="392" ypos="645" width="0" height="0"/>
                </ielement>
                <ielement id="AA10" name="Submit credit application" type="goal" sannotation="letter-of-credit application-form">
                    <graphic content="basic" xpos="313" ypos="439" width="0" height="0"/>
                    <ielementLink type="decomposition" iref="AA04" aref="AA02" value="and"/>
                    <ielementLink type="decomposition" iref="AA05" aref="AA02" value="and"/>
                </ielement>
                <ielement id="AA06" name="Rent equipments" type="plan" sannotation="rent">
                    <graphic content="basic" xpos="538" ypos="649" width="0" height="0"/>
                </ielement>
                <ielement id="AA07" name="Buy equipments" type="plan" sannotation="buy invest">
                    <graphic content="basic" xpos="687" ypos="653" width="0" height="0"/>
```


Figure 6.33 The plan "Request for assistance" with its semantic annotation from goal diagram


```xml
<istarml version="1.0">
    <diagram name="Case Study global model">
    <actor id="01" name="Bank" sannotation="Payment">
        <boundary>
            <ielement id="100" name="payment of services" type="service" sannotation="Payment"/>
        </boundary>
    </actor>
    <actor id="02" name="Finance Department" sannotation="Department">
        <boundary>
            <ielement id="101" name="Official receipt generation" type="service" sannotation="Payment Enrollment"/>
        </boundary>
    </actor>
    <actor id="03" name="Finance system" sannotation="Administrator Department">
        <boundary>
            <ielement id="102" name="Financial management" type="service" sannotation="Enrollment"/>
        </boundary>
    </actor>
    <actor id="04" name="Vigilance agent" sannotation="Assistant">
        <boundary>
            <ielement id="103" name="register student entrance" type="service" sannotation="TuitionStudent"/>
        </boundary>
    </actor>
    <actor id="05" name="Student" sannotation="Student MastersLevel DoctoralLevel"/>
    <actor id="06" name="Thesis advisor" sannotation="Professor FullTimeInstructor MasterThesisAdvisor">
        <boundary>
            <ielement id="104" name="Analyze courses" type="service" sannotation="StudyProgram ClassroomCourseProgram"/>
            <ielement id="105" name="Authorize shedule" type="service" sannotation="Assignment ThesisProgram"/>
            <ielement id="106" name="Propose courses" type="service" sannotation="StudyProgram ClassroomCourseProgram CourseOffering"/>
        </boundary>
    </actor>
    <actor id="07" name="Student control department" sannotation="Department">
        <boundary>
            <ielement id="108" name="Register students" type="service" sannotation="TuitionStudent"/>
```


Figure 6.34 The service "Propose courses" with its semantic annotation from global model

```xml
<istarml version="1.0">
  <diagram name="Case Study process model">
    <actor id="01" name="Student Control Department" sannotation="office department-academic">
      <boundary>
        <ielement id="100" name="Publish information about registration" type="goal" sannotation="indicate publish"/>
        <ielement id="300" name="Obtain information about registration" type="process" sannotation="information-obtain determine define">
          <ielementLink type="process-goal_relationship">
            <ielement iref="100"/>
          </ielementLink>
          <ielementLink type="process_relationship">
            <ielement iref="301"/>
          </ielementLink>
        </ielement>
        <ielement id="101" name="Capture course information" type="goal" sannotation="determine write type"/>
        <ielement id="301" name="Register course, schedule, professor" type="process" sannotation="register record-text">
        <ielement id="302" name="Obtain final list of courses" type="process" sannotation="information-obtain determine define">
          <ielementLink type="process_relationship">
            <ielement iref="301"/>
          </ielementLink>
          <ielementLink type="process-goal_relationship">
            <ielement iref="101"/>
          </ielementLink>
        </ielement>
        </ielement>
        <ielement id="102" name="Control the registration" type="goal" sannotation="register record-text annotate"/>
        <ielement id="303" name="Request support to vigilance agent" type="process" sannotation="customer-service help">
          <ielementLink type="process-goal_relationship">
            <ielement iref="102"/>
          </ielementLink>
          <ielementLink type="process_relationship">
            <ielement iref="302"/>
          </ielementLink>
        </ielement>
```


Figure 6.35 The process “Obtain information about registration” with its semantic annotation from process model


```xml
<istarml version="1.0">
  <diagram name="Case Study CENIDET">
    <actor id="02" name="Finance Department" sannotation="Department">
      <boundary>
        <ielement id="55" name="Verify bank payments with receipts" type="task" sannotation="Receipt Management"/>
        <ielement id="56" name="Request bank receipt" type="task" sannotation="Receipt Management"/>
        <ielement id="57" name="Capture data in System" type="task" sannotation="Management Enrollment"/>
        <ielement id="58" name="Deliver official receipt" type="task" sannotation="Receipt Management"/>
        <ielement id="59" name="Manage Students payments" type="task" sannotation="Management">
          <ielementLink type="decomposition" value="and">
            <ielement iref="55"/>
            <ielement iref="56"/>
            <ielement iref="57"/>
            <ielement iref="58"/>
          </ielementLink>
        </ielement>
        <ielement id="60" name="Manage finances of institution" type="goal" sannotation="Management">
          <ielementLink type="means-end">
            <ielement iref="59"/>
          </ielementLink>
        </ielement>
      </boundary>
    </actor>
        <ielement id="500" name="List of Finances operation" type="resource" sannotation="Receipt">
          <dependency>
            <depender iref="55" aref="02"/>
            <dependee iref="50" aref="01"/>
          </dependency>
        </ielement>
    <actor id="03" name="Student" sannotation="Student MastersLevel DoctoralLevel">
      <boundary>
        <ielement id="61" name="Deliver turn" type="task" sannotation="Enrollment"/>
```


Figure 6.36 The task “Manage Students payment” with its semantic annotation from protocol model

6.3.3 Step 3: Integrating the organizational model ontology with a domain ontology

This third step consists of integrating the organizational model with a domain ontology. The input of this step is an annotated organizational model represented in an iStarML file and the domain ontology. The output is the organizational model ontology integrated with a domain ontology represented in an OWL file and the documentation of this integration represented in a text file.

In this section, the integration of each organizational model with the domain ontology is presented. In order to carry out this integration is used TAGOOn+ tool proposed in this research. Each element of the model is integrated with one or more domain concepts by means of links "*is a*".

Protégé 4.1 tool [83] was used to open the OWL file generated by TAGOOn+. The file generated is opened with Protégé and is shown the links "*is a*" among domain concepts and the elements of the model. In order to validate this integration a fragment of the documentation is visualized. In each section a table describes the number of suggestions applied for organizational model.

6.3.3.1 i* Strategic dependency model

In Figure 6.37 in the top-left the goal element "*Presented Card for Transaction*" is shown. This element was annotated with domain concepts *"@identify, @authenticate, @negotiate-transaction*" from OntoSem ontology. When all i* strategic dependency model is exported, this element represented in iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element.

Then TAGOOn+ has been executed. A short view of this file opened with Protégé in the center left is shown. The object property assertions of individual "*Presented card for transaction*" are: "*is a authenticate*", "*is a identify*" and "*is a negotiate-transaction*". This object property indicates that this element have relationships of type "*is a"* with these concepts. The XML tag "*sannotation*" from iStarML file is transformed to data property assertion "*Node_sannotation*" with the same values. Each space the tag "*sannotation*" is presented with the symbol "&". Moreover, graphically this integration is shown in the center graph.

The individual "*Presented card for transaction*" is integrated with domain concepts "*authenticate*", "*negotiate-transaction*" and "*identify*". The specific semantic annotation suggestion for goal elements indicates that "*ME:* ***Goal*** $\xrightarrow{AB}$ *OC:mental-event v OC:social-event v OC:mental:object*". In this way, the Figure 6.37the domain concepts "*authenticate*" and "*negotiate-transaction*" belong to "*social-event*" superconcept and the "*identify*" belongs to "*mental-event*" are shown. Therefore the integration of this element has followed the suggestion and is correct. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual "*Presented card for transaction*" is a dependum of type "*goal*" and the three domain concepts and the description of each one of them is shown.

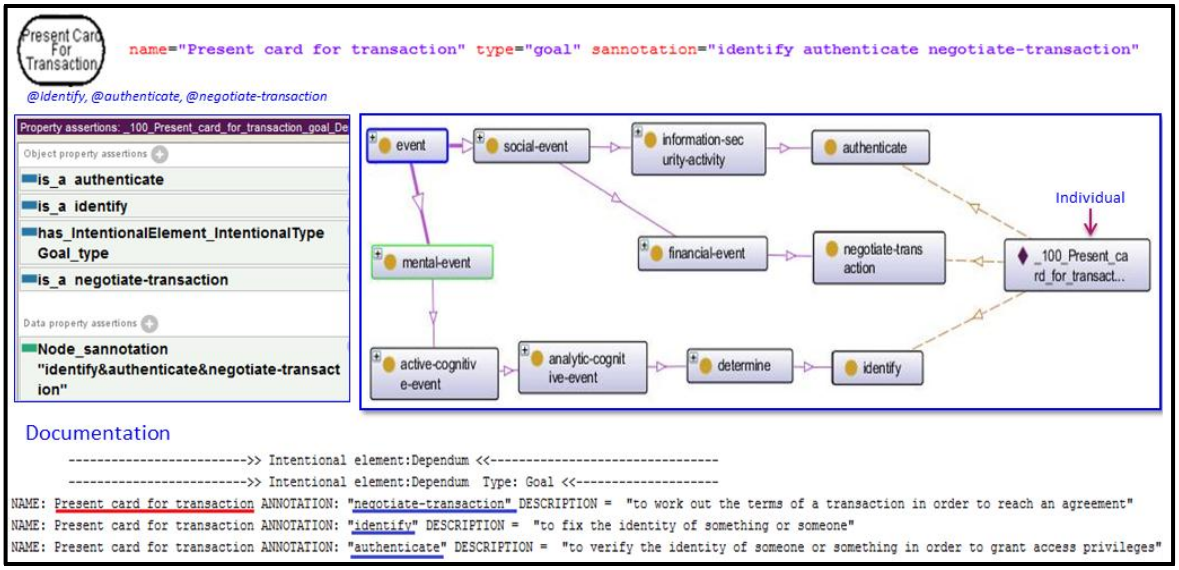


Figure 6.37 i* Strategic dependency model integrated with OntoSem ontology

In Table 6-1 the numbers of relationships to i* Strategic dependency model applying the specific semantic annotation suggestions are shown. SSAS means "*Specific Semantic Annotation Suggestions*".

Table 6-1 Created relationships for the i* strategic dependency model

Specific semantic suggestion	Number of relationships "is a"	Description
SSAS_1	11	11 relationships between "*object*" superconcept and "Actor" node
SSAS_2	8	8 relationships among "*mental-event*", "*social-event*", "*mental-object*" superconcept and the element type "*goal*".
SSAS_3	11	11 relationship between "*abstract-object*" superconcept and the element type "*softgoal*"
SSAS_4	4	4 relationships among "*active-cognitive-event*", "*social-event*" superconcepts and the element type "*task*".
SSAS_6	15	15 relationships among "*physical-object*", "*mental-object*" superconcept and the element type "*resource*".

6.3.3.2 i* Strategic rationale model

In Figure 6.38 in the top-left the goal element "*Create new account*" is shown. This element was annotated with domain concepts *"@open-account @bank-account"* from OntoSem ontology. When all i* strategic dependency model is exported, this element represented in iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element.

Then TAGOOn+ has been executed. A short view of this file opened with Protégé in the center left is shown. The object property assertions of individual "*Create new account*" are: "*is a open-account*" and "*is a bank-account*". This object property indicates that this element have relationships of type “*is a”* with these concepts. Moreover, graphically this integration is shown in the center graph. The individual “*Create new account*” is integrating with domain concepts “*open-account*” and “*bank-account*”. The specific semantic annotation suggestion for goal elements indicates that “*ME:* ***Goal*** $\xrightarrow{AB}$ *OC:mental-event v OC:social-event v OC:mental:object*”.

In the Figure 6.38 domain concept “*open-account*” belongs to “*social-event*” superconcept and the “*bank-account*” belong to “*mental-object*” is shown. Therefore the integration of this element following the suggestion is correct. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual “*Create new account*” is an internal element of type “*goal*”, and the two domain concepts and the description of each one of them is shown.

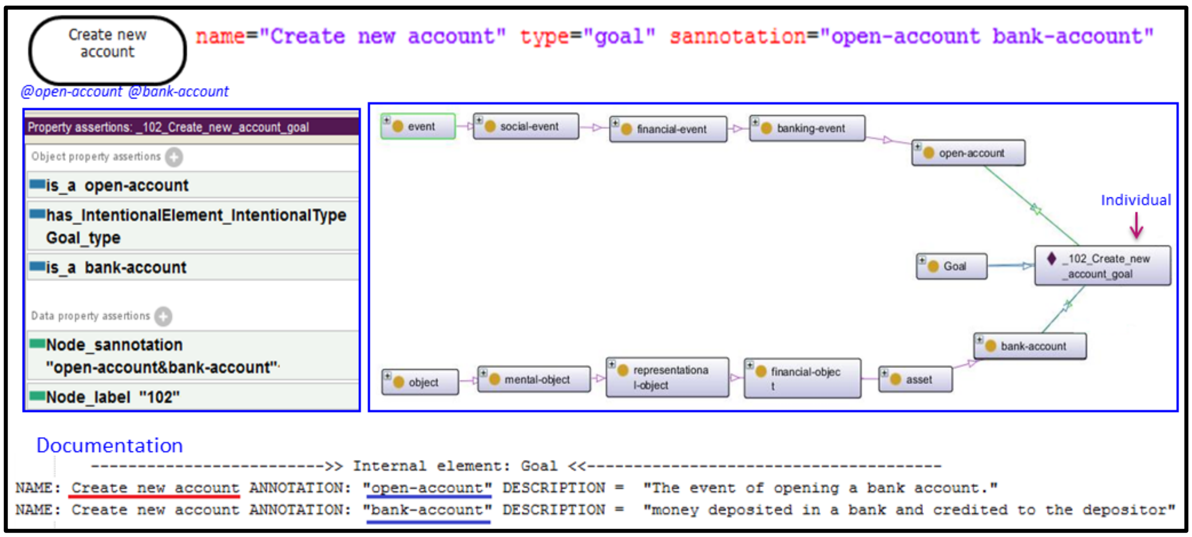


Figure 6.38 i* Strategic rationale model integrated with OntoSem ontology

In Table 6-2 the numbers of relationships to i* Strategic rationale model applying the specific semantic annotation suggestions are shown.

Table 6-2 Created relationships for the i* strategic rationale model

Specific semantic suggestion	Number of relationships “is a”	Description
SSAS_1	11	11 relationships between “object” superconcept and “*Actor*” node
SSAS_2	24	24 relationships among “*mental-event*”, “social-event”, “*mental-object*” superconcept and the element type “*goal*”.
SSAS_3	16	16 relationship between “*abstract-object*” superconcept and the element type “softgoal”
SSAS_4	69	69 relationships among “*active-cognitive-event*”, “*social-event*” superconcepts and the element type “*task*”.
SSAS_6	26	26 relationships among “*physical-object*”, “*mental-object*” superconcept and the element type “*resource*”.

6.3.3.3 Tropos actor diagram

In Figure 6.39 in the top-left the actor element "*Regional office*" is shown. This element was annotated with domain concepts *"@office @inspection-organization"* from OntoSem ontology. When all i* strategic dependency model is exported, this element is represented in iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element. Then TAGOOn+ has been executed.

A short view of this file opened with Protégé in the center left is shown. The object property assertions of individual "*Get information*" are: "*is a inspection-organization*" and "*is a office*". This object property indicates that this element have relationships of type "*is a"* with these concepts. Moreover, graphically this integration in the center graph is shown.

The individual "*Regional office*" is integrating with domain concepts "*inspection-organization*" and "*office*". The specific semantic annotation suggestion for goal elements indicates that "*ME:* ***Actor*** $\xrightarrow{AB}$ *OC:object*". In this way, the Figure 6.39 that the domain concepts "*office*" and "*inspection-organization*" belong to "*object*" superconcept are shown. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual "*Regional office*" is a node "*Actor*" and let see the two domain concepts and the description of each one of them.

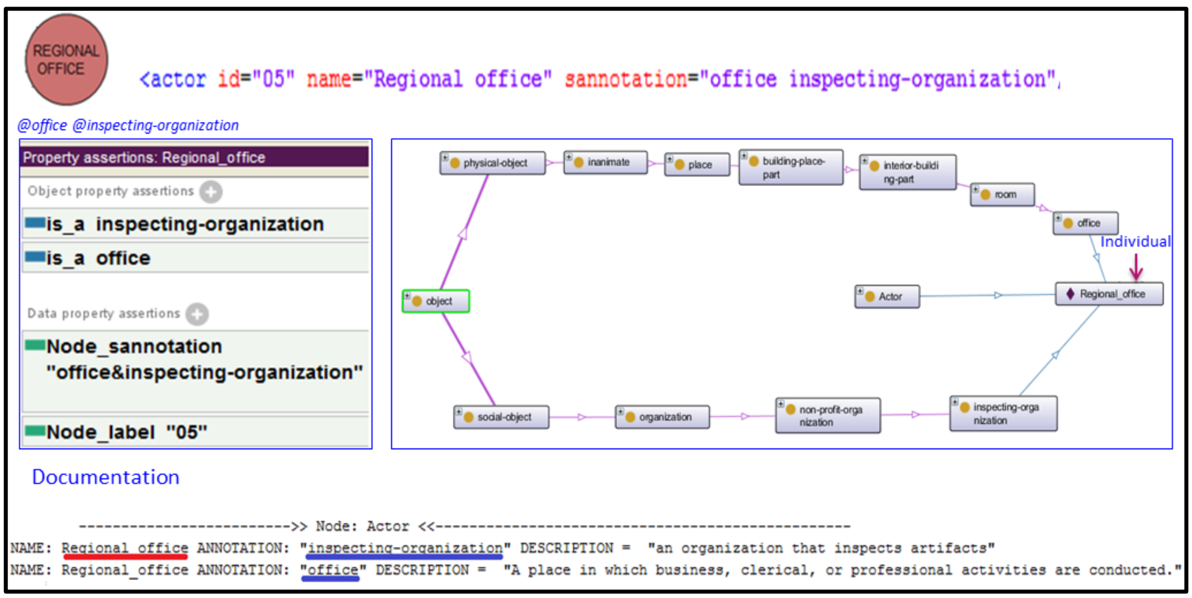


Figure 6.39 Tropos actor diagram integrated with OntoSem ontology

In Table 6-3 the numbers of relationships to Tropos actor diagram applying the specific semantic annotation suggestions are shown.

Table 6-3 Created Relationships for the Tropos actor diagram

Specific semantic suggestion	Number of relationships "is a"	Description
SSAS_1	9	9 relationships between "*object*" superconcept and "*Actor*" node
SSAS_2	13	13 relationships among *"mental-event", "social-event", "mental-object"* superconcept and the element type "goal".
SSAS_3	2	2 relationship between "*abstract-object*" superconcept and the element type "softgoal"
SSAS_5	4	4 relationships among *"active-cognitive-event", "social-event"* superconcepts and the element type "*plan*".
SSAS_6	12	12 relationships among "*physical-object*", "*mental-object"* superconcept and the element type "*resource*".

6.3.3.4 Tropos goal diagram

In Figure 6.40 in the top-left the softgoal element "*Timeliness*" is shown. This element was annotated with domain concept *"@opportunity"* from OntoSem ontology. When all i* strategic dependency model is exported, this element is represented in iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element. Then TAGOOn+ has been executed. A short view of this file opened with Protégé is shown in the center left. The object property assertions of individual "Timeliness" is: "*is a opportunity*". This object property indicates that this element have relationships of type "*is a"* with this concept. Moreover, graphically this integration in the center graph is shown.

The individual "*Timeliness*" is integrating with domain concept "*opportunity*". The specific semantic annotation suggestion for softgoal elements indicates that "*ME:* ***Softgoal***$\xrightarrow{AB}$ *OC:abstract-object*". In this way, the Figure 6.40 that the domain concept "*opportunity*" belongs to "*abstract-object*" superconcept is shown. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual "*Timeliness*" is an internal element of type "*Softgoal*" and the domain concept and the description are shown.

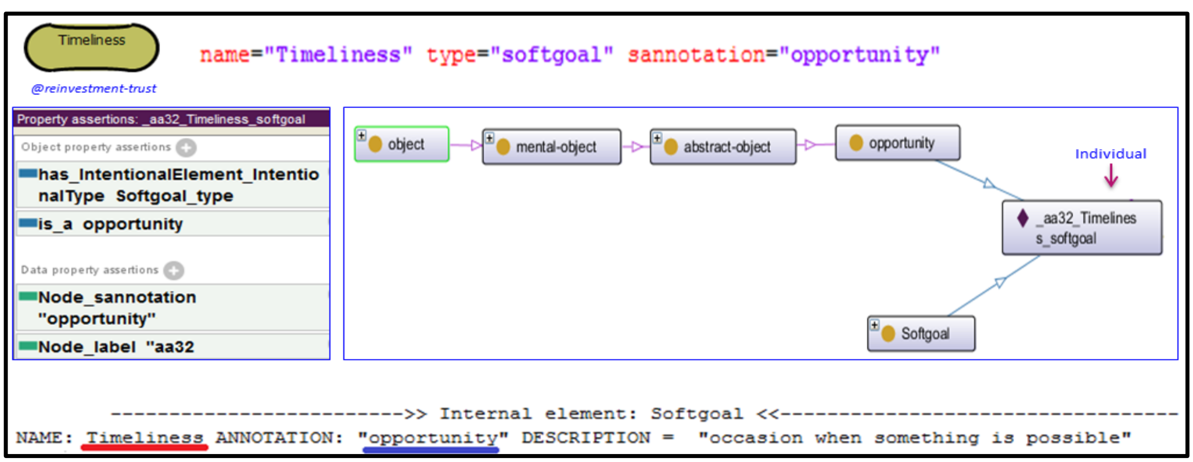


Figure 6.40 Tropos goal diagram integrated with OntoSem ontology

In Table 6-4 the numbers of relationships to Tropos goal diagram applying the specific semantic annotation suggestions are shown.

Table 6-4 Created relationships for the Tropos goal diagram

Specific semantic suggestion	Number of relationships "is a"	Description
SSAS_1	3	3 relationships between "object" superconcept and "Actor" node
SSAS_2	73	73 relationships among "mental-event", "social-event", "mental-object" superconcept and the element type "goal".
SSAS_3	11	11 relationship between "abstract-object" superconcept and the element type "softgoal"
SSAS_5	18	18 relationships among "active-cognitive-event", "social-event" superconcepts and the element type "plan".
SSAS_6	2	2 relationships among "physical-object", "mental-object" superconcept and the element type "resource".

6.3.3.5 Service-oriented global model

In Figure 6.41 in the top-left the service element "*Financial Management*" is shown. This element was annotated with domain concepts *"@tax-management @coordinate"* from OntoSem ontology. When all i* strategic dependency model is exported, this element is represented in iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element. Then TAGOOn+ has been executed. A short view of this file opened with Protégé in the center left is shown.

The object property assertions of individual "*Financial Management*" are: "*is a tax-management*" and "*is a coordinate*". This object property indicates that this element have relationships of type "*is a"* with these concepts. Moreover, graphically this integration in the center graph is shown. The individual "Financial Management" is integrating with domain concepts "*tax-management*" and "*coordinate*". The specific semantic annotation suggestion for service elements indicates that "*ME:* ***Service*** $\xrightarrow{AB}$ *OC:social-event*".

In this way, the Figure 6.41 the domain concepts "*tax-management*" and "*coordinate*" belong to "*social-event*" superconcept are shown. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual "Financial Management" is an internal element of type "*Plan*" and let see the two domain concepts and the description of each one of them.

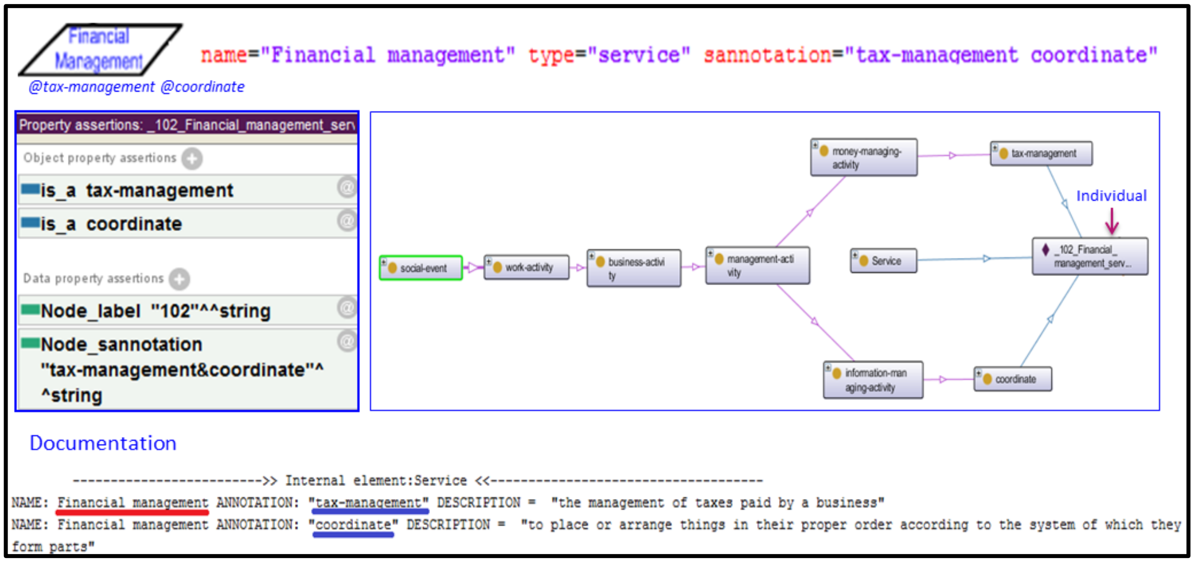


Figure 6.41 Service-oriented global model integrated with OntoSem ontology

In Table 6-5 the numbers of relationships to service-oriented global model applying the specific semantic annotation suggestions are shown.

Table 6-5 Created relationships for the Service-oriented global model

Specific semantic suggestion	Number of relationships "is a"	Description
SSAS_1	23	23 relationships between "*object*" superconcept and "*Actor*" node
SSAS_2	54	54 relationships among "*mental-event*", "*social-event*", "*mental-object*" superconcept and the element type "*goal*".
SSAS_7	26	26 relationships among "*social-event*" superconcept and the element type "*service*".

6.3.3.6 Service-oriented process model

In Figure 6.42 in the top-left the process element "*Request control number*" is shown. This element was annotated with domain concepts *"@information-obtain @identify"* from OntoSem ontology. When all i* strategic dependency model is exported, this element is represented in iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element. Then TAGOOn+ has been executed.

A short view of this file opened with Protégé is shown in the center left. The object property assertions of individual "*Request control number*" are: "*is a identify*" and "*is a information-obtain*". This object property indicates that this element have relationships of type "*is a*" with these concepts. Moreover, graphically this integration in the center graph is shown. The individual "*Request control number*" is integrating with domain concepts "*information-obtain*" and "*identify*". The specific semantic annotation suggestion for process elements indicates that "*ME: **Process*** $\xrightarrow{AB}$ *OC:active-cognitive-event v OC:social-event*". In this way, the Figure 6.42 the domain concept "*information-obtain*" belongs to "*social-event*" superconcept and "*identify*" belongs to "*active-cognitive-event*" superconcept are shown. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual "*Request control number*" is an internal element of type "*Process*" and let see the two domain concepts and the description of each one of them.

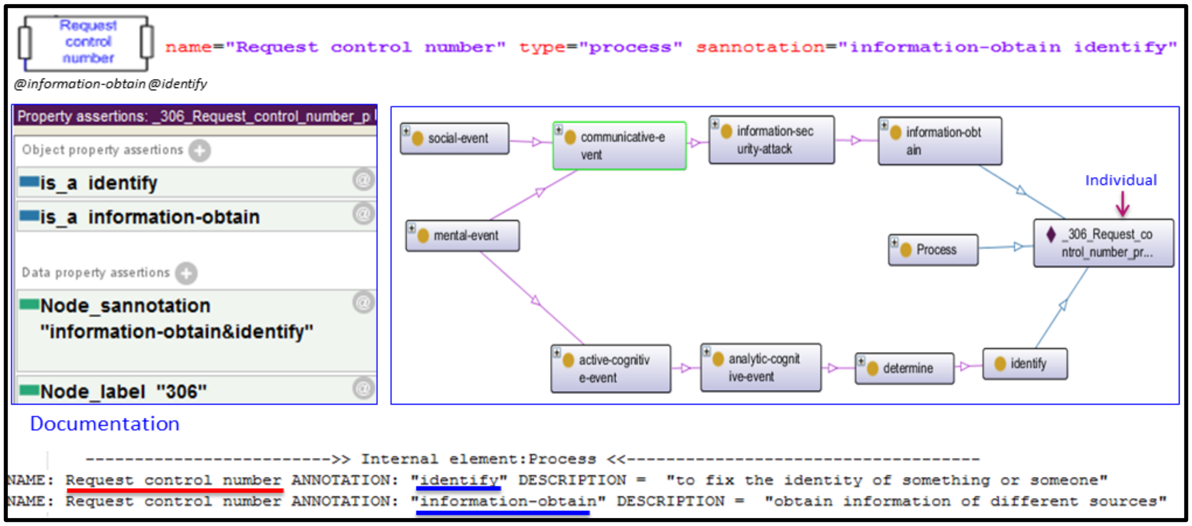


Figure 6.42 Service-oriented process model integrated with OntoSem ontology

In the Table 6-6 the numbers of relationships to service-oriented process model applying the specific semantic annotation suggestions are shown.

Table 6-6 Created relationships for the Service-oriented process model

Specific semantic suggestion	Number of relationships "is a"	Description
SSAS_1	4	4 relationships between "*object*" superconcept and "Actor" node
SSAS_2	42	42 relationships among "*mental-event*", "*social-event*", "*mental-object*" superconcept and the element type "goal".
SSAS_7	2	2 relationships among "*social-event*" superconcept and the element type "*service*".
SSAS_8	25	25 relationships among "*active-cognitive-event*", "*social-event*" superconcepts and the element type "process".

6.3.3.7 Service-oriented protocol model

In Figure 6.43 in the top-left the process element "*Print official receipt*" is shown. This element was annotated with domain concepts *"@print-from-computer @file-document"* from OntoSem ontology. When all i* strategic dependency model is exported, this element is represented iStarML in the top-center is shown. Let see the tag "*name*", "*type*" and "*sannotation*". This last tag stored the domain concepts for this element. Then TAGOOn+ has been executed.

A short view of this file opened with Protégé is shown in the center left. The object property assertions of individual "*Print official receipt*" are: "*is a print-from-computer*" and "*is a file-*

document". This object property indicates that this element have relationships of type "*is a*" with these concepts. Moreover, graphically this integration in the center graph is shown. The individual "*Print official receipt*" is integrating with domain concepts "*print-from-computer*" and "*file-document*". The specific semantic annotation suggestion for task elements indicates that "$\textit{ME:}\ \textbf{\textit{Task}} \xrightarrow{AB} \textit{OC:active-cognitive-event}\ v\ \textit{OC:social-event}$".

In this way, the Figure 6.43the domain concept "*print-from-computer*" and "*file-document*" belongs to "*social-event*" superconcept are shown. Finally, a fragment of the documentation generated by TAGOOn+ in the bottom-center is shown. The individual "*Print official receipt*" is an internal element of type "*Task*" and let see the two domain concepts and the description of each one of them.

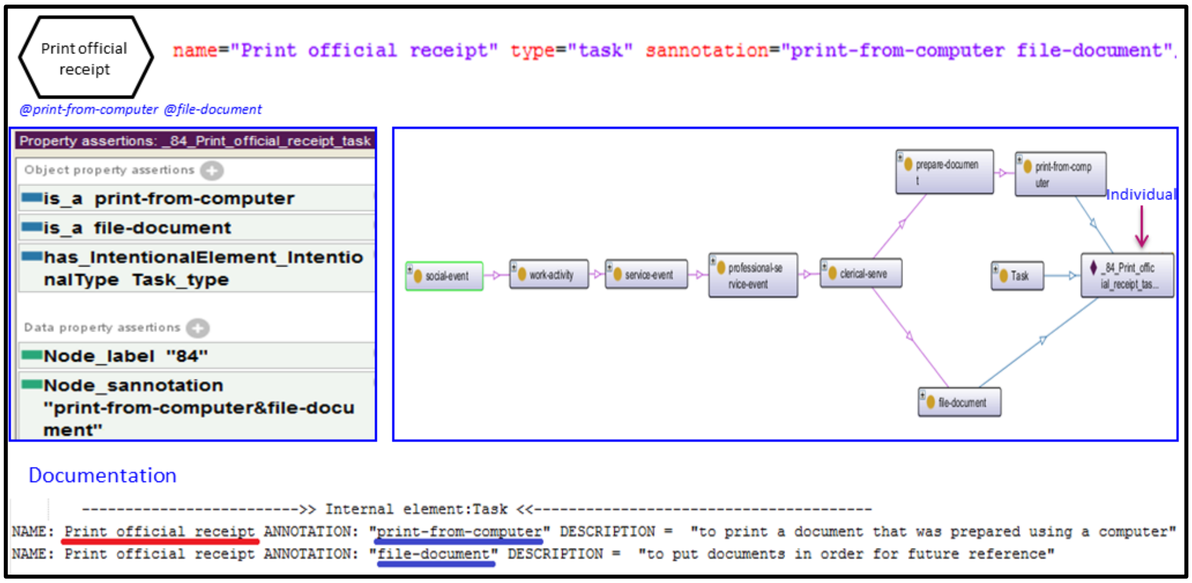


Figure 6.43 Service-oriented protocol model integrated with OntoSem ontology

In Table 6-7 the numbers of relationships to service-oriented process model applying the specific semantic annotation suggestions are shown.

Table 6-7 Created relationships for the Service-oriented protocol model

Specific semantic suggestion	Number of relationships "is a"	Description
SSAS_1	23	23 relationships between "*object*" superconcept and "*Actor*" node
SSAS_2	63	63 relationships among "mental-event", "*social-event*", "*mental-object*" superconcept and the element type "*goal*".
SSAS_4	274	274 relationships among "*active-cognitive-event*", "*social-event*" superconcepts and the element type "*task*".
SSAS_6	47	47 relationships among "*physical-object*", "*mental-object*" superconcept and the element type "*resource*".

In Table 6-8 presents a summary of the general semantic annotation suggestions. The first column the name each one organizational model is presented, the second columns the name of the domain ontology applied is presented, the third column the number of domain concepts of each domain ontology is described, the fourth column the total number of semantic annotation used is described and the last column the number of relationships "*is a*" between model elements and domain concepts is presented.

Table 6-8 Summary table of general semantic annotation suggestions

Model	Domain ontology	Domain concepts	Annotation	Relationships between model elements and domain concepts
i* Strategic dependency	Ife_2007 ontology	320	43	43
i* Strategic rationale	Ife_2007 ontology	320	102	102
Tropos actor diagram	FarmOntology	105	31	31
Tropos goal diagram	FarmOntology	105	127	127
Test05_Global model	UniversityOntology	100	93	93
Test06_Process model	UniversityOntology	100	50	50
Test07_Protocol model	UniversityOntology	100	251	251

6.4 Summary

In this chapter has been presented the testing of the guideline proposed in this research. The guideline were testing with seven organizational models represented in the variants i*, Tropos and Service-oriented i*. Three different ontologies obtained of the Web were used to annotate all models for the case study using the general semantic annotation suggestions. Then each model was exported to iStarML interchange format.

The XML attribute "*sannotation*" is generated for the extension of plug-in for JUCMNav. This file is the input of TAGOON+ and the OWL file generated by TAGOOn+ is open with Protégé tool. All the model elements annotated were integrated with domain concepts.

Moreover, to test the specific semantic annotation suggestions the same models were annotated with domain concepts from OntoSem ontology and exported to iStarML files. Each model was integrated with the general ontology without errors and redundancy.

A fragment of each model integrated with the domain ontology was shown graphically. The relationship "*is a*" between domain concepts and the elements annotated using the specific semantic annotation suggestions was presented. Finally a fragment of documentation generated by TAGOOn+ was shown too.

This documentation is useful for the technical peoples because allow a better understanding of the organizational modeled. The process of enriching models using ontologies is better with general ontologies because the extension and the diversity of concepts existents, however the process annotation with domain ontologies is useful due to that the ontology is applied a domain specific. The proposed the enrichment models is to clarify the hidden semantic in the model elements. Moreover, the enrichment of model with domain concepts allows us to group model elements according to similar situations and description.

In this way, the concepts domain permits the standardization of elements by means of common denominators.

Chapter 7 Conclusions and future work

7.1 Conclusions

In this section the conclusions, the summary of the contributions and the future works of our research work will be presented.

The main objective of our thesis is: "E*nrich the organizational models with annotations, characterized by a semantics explicitly organized in a structured source of knowledge. This annotation provides a precise and clear formal meaning to the elements of model*".

The advantages of enriching of organization visual models with semantic annotation are:

- ✓ The semantic annotations provide a precise meaning to elements of the model.
- ✓ The annotations can improve performance the performance of the labeling activity and also, avoid inconsistency.
- ✓ The propose approach permits to improve the process of analysis of visual models and organizational knowledge.
- ✓ The explicit representation of meanings of elements permits the reuse of information.
- ✓ The annotation could resolve the ambiguity of natural language description.
- ✓ The approach enables the analyst to support the discovering and implementation of services by means of domain concepts.

For the accomplishment of the main objective, a set of specific objectives were identified. These objectives are described below as well as the activities carried out for their achievement.

1. "*The development of an approach for building of ontologies integrated with an organizational model ontology*". For the accomplishment of this objective, our approach developed consisted of two phases. The first phase focused on representing an annotated model into iStarML format, which required two processes. Process 1 "*Semantic annotation suggestion development*" consisted of developing a set of generals and specifics semantic annotation suggestions to guide the annotation process.

   Process 2 "*Extension of iStarML*" which consisted of representing the annotated model into iStarML format. The second phase consisted of developing a tool to integrate an annotated model represented in iStarML with a domain ontology. A case study was presented to validate our approach.

2. "*The development of the semantic annotation suggestions to annotate the organizational model using the domain concepts*". For developing the suggestions, three steps were carried out. Step 1 The primitives of the models were analyzed and compared among them. The result was to obtain a single definition for each one of the primitives. Step 2 Several general and domain ontologies were analyzed. This analysis consisted of exploring the hierarchy and the relationships between domain concepts. The result was to establish relationships between the definition of each primitive (Step 1) towards one or more domain concepts. Step 3 Each primitive was established formally with one or more domain concepts.

   The result of this step was a set of general semantic annotation suggestions and a set of specific semantic annotation suggestions. The first suggestions are applied to any domain ontologies. The second suggestions are applied to OntoSem ontology and its extensions. On the one hand, the general suggestions have certain freedom to relate each primitive with domain concepts. For example, the primitive "*goal*" should be mapped into domain concepts that describe a clear and precise condition, interest or desire. On the other hand, the specific semantic annotation suggestions present the relationships of each primitive into one or more domain concepts of OntoSem. For example, the primitive "*goal*" should be mapped into one of the concepts "*mental-event, social-event and mental-object*". This means that all the instances of a primitive of type goal should map to its concepts, independently of the model´s domain.

3. "*The extensions of an existing plug-in to export an iStarML file adding the semantic annotation for each element of the model*". For the accomplishment of this objective, we automated the generation of a model annotated represented in iStarML format by extending the JUCMNav tool. The "*import-export plug-in*" for JUCMNav exports and imports goal models into the iStarML format.

   We extended the graphical editor JUCMNav to add the semantic annotation. In particular, the "*description*"" property is used, together with a demarking symbol "@". We extended the plug-in also to generate the new iStarML file adding the semantic annotation using the new attribute "*sannotation*", resulting in an automatic generation of iStarML files with model annotations.

4. "*The application of the approach to the variants: i*, Tropos and Service-oriented i* integrating towards a domain ontology. This, by the extension of TAGOOn (Tool for the Automatic Generation of Organizational Ontologies) in order to demonstrate the effectiveness of the guideline*". For the accomplishment of this objective we proposed the tool called "*Tool for the Automatic Generation of Organizational Ontologies and*

Integration" (TAGOOn+). The input of this tool is the annotated model represented in iStarML and the validated domain ontology. TAGOOn+ is the extension of the tool generated by Najera in her master thesis. The added modules: "*Automatic parsing process*", "*Automatic linking process*" and "*Automatic documentation process*".

The first module parsed the iStarML file and the OWL file to store all the structure of each document. The second module transforms the organizational model into an ontology. Then, each element of the models is linked with one or more domain concepts. Each attribute "*sannotation*" from iStarML file is represented as dataproperty for each individual in the ontology. The links between ontologies are of the type "*is a*".

The third module describes this integration in a text document. The idea is to generalize the elements of the model using generic concepts to improve the labeling activity and to implement services from the generated ontology. The output of our proposed tool is the organizational ontology integrated with a domain ontology and a summary document that visualized each element with its semantic annotation and description.

5. *"Validate the whole semi-automated semantic annotation process on a set of examples"*. For the accomplishment of this objective, we validated our approach taking seven organizational models from three case studies represented in the variants i*, Tropos and Service-oriented i*. The i* models described a generic card-based payment system, the Tropos models described the designing distributed agricultural information services for developing countries and the services oriented models described the process to register students at a postgraduate institution.

   Three domain ontologies were obtained of the web in order to annotate the elements of the models. Each element was annotated following the general semantic suggestions, and then the models were exported into iStarML format and finally were integrated with its domain ontology. In order to validate this integration each OWL file generated by TAGOOn+ was opened using Protégé. A fragment of each element integrated with one or more domain concepts was presented.

   Moreover to test the specific semantic annotation, the models were annotated following the specific semantic suggestions and using OntoSem ontology. Then each model was exported and integrated with this generic ontology. In this way, our approach was testing to validate the enrichment of organizational models taken concepts from domain ontologies.

7.1.1 Summary of contributions

The core of this thesis is the presentation of a guideline that formalizes organizational models enriched with semantic annotation from source of knowledge. The semantic annotations of organizational models, in fact, can be used to provide a precise, formal meaning to model elements, thus making them more understandable to people and allowing further analysis.

Several contributions have been implemented in this thesis:

- ✓ The presentation of an approach to enrich organizational model with annotations characterized by semantics explicitly organized in a structured source of knowledge.
- ✓ The development of a set of general semantic suggestions and a set of specific semantic suggestions are the guidelines to integrate an organizational model to domain ontology.
- ✓ The presentation of an approach to integrate an organizational model ontology into domain ontology.
- ✓ The extension of iStarML interchange format adding the XML attribute "*sannotation*" allows us to store the semantic annotation for each element of the model.
- ✓ The extension of an existing plug-in to automatically generate an annotated model to iStarML format.
- ✓ The extension of an initial organizational model represented with the variants: i*, Tropos and Service-oriented i* with concepts available in a domain ontology.
- ✓ The development of TAGOOn+ allows us to integrate an annotated model represented with the variants: i*, Tropos and Service-oriented i* to a domain ontology.
- ✓ The automatic generation of documentation of an organizational model integrated with a domain ontology.
- ✓ The categorization of the elements of the model into domain concepts allows us the reuse of parts of the elements when creating new models, the detection of cross-item relationships and reasoning between elements.
- ✓ The obtaining of a richer organizational model with a clear semantic with the support of domain ontology.
- ✓ The standardization of elements of the model using domain concepts improves the process of analysis and achieves the reuse of information.

The main contribution of our research is the presentation of a set semantic annotation suggestions to enrich the organizational model. The model annotated with concepts from domain ontologies is useful to improve the labeling activity. The annotations clarify the hidden semantic in the models avoiding the ambiguity and categorize the elements with similar description. Moreover a model annotated with semantic annotations is clear for humans and accessible to machines because to the integration with ontologies. It is important mention that is preserved the knowledge of each label of the model.

The benefits of having semantic information in organizational model would be: on one side, it would facilitate the analysis and understanding of a model proving a clear model supported for domain concepts and, on the other, it would allow the implementation of services through of the integration of elements of the model according to similar situations and description.

7.2 Related publications

Part of the contributions of this thesis is supported by two publications out throughout this research work:

- Blanca Vazquez, Alicia Martinez, Anna Perini, Hugo Estrada, Mirko Morandini. "*Enriching Organizational Models through Semantic Annotation*". In "The 2013 Ibero-American Conference on Electronics Engineering and Computer Science" CIIEEC 2013. (To be published).
- Blanca Vazquez, Alicia Martinez, Anna Perini, Hugo Estrada, Mirko Morandini. "*Integrating organizational model with domain ontology*". In "XVI Ibero-American Conference on Software Engineering" CIbSE2013. (To be published).

7.3 Ongoing and future work

At the present time, we are focusing on enriching the organizational model describing its elements with generic concepts. As future work it is pretended to cover, with natural language processing techniques the annotation of each element of the model. In this way, the automatic suggestions will be provided to the analyst to annotate the model. Other future activity will be the development of a tool to implement the services from the ontology generated by TAGOOn+.

Bibliography

[1] A. Martínez, H. Estrada; A. Gama, “Una guía rápida de la metodología Tropos,” *Revista Gerencia Tecnológica Informática - GTI Colombia : Universidad Industrial de Santander*, vol. 7, no. 19, pp. 67–77, 2008.

[2] E. Yu, “Modelling Strategic Relationships for Process Reengineering,” PhD. thesis, University of Toronto, Department of Computer Science, 1995.

[3] H. Estrada, O. Pastor, “Generación de modelos organizacionales a partir de estructuras de refinamiento de metas,” *Informatics Technology Management Journal*, vol. 4, no. 10, 2005.

[4] H. Estrada Esquivel, “A service-oriented approach for the i* framework,” Ph.D. dissertation, Valencia University of Technology, Valencia, Spain, 2008.

[5] C. Cares, X. Franch, “iStarML: Principles and Implications,” *CEUR Proceedings of the 5th International i* Workshop (iStar 2011)*, pp. 8–13.

[6] E. Yu, P. Giorgini, N. Maiden, J. Mylopoulos, *Social Modeling for Requirements Engineering*, MIT Press, ©2011. Massachusetts Institute of Technology: Cambridge, Mass, 2011.

[7] C. Cares, X. Franch, “A Metamodelling Approach for i* Model Translations,” in *Advanced Information Systems Engineering*, 23rd International. Conference, CAiSE 2011, London, UK., vol. 6741, Lecture Notes in Computer Science, 2011, pp. 337–351.

[8] C. Ayala, C. Cares, J. Carvallo, G. Grau, M. Haya, G. Salazar, X. Franch, E. Mayol, C. Quer, “A Comparative Analysis of i*-Based Agent-Oriented Modeling Languages,” *In Proceedings of 17th International Conference on Software Engineering and Knowledge Engineering (SEKE’05)*, pp. 43–50, 2005.

[9] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, A. Perini, “TROPOS: An Agent-Oriented Software Development Methodology,” in *Autonomous Agents and Multi-Agent Systems*, vol. 8, 2004, pp. 203–236.

[10] D. Amyot, G.Mussbacher, “URN: Towards a New Standard for the Visual Description of Requirements,” *Third International Workshop. Telecommunications and beyond: The Broader Applicability of SDL and MSC.*, pp. 24–26, Aberystwyth, UK 2002.

[11] C. Di Francescomarino, “Semantic annotation of business process models,” PhD Thesis, International Doctorate School in Information and Communication Technologies DISI - University of Trento, 2011.

[12] Y. Lin, D. Strasunskas, S. Hakkarainen, J. Krogstie, A. Sølvberg, “Semantic Annotation Framework to Manage Semantic Heterogeneity of Process Models,” in *Advanced Information Systems Engineering*, Lecture Notes in Computer Science., vol. 4001, 2006, pp. 433–446.

[13] K.Decreus, G. Poels, “Mapping semantically enriched Formal Tropos to Business Process Models,” *SAC ’09 Proceedings of the 2009 ACM symposium on Applied Computing. Honolulu, Hawaii, U.S.A.*, pp. 371–376, 2009.

[14] D. Colomer, “iStarML Import/Export plugin for jUCMNav,” 24-Feb-2011. [Online]. Available: http://www.upc.edu/gessi/istarml/resources.html.

[15] K. Najera, “An ontology-based approach for integrating i* variants,” Master’s thesis, National Center for Research and Technological Development, Cuernavaca, Morelos, Mexico, 2011.

[16] A. Perini, A. Susi, “Developing Tools for Agent-Oriented Visual Modeling,” in *Multiagent System Technologies*, vol. 3187, Lecture Notes in Computer Science, 2004, pp. 169–182.

[17] A. Watson, “Visual Modelling: past, present and future,” *Object Management Group (OMG)*. [Online]. Available: http://www.uml.org/. [Accessed: 15-May-2012].

[18] P. Giorgini, J. Mylopoulos, A. Perini, A Susi, “The Tropos Methodology and Software Development Environment,” in *Social Modeling for Requirements Engineering*, MIT Press, Cambridge MA, 2010, p. Chapter 11.

[19] C. Di Francescomarino, P. Tonella, “Supporting ontology-based semantic annotation of business processes with automated suggestions,” *Lecture Notes in Business Information Processing*, vol. 29, no. Part 2, Part 6, pp. 211–223, 2009.

[20] G. Guizzardi, H. Herre, G. Wagner, “On the General Ontological Foundations of Conceptual Modeling,” in *Proceedings of 21th International Conference on Conceptual Modeling ER-2002*, Springer-Verlag, Berlin, Lecture Notes in Computer Science., 2002, pp. 65–78.

[21] C. C. X. Franch, “Towards a framework for Improving Goal-Oriented Requirement Models Quality,” *Proc. of the 12th Workshop on Requirements Engineering*, pp. 3–14, 2009.

[22] “Tropos web page.” [Online]. Available: http://www.troposproject.org/. [Accessed: 11-Nov-2011].

[23] T. Gruber, “Towards Principles for the Design of Ontologies Used for Knowledge Sharing,” in *Formal Ontology in Conceptual Analysis and Knowledge Representation*, vol. 43, 1993, pp. 625–640.

[24] D. Oberle, “Semantic Management of Middleware,” in *Semantic Web And Beyond Computing for Human Experience*, 2006, vol. 1.

[25] D. Sánchez,J. Cavero, E. Marcos, “Ontologías y MDA: una revisión de la literatura,” *CEUR Workshop Proceedings - II Workshop on Model-Driven Development. MDA and Applications*, 2005.

[26] N. Guarino, “Formal Ontology and Information Systems,” in *Formal Ontology and Information Systems. Proceedings of FOIS’98 Trento Italia*, Amsterdam, IOS Press, 1998, pp. 3–15.

[27] S. Borgo, “WonderWeb: Ontology Infrastructure for the Semantic Web,” 2004.

[28] Ontotext AD, “Semantic Annotation,” *ontotext*, 2011. [Online]. Available: http://www.ontotext.com/kim/semantic-annotation.

[29] T. Berners-Lee, J. Hendler, O. Lassila, “La Red semántica,” *Scientific American*, vol. 298, pp. 38–47, 2001.

[30] M. Grčar, E. Klien, B. Novak, “Using Term-Matching Algorithms for the Annotation of Geo-services,” *In Knowledge Discovery Enhanced with Semantic and Social Information*, vol. 220, pp. 127–143, 2009.

[31] V. Saquicela, L. Vilches-Blázquez,O. Corcho, “Semantic Annotation of RESTful Services Using External Resources,” in *Current Trends in Web Engineering*, Lecture Notes in Computer Science., vol. 6385, Springer, 2010, pp. 266–276.

[32] R. Navigli, P. Velardi, “Ontology enrichment through automatic semantic annotation of On-Line Glossaries,” *Managing Knowledge in a World of Networks , Springer-Verlag Berlin Heidelberg*, vol. 4248/2006, pp. 126–140, 2006.

[33] N. Kiyavitskaya, N. Zeni, J. Cordy, L. Mich, J. Mylopoulos, “Cerno: Light-Weight Tool Support for Semantic Annotation of Textual Documents,” *Journal Data & Knowledge Engineering*, vol. 68 Issue 12, Dec. 2009.

[34] “Annotea Project,” *W3C annotea*. [Online]. Available: http://www.w3.org/2001/Annotea/. [Accessed: 05-Jan-2012].

[35] V. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, F. Ciravegna, “Semantic annotation for knowledge management: Requirements and a survey of the state of the art,” *Journal Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 4, pp. 14–28, 2006.

[36] "OWLIR: Information Retrieval on The Semantic Web." [Online]. Available: http://ebiquity.umbc.edu/project/html/id/20/OWLIR-Information-Retrieval-On-The-Semantic-Web. [Accessed: 05-Jan-2012].

[37] "SMORE - Create OWL Markup for HTML Web Pages," *Maryland Information and Network Dynamics Lab Semantic Web Agents Project*. [Online]. Available: http://www.mindswap.org/2005/SMORE/.

[38] "Apolda," *Apolda - User Documentation*. [Online]. Available: http://apolda.sourceforge.net/. [Accessed: 05-Jan-2012].

[39] "Armadillo from The University of Sheffield," *Armadillo*. [Online]. Available: http://www.aktors.org/technologies/armadillo/. [Accessed: 05-Jan-2012].

[40] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, M. Goranov, "KIM – Semantic Annotation Platform," *Journal of Natural Language Engineering*, vol. 10, no. 3–4, pp. 375–392, Sep. 2004.

[41] F. Ciravegna, A. Dingli, J. Iria, Y. Wilks, "Multi-strategy Definition of Annotation Services in Melita," *Proceedings of Workshop on Human Language Technology for the Semantic Web and Web Services, 2nd International Semantic Web Conference, Sanibel Island, Florida*, 2003.

[42] Raytheon BBN Technologies, "SemWebCentral," *OntoMat-Annotizer*. [Online]. Available: http://projects.semwebcentral.org/projects/ontomat/. [Accessed: 05-Jan-2012].

[43] D. Maynard, Y. Li, W. Peters, "NLP Techniques for Term Extraction and Ontology Population," *Proceedings of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, 2008.

[44] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. Tomlin, J. Zien, "SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation," *WWW '03 Proceedings of the 12th international conference on World Wide Web*, 2003.

[45] F. Giunchiglia, J. Mylopoulos, A. Perini, "The Tropos Software Development Methodology: Processes, Models and Diagrams," in *Agent-Oriented Software Engineering III*, vol. 2585, Springer, 2003, pp. 162–173.

[46] A. Ghose, G. Koliadis, "Actor Eco-systems: From High-Level Agent Models to Executable Processes via Semantic Annotations.," *COMPSAC (2) 2007*, pp. 177–184.

[47] A. Cuevas, "Unión de ontologías usando propiedades semánticas," Ph.D. Dissertation of Philosophy in Computer Science, Instituto Politécnico Nacional, Mexico, DF, 2006.

[48] R. Guizzardi, G. Guizzardi, “Applying the UFO Ontology to Design an Agent-Oriented Engineering Language,” *ADBIS’10 Proceedings of the 14th east European conference on Advances in databases and information systems*, 2010.

[49] S. A. DeLoach, L. Padgham, A. Perini, A. Susi, and J. Thangarajah, “Using three AOSE toolkits to develop a sample design,” *Int. J. Agent-Oriented Softw. Eng.*, vol. 3, no. 4, pp. 416–476, May 2009.

[50] J. Gómez-Berbís, R. Colomo-Palacios, J.López-Cuadrado, I. González-Carrasco, A. García-Crespo, “SEAN: Multi-ontology semantic annotation for highly accurate closed domains,” *International Journal of the Physical Sciences*, pp. 1440–1451, 2011.

[51] M. Erdmann, A. Maedche, H.-P. Schnurr, S. Staab, “From manual to semi-automatic semantic annotation: About ontology-based text annotation tools,” *Proceedings Of The Coling 2000 Workshop On Semantic Annotation And Intelligent Content*, 2000.

[52] “TravelOntology,” *Semantic Web Document*. [Online]. Available: http://fivo.cyf-kr.edu.pl/ontologies/test/VOTours/TravelOntology.owl. [Accessed: 15-Feb-2012].

[53] “Urban ontology,” *SemanticWebDocument*. [Online]. Available: http://cui.unige.ch/isi/onto/2010/urba-en.owl. [Accessed: 15-Feb-2012].

[54] “University ontology for benchmark tests,” *Semantic Web Document*, 15-Feb-2012. [Online]. Available: http://swat.cse.lehigh.edu/onto/univ-bench.owl.

[55] SemanticWebDocument, “University.OWL,” 15-Feb-2012. [Online]. Available: http://www.webkursi.lv/luweb05fall/resources/university.owl.

[56] “ScienceOWLq.owl,” *Semantic Web Document*. [Online]. Available: http://www.cin.ufpe.br/~mlfc/onto/scienceOWLq.owl. [Accessed: 15-Feb-2012].

[57] “Portal OWL,” *Semantic Web Document*. [Online]. Available: http://www.aktors.org/ontology/portal. [Accessed: 15-Feb-2012].

[58] “Research ontology,” *Semantic Web Document*. [Online]. Available: http://www.mindswap.org/2004/multipleOnt/FactoredOntologies/FactoredAcademia/ka.owl. [Accessed: 15-Feb-2012].

[59] “Travel ontology,” *Semantic Web Document*. [Online]. Available: http://fivo.cyf-kr.edu.pl/ontologies/test/VOTours/TravelOntology.owl. [Accessed: 15-Feb-2012].

[60] “Organization.owl,” *Semantic Web Document*. [Online]. Available: http://ias.in.tum.de/ontology/organisation. [Accessed: 12-Feb-2012].

[61] “Anatomy ontology,” *Semantic Web Document*. [Online]. Available: http://onto.eva.mpg.de/obo/fly_anatomy.owl. [Accessed: 15-Feb-2012].

[62] “People ontology,” *Semantic Web Document*. [Online]. Available: http://owl.cs.manchester.ac.uk/2007/07/sssw/people.owl. [Accessed: 15-Feb-2012].

[63] “Wine ontology,” *Semantic Web Document*. [Online]. Available: http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine. [Accessed: 15-Feb-2012].

[64] “Robot-world,” *Semantic Web Document*. [Online]. Available: http://www.informatik.uni-bremen.de/~shi/Lehre/lang-tech-bremen-05/student-work/semantic-rep-assignment/du-liang/semantic_analysis/robot-world.owl. [Accessed: 15-Feb-2012].

[65] S. Nirenburg, V. Raskin, *Ontological Semantics*, Cambridge, MA: The MIT press. 2004.

[66] S.Nirenburg, S. Beale, M. McShane, “Baseline Evaluation of WSD and Semantic Dependency in OntoSem,” presented at the Proceeding STEP ’08 Proceedings of the 2008 Conference on Semantics in Text Processing, 2008, pp. 315–326.

[67] M. McShane, S. Beale, S. Nirenburg, “OntoSem methods for processing semantic ellipsis,” *Proceeding CLS ’04 Proceedings of the HLT-NAACL Workshop on Computational Lexical Semantics*, pp. 1–8.

[68] S.Nirenburg, S. Beale, M. McShane, “Increasing Understanding: Interpreting Events of Change,” *Proceedings of the OntoLex 2005 Workshop*, pp. 43–52, 2005.

[69] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, “Sweetening WORDNET with DOLCE,” *AI Magazine*, vol. 24, no. 3, pp. 13–24, 2003.

[70] G. Barzdins, N. Gruzitis, R. Kudins, “Re-engineering OntoSem Ontology Towards OWL DL Compliance,” in *Proceedings of the Seventh Joint Conference on Knowledge-Based Software Engineering*, 2006, pp. 157–166.

[71] Semantic Web Document, “OntoSem ontology.” [Online]. Available: http://www.csee.umbc.edu/~aks1/ontosem.owl. [Accessed: 05-Feb-2012].

[72] F. Alencar, J. Castro, G. Cysneiros, J. Mylopoulos, “From Early Requirements Modeled by the i* Technique to Later Requirements Modeled in Precise UML,” *WER 2000*, pp. 92–108.

[73] C. Cares, X. Franch, A. Perini, A. Susi, “Towards interoperability of i* models using iStarML,” in *Computer Standards & Interfaces*, vol. 33, 2011, pp. 69–79.

[74] C. Cares, X. Franch, A. Perini, A. Susi, “iStarML. The i* Mark-up Language Reference’s Guide,” *LSI Technical Report.*, 2007.

[75] C. Cares, X. Franch, A. Perini and A. Susi, “iStarML: an XML-based Interchange Format for i* Models,” *Proceedings of the 3rd International i* Workshop – istar08*, 2008.

[76] D. Amyot, “jUCMNav: Juice up your modelling!,” 05-Apr-2012. [Online]. Available: http://jucmnav.softwareengineering.ca/ucm/bin/view/ProjetSEG/WebHome. [Accessed: 18-May-2012].

[77] K. Najera, A. Perini, A. Martinez, and H. Estrada, “Supporting i* Model Integration through an Ontology-based Approach,” presented at the iStar, 2011, pp. 43–48.

[78] W3C, “RDF Validation Service,” *Check and Visualize your RDF documents*, 2007. [Online]. Available: http://www.w3.org/RDF/Validator/.

[79] E. S. K. Yu and L. Liu, “Modelling Trust for System Design Using the i* Strategic Actors Framework,” presented at the Trust in Cyber-societies, 2000, pp. 175–194.

[80] H. Sahilu, A. Villafiorita, K. Weldemariam, M. Belachew, and A. Zewge, “Designing distributed agricultural information services for developing countries,” presented at the ACM DEV, 2012, p. 24.

[81] E. Yu, Y. Yu, “Organization Modelling Environment,” *University of Toronto, CANADA*, 03-May-2000. [Online]. Available: http://www.cs.toronto.edu/km/ome/.

[82] A. Siena, M. Morandini, “Tool for Agent Oriented Modeling,” *A projecy of the SE Division at Foundaziones Bruno Kessler*. [Online]. Available: http://selab.fbk.eu/taom/.

[83] Copyright © 2012 Stanford Center for Biomedical Informatics Research, “protégé,” *Protégé Project*.