VRIJE
UNIVERSITEIT
BRUSSEL

GHENT
UNIVERSITY

Faculty of Economic and
Social Sciences and
Solvay Business School

Faculty of Economics
and Business
Administration

On the Symbiosis between Conceptual Modeling and Ontology Engineering:

Recommendation-Based Conceptual Modeling

Dissertation submitted in fulfillment of the requirements of the degree of Doctor of Business Economics

Nadejda Alkhaldi
PhD Candidate
PhD program for joint supervision and awarding of a doctorate diploma between the Vrije Universiteit Brussel and Ghent University

Supervisor Vrije Universiteit Brussel: Prof. dr. Wouter Verbeke
Supervisor Ghent University: Prof. dr. Frederik Gailly
Supervisor Universitat Jaume I de Castelló: Prof. dr. Sven Casteleyn

Academic year: 2017 – 2018

Examination Board

Prof. dr. Wouter Verbeke (VUB)
Prof. dr. Frederik Gailly (UGent)
Prof. dr. Sven Casteleyn (Universitat Jaume I de Castelló)
Prof. dr. Peter De Bruyn (VUB)
Prof. dr. Tias Guns (VUB)
Prof. dr. Geert Poels (UGent)
Prof. dr. Sergio de Cezare (University of Westminster)
Prof. dr. Monique Snoeck (KU Leuven)

Acknowledgement

I would like to show appreciation for my supervisors: Frederik Gailly, Sven Casteleyn, and Wouter Verbeke for their advice and support during the writing of this dissertation. I want to say thank you to my colleagues at the VUB and UGent for fun conversations over lunch, and emotional support during the time of hardship. Additionally, I want to thank my friends who helped me to get the most of my stay in Belgium, and were there for me when needed. Finally, thank you to my parents and siblings for their support and for having trust in me even when I did not seem to find it within myself.

Table of Contents

Examination Board **2**

Acknowledgement **3**

List of Figures **7**

List of Tables **8**

Abstract **10**

Introduction **14**

1.1 Background 14
1.1.1 Ontology 14
1.1.2 Ontology Engineering 16
1.1.3 Conceptual Modeling 19
1.1.3.1 Business process modeling 20
1.1.3.2 Goal modeling 23
1.2 Conceptual Model vs Ontologies 24
1.3 Problem Description 27
1.4 Research Goals 30
1.5 Methodology 31
1.6 Publications 33
1.7 Thesis Structure 34

Recommendation-Based Conceptual Modeling and Ontology Evolution Framework **36**

2.1 Semantic Alignment of Conceptual Models 36
2.2 Requirements for CMOE+ 40
2.3 Development of CMOE+ 43
2.3.1 Ontology Evolution Cycle: 45
2.3.1.1 Ontology setup 46
2.3.1.2 Ontological storage and recommendation services 47
Recommendation services 48
2.3.1.3 Community – based ontology feedback evaluation 55
2.3.1.4 Ontology evolution 56
2.3.2 Conceptual Modeling Cycle 57
2.3.2.1 Ontological analyses of the modeling language 57
2.3.2.2 Conceptual model creation 60
2.3.2.3 Conceptual model evaluation 63
2.4 Correspondence Between CMOE+ Requirements and Phases 65
2.5 Conclusion 67

Recommendation-Based Process Modeling....72
3.1 Introduction....72
3.2 Recommendation-Based Conceptual Modeling and Ontology Evolution (CMOE+) Framework....74
3.2.1 Ontology Setup....74
3.2.2 Ontological Analysis of the Conceptual Modeling Language....75
3.2.3 Ontology Storage and Recommendation Services....76
Recommendation Services....78
3.2.4 The Conceptual Model Creation Phase....81
3.3 Recommendation-Based Business Process Modeling (CMOE+BPMN)....82
3.3.1 Ontology Storage....84
3.3.2 Recommendation Services....85
3.4 Evaluation of CMOE+BPMN....86
3.4.1 Experimental Design....87
3.4.2 Experiment Measures....88
3.4.3 Experimental Results....89
3.5 Conclusions and Future Work....93

Aligning I* Models and BPMN Models Using the CMOE+ Framework....96
4.1 Aligning Goal and Process Models....96
4.2 CMOE+ I* Tool....99
4.3 Extending CMOE+BPMN with I*-specific Recommendation Rules....102
4.3.1 Model Repository....102
4.3.2 Rule-Based Recommendation Services....103
4.4 Demonstration....108
4.5 CMOE+X Tool Architecture....111
4.6 Conclusion....112

Conclusions and Future Work....116
5.1 Research Results....116
5.2 Contributions....121
5.3 Limitations and Future Work....122

Acronyms....126

References....128

BPMN Constructs....138

I* Constructs....141

Unified Foundational Ontology (UFO)....144

Correspondence between ESO and UFO....146

Loan Application Case Description....148

Pre-survey 149

Post-survey..... 150

Reference Model..... 151

List of Figures

Figure 1: Main activity categories within ontology engineering process. Adapted from (Gomez - Perez et al. 2003) 17
Figure 2: Differences between ontology and conceptual model taken from (Fonseca & Martin 2007) 27
Figure 3: Regulative cycles of the PhD inspired by (Wieringa & Heerkens 2006) 32
Figure 4: Different phases of CMOE+, and the interaction between them 44
Figure 5: IS research framework followed during the development of CMOE+ 45
Figure 6: Recommendation services 50
Figure 7: Model language-based recommendation service 52
Figure 8: Label-based recommendation service 53
Figure 9: Rule-based recommendation service 55
Figure 10: Recommendation-based Conceptual Modeling and Ontology Evolution (CMOE+) Framework 75
Figure 11: Ontologies within CMOE+ framework 78
Figure 12: BPMN tool 83
Figure 13: BPMN meta-model 85
Figure 14: Ontology Recommendation view (Left) and Ontology Concept Properties view (Right) 86
Figure 15: I* goal dependency 100
Figure 16: Screenshot of CMOE+i* 102
Figure 17: An excerpt of an i* model created by the CMOE+i* tool (on the left), and the corresponding OWL representation stored in the model repository (on the right) 103
Figure 18: Graphical representation of Rule 3 106
Figure 19: Graphical representation of Rule 4 107
Figure 20: Graphical representation of Rule 5 108
Figure 21: Simplified process model for the emergency department 110
Figure 22: Components of CMOE+ tool. The components highlighted in red need to be altered for every modeling language 112
Figure 23: The final version of CMOE+ 114
Figure 24: Phases of CMOE+ which are elaborated in detail within this dissertation 119

List of Tables

Table 1: Existing approaches to enhancing semantic alignment of models using ontology 38
Table 2: Requirements for the proposed framework 65
Table 3: Summary of different phases of CMOE+ 67
Table 4: Correspondence between BPMN and UFO 85
Table 5: SWRL rules used by the rule-based recommendation service 87
Table 6: Results of semantic quality evaluation model annotation 90
Table 7: Naming for BPMN elements with underlying meaning "customer". Columns are modeler-entered labels; rows are treatments; cells denote number of uses of the label / total number of occurrences of BPMN constructs with underlying meaning "customer" 91
Table 8: Naming for BPMN elements with underlying meaning "loan application". Columns are model-entered labels; rows are treatments; cells denote number of uses of the label / total number of occurrences of BPMN constructs with underlying meaning "loan application" 91
Table 9: Time needed for model creation 91
Table 10: Post-survey results for Ease of Use (PEOU), Usefulness (PU), and Community Acceptance (IU); cell values denote a Likert scale value (1-5), with 1 being best and 5 worst for PEOU, and 5 best and 1 worst for PU and IU 92
Table 11: Ontological analyses of i* 99

Abstract

Within an enterprise, different conceptual models, such as process, data, and goal models, are created by various stakeholders. These models are fundamentally based on similar underlying enterprise (domain) concepts, but they have a different focus, are represented using different modeling languages, take different viewpoints, utilize different terminology, and are used to develop different enterprise artefacts (such as documents, software, databases, etc.); therefore, they typically lack consistency and alignment. Another issue is that modelers have different vocabulary selections and different modeling styles. As a result, the enterprise can find itself accumulating a pile of models which cover similar aspects in different manners. Those models are not machine-readable and cannot be processed automatically. Enterprise-Specific Ontologies (ESOs) aim to solve this problem by serving as a reference during the conceptual model creation. Using such a shared semantic repository makes conceptual models semantically aligned and facilitates model integration. However, managing those ontologies is complicated; an enterprise is an evolving entity, and as it changes, the ESO might become outdated.

This research aims to assist modelers within an enterprise to create aligned conceptual models by basing them on the ESO. To do so, the ESO has to be encapsulated and presented to the modelers through an interface which does not require advanced knowledge on ontologies. Since the ESO can be very extensive in size, the proposed solution must incorporate recommendation services to support the modeler in viewing the ESO concepts in an ordered and supportive manner.

During the years of research dedicated to this dissertation, the Recommendation-Based Conceptual Modeling and Ontology Evolution (CMOE+) framework was developed. This framework establishes a symbiotic relationship between the Ontology engineering and the Conceptual modeling fields. CMOE+ consists of two cycles: the Ontology Evolution cycle and the Conceptual Modeling cycle. The *Ontology Evolution cycle* is responsible for setting up the initial version of the ESO and updating it as the knowledge within the enterprise is

updated. Additionally, this cycle encapsulates recommendation services to perform ontology look-up and to present the most relevant ESO concepts in support of the modeler. The *Conceptual Modeling cycle* is responsible for the creation of conceptual models in different modeling languages based on the ESO. CMOE+ was developed based on requirements identified as a result of a literature review and a case study. The development process follows the Design Science Research Methodology (DSRM). After the initial version of CMOE+ was put forward, the focus was narrowed towards the recommendation-based conceptual modeling part of CMOE+, and continued to gradually improve the framework in iterations until it reached its current state. The Ontology Evolution Cycle is not fully addressed within this dissertation. CMOE+ is a general framework which is not bound to a particular modeling language. Moreover, different phases of CMOE+ can be adapted to accommodate the particular needs of the enterprise.

In order to demonstrate the performance and usability of CMOE+, it was exemplified for process modeling using BPMN and goal modeling using i*. This thesis presents a detailed instantiation, and explains which steps are to be performed in order to instantiate CMOE+ for other modeling languages. In order to evaluate the process modeling instance of CMOE+, a CMOE+BPMN tool was implemented. This tool incorporates a BPMN modeler, facilitates storage and access of the ESO, and includes all algorithms functioning within CMOE+ for the BPMN modeling language (as some of the algorithms are language dependent). Next, CMOE+ was exemplified using the i* goal modeling language. Finally, we tested the ability of CMOE+ to perform alignment between i* and BPMN models, in order to show that CMOE+ is indeed beneficial in achieving interoperability among models created in different modeling languages and covering distinct aspects of the enterprise.

The main contributions of this thesis are summarized as following:

1. Proposing a comprehensive framework for conceptual model creation based on an ESO, while simultaneously updating the ESO based on the knowledge captured from those models. This framework is language-independent, and it improves semantic alignment of models already during model creation. Additionally, this framework applies the theory of ontological analyses of the modeling language in practice.

2. Establishing recommendation services which access the ESO, scan it, and, by using a reconfigurable set of algorithms determine the ESO concepts with the most potential relevance for the modeler during model creation.

3. Implementing a prototype of the CMOE+BPMN tool which incorporates all the features addressed in this thesis. This tool was evaluated with students using Moody's Method Evaluation Model (MEM). Moreover, this work explains what needs to be adjusted while instantiating CMOE+ for other modeling languages.

4. Establishing semantic alignment between goal models in i* and process models in BPMN. This work demonstrates how CMOE+ framework is used to create BPMN models which are semantically aligned to i* models previously created within an enterprise.

5. Facilitating semantic annotation of different types of conceptual models. CMOE+ framework allows annotating modeling element with concepts from the enterprise-specific ontology. The annotation is performed during model creation.

1

Introduction

This chapter provides the background on which the research is built. It clarifies the main concepts and approaches featuring in the thesis including Ontology, Ontology Engineering, and Conceptual Modeling. It explains the difference between ontologies and conceptual models within the context of this thesis. Further, the existing problem, research goals are described. An overview of the followed research methodology is provided. Finally, this chapter lists the papers published within the course of this PhD, and concludes with giving an overview of the structure.

1.1 Background

This PhD thesis touches upon two main topics: Ontology Engineering and Conceptual Modeling. It aims at exploring the relationship between both concepts and creating an environment where both of them can coexist and greatly benefit from each other. This first section defines both concepts, and clarifies the difference between the two.

1.1.1 Ontology

Ontologies originate in the philosophy discipline, but nowadays they are widely used in artificial intelligence, computer science, knowledge engineering, and many other fields. Given this popularity, many definitions of ontology were suggested by researchers. One of the first definitions of Ontology was put forward by Neches and his colleagues (Neches et al. 1991, page 4) and states: “An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary”. The most cited definition of the ontology was provided a few

years later by Gruber: “Ontology is an explicit representation of conceptualization” (Gruber 1993, page 1). Ontologies are sometimes confused with taxonomies (Studer et al. 1998). However, an ontology captures more domain semantics than pure taxonomical arrangement of domain concepts. In addition to taxonomical relationships, the ontology adds relationships, properties of concepts, axioms and constraints on concepts’ usage.

Through the years, ontologies were built and represented using a variety of representation techniques. In the beginning of 90s, AI modeling techniques based on frames and first order logic were used (Lenat and Guha, 1990). In the same field, Gruber (1993) suggests using five main components: *classes* which represent ontological concepts, *relations* standing for associations among the concepts, *functions* are special types of relations where one element depends on another, *formal axioms* allow modeling sentences which are always true. And finally, *instances* represent individuals in the ontology. Later, when the emergence of semantic web, description logics based techniques (Baader et al, 2003) gained popularity for ontology representation, and various description logic languages were developed such as Ontology Inference Layer OIL (Fensel et al. 2000) and Web Ontology Language OWL (Bechhofer et al. 2003). Description logics techniques permit formalizing ontologies using three components: *concepts* stand for ontological concepts, *roles* which are binary relations among ontological concepts, and *individuals* represent instances of ontological concepts. Representing an ontology by means of description logics allows the separation of terminological knowledge from assertional knowledge by using respectively the TBox and ABox mechanisms of description logics theory. TBox is responsible for presenting terminological knowledge of the ontology covering concepts with their definitions, formal properties and roles, while ABox is used to represent individuals of concepts in a particular domain.

Apart from AI related techniques, ontologies can be represented using software engineering techniques, such as the Unified Modeling Language (UML) (OMG 2007). The UML representation is easy to understand for people without computer science background, and there are many Computer Aided Software Engineering CASE tools available. UML allows modeling ontological concepts, their instances, and relationships between the concepts. Database technology can also be used for modeling ontologies, such as Entity / Relationship diagrams (ER) (Chen 1976).

Ontology researchers make a distinction between ontologies at three different levels: core ontologies, domain ontologies and application ontologies (Guarino 1998). A core (high-level) ontology describes general concepts that are independent of a specific domain. Examples are the Suggested Upper Merged Ontology (SUMO) (IEEE Standard Upper Ontology Working Group n.d.), Descriptive Ontology for Linguistic and Cognitive Engineering DOLCE (Gangemi et al. 2002) and the Unified Foundational Ontology (UFO) (Guizzardi & Wagner 2010c). A domain ontology describes the concepts, relations and axioms specific to a particular

domain (van Heijst et al. 1997) (like medicine, or automobiles) by specializing the terms introduced in the core ontology. Finally, an application ontology adds specificities to the domain ontology which are only relevant for the application considered, but are not shared across the entire domain.

An enterprise ontology is considered as a specific type of domain ontologies. It describes shared concepts and relationships across an enterprise. In literature, the term "enterprise ontology" refers to an ontology that has its origins in economic and business theory, and that is not specific to a particular business. Well-known examples are the Enterprise Ontology (Uschold et al. 1996), and Toronto Virtual Enterprise Ontology TOVE (Fox 1992).

The research presented here benefits from ontologies at various levels but the main concern and the final product of this thesis is an Enterprise-Specific Ontology (ESO). ESOs are a special type of ontologies that differ from enterprise ontologies in the fact that their Universe of Discourse is a specific enterprise, rather than the enterprise domain. They may have their origin in an established domain ontology or in an enterprise ontology, but their main goal is describing the concepts, relations and axioms that are shared within a particular enterprise. Enterprise-specific ontologies are getting increasingly important in the context of data governance and knowledge representation (Bera et al. 2011).

1.1.2 Ontology Engineering

Gomez – Perez et al define *ontology engineering* as "the set of activities which concerns the ontology development process, the ontology life cycle, and the methodologies, tools and languages for building ontologies" (Gomez - Perez et al. 2003). Building and maintaining ontologies is a tedious and complex process. Following a methodologic framework can considerably facilitate this process by adding structure, decomposing the process into manageable tasks and clarifying the responsibilities of each participant (Simperl & Tempich 2006). One of the first researchers proposing such a framework was Fernandez – Lopez and his colleagues in 1997 (Fernández-López et al. 1997). They offered a methodology for ontology construction – METHONTOLOGY - which was based on IEEE standards for software development (IEEE 1997). Later, the literature identified three activity categories crucial within the ontology development process, especially when the cooperating team is distributed over geographical locations. According to (Gomez - Perez et al. 2003) those categories are ontology management, ontology development, and ontology support as presented in Figure 1 below.

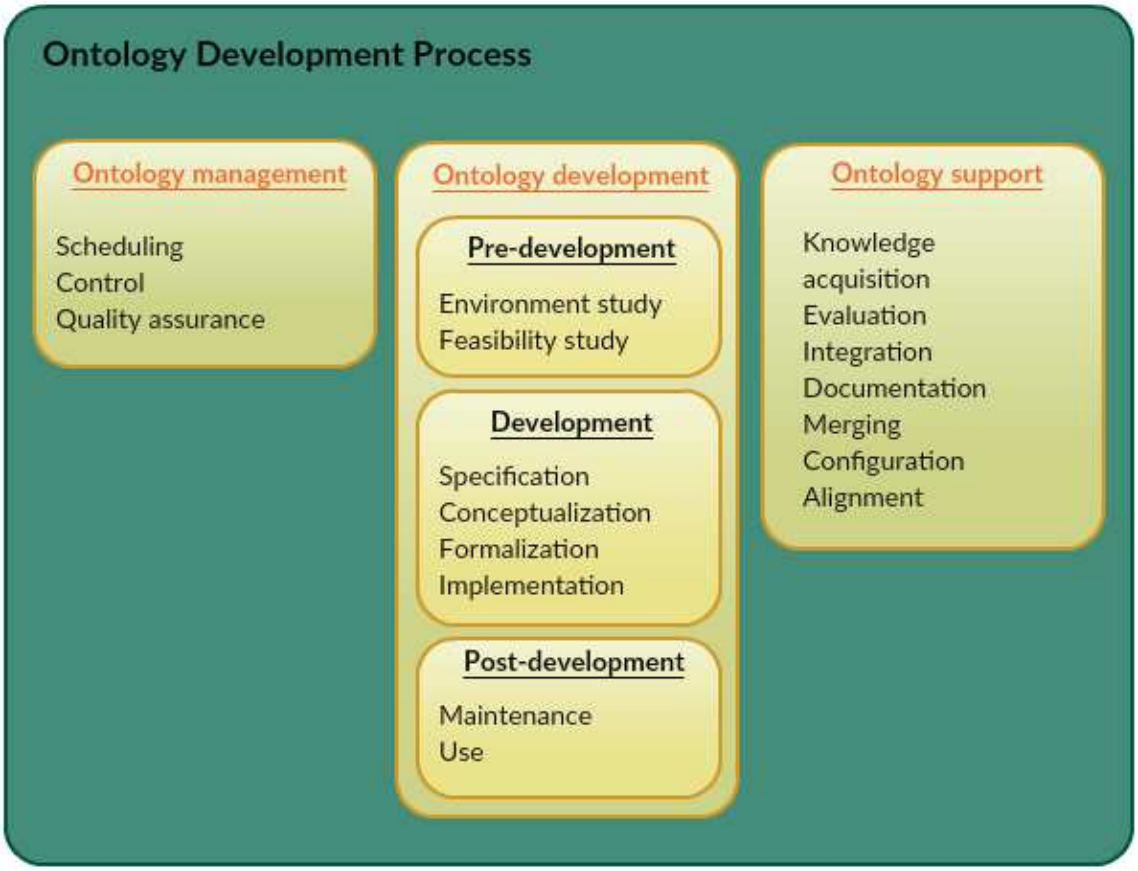

Figure 1: Main activity categories within ontology engineering process. Adapted from (Gomez - Perez et al. 2003)

The **Ontology management** category constitutes three activities: Scheduling, Control, and Quality assurance. The *Scheduling* activity prescribes the tasks to be performed, their order, and time and resources required to accomplish each task. The *Control* activity is responsible for monitoring the scheduled tasks and ensuring that everything runs according to the predefined schedule. The last activity, *Quality assurance*, ensures that the quality of the final product corresponds to quality standards.

The activities within the **Ontology development** category are grouped into Pre – development, Development, and Post – development. During P*re – development* activities an environmental study is performed to capture information about the platform and the applications where the ontology will be integrated. The *Development* activities include documenting intended uses of the ontology and its end – users, structuring domain knowledge as meaningful models, formalizing those models, and implementing them into an ontology language. The *Post – development* activities include ontology maintenance, update and use.

The **Ontology support** category constitutes of activities related to Knowledge acquisition, Evaluation, Integration, Merging, Alignment, Documentation, and Configuration. Those activities are performed in parallel with ontology development activities.

Various methods and methodologies put forward in the literature are not only concerned with ontology development from scratch, but also with utilization and reuse of what is available. Ontology development is an effort intensive process, hence complete or partial reuse of existing products is absolutely necessary. An example of such a methodology is the one proposed by (Gómez-Pérez & Rojas-Amaya 1999). If the knowledge to be reused is presented in several ontologies, those ontologies can be either merged or aligned. Noy and Musen (Noy & Musen 1999) define **ontology alignment** as constructing mappings (or links) between two ontologies in a way that both ontologies are preserved. **Ontology merging** is defined as a process of generating a new ontology from the original ones. Ontology reuse started to capture the attention of scientists. Recently, Katsumi and Gruninger published their work (Katsumi & Gruninger 2016) where they define the boundaries of ontology reuse and its possibilities.

If an ontology engineer has decided to build an ontology from scratch, he can benefit from ontology learning techniques. **Ontology learning** methods, such as (Kietz et al. 2000), are used to minimize efforts necessary for acquiring the knowledge for building ontologies. Ontology learning techniques partially automate knowledge acquisition process by means of natural language analyses or machine learning techniques. Maedche and Staab (2000) distinguish four ontology learning approaches. (1) Learning from text, which utilizes text corpora (2) learning from instances where concrete instances are used (3) learning from schemata such as relational database, XML schemas and data models (4) learning from interoperability where semantic mappings between elements of two ontologies are learnt.

Following from the fact that an ontology should be shared by a community, the members of a community also play an important role in the development process. Collaborative ontology engineering includes methods and tools that are designed to support a decentralized group of stakeholders with varying level of skills, experience and responsibility, and divergent agendas, distributed over geographical locations to reach consensus in an incremental fashion. (Simperl & Luczak-Rösch 2014). To guide the collaboration process during the ontology development and give it direction, some methodologies utilize the notion of a specialized *process model* which distributes the role and prescribes policies and tools to be used. Such a process model is found in (Vrandecic et al. 2005) and (Holsapple & Joshi 2002). Collaborative ontology engineering is often an iterative process. Furthermore, several community members at different locations might access the ontology concurrently. This is managed through special versioning software such as (Noy & Musen 2004) and (Luczak-Rösch et al. 2010). In order to structure community negotiation and interaction, methodology such as the Delphi technique (Gupta & Clarke 1996) and Nominal group (Dunnette et al. 1963) are incorporated in the process. Those methodologies are mediated by communication channels such as elementary forums and chats (Tudorache et al. 2008). A group of researchers also attempted to use wikis in collaborative ontology engineering settings. IkeWiki, developed by Schaffert, is a wiki system which allows users to annotate

pages and links between them using semantic OWL and RDF thereby formalizing informal text into formal ontologies (Schaffert 2006). Another example is the OntoWiki approach proposed by Hepp et al (2006) which allows community members to create a URI for a concept, describe and refine its definition, and link it to concepts already available on Wikipedia.

The first comprehensive methodology for collaborative ontology engineering was proposed by Holsapple and Joshi (2002). According to this methodology, an initial ontology was constructed by integrating existing ontologies in the domain. Later the initial ontology was subject to community discussion and revision. Nowadays various collaborative ontology engineering methodologies are available for different purposes. Moor et al propose a methodology specialized in interorganizational settings with many pre-existing ontologies, and ill-defined changing requirements. Hereby, it supports construction of a sequence of increasingly complex versions of interrelated ontologies over time (Moor et al. 2006). Missikoff et al describe an iterative methodology to construct domain-specific ontologies to capture knowledge of a particular enterprise (Missikoff et al. 2015). Some of the methodologies are very rigid and well detailed such as (Vrandecic et al. 2005) and (Kotis & Vouros 2006). Others are more liberal and simple with the aim to involve domain experts, who are not experts in ontologies (Auer 2006).

1.1.3 Conceptual Modeling

Conceptual models describe formal aspects of physical and social world, for the purpose of communication and understanding (Mylopoulos 1992). Conceptual modeling is regarded as a related discipline to requirements engineering (Shanks & Darke 1997) and software engineering (Moody 2005) as conceptual models are used to capture user requirements and build information systems satisfying those requirements. Currently, conceptual modeling goes beyond requirements engineering. The purpose of the conceptual modeling task is to represent phenomena in a domain of choice. It is able to represent both: static (such as concepts, properties and relations) and dynamic phenomena (such as processes and events) (Commentary et al. 2002). Gemino and Wand describe the conceptual modeling task as “a process whereby individuals reason and communicate about a domain in order to improve their common understanding of it” (Gemino & Wand 2004). Thus, different types of conceptual models (data models, requirements models, process models, etc) are used by enterprises for prescribing a certain reality in a company. In (Mehmood et al. 2009) conceptual models are associated with the following objectives:

- Meeting user requirements
- Formally representing observed reality
- Serving as basis for implementation and evaluation of information systems

In this PhD project two types of conceptual modeling languages will be used: business process modeling and goal modeling. We have opted for business process modeling as the first demonstration of our work as according to Davies and colleagues, process modeling was the primary reason to engage in conceptual modeling (Davies et al. 2006). Moreover modeling business processes has become a practice in many organizations (Rosemann 2006), (Pittke et al. 2013). However, despite the importance and wide spread use of those conceptual models, the possibility to retrieve and reuse the captured knowledge is still limited (Lin et al. 2006).

Where process models are used to describe the processes within a particular enterprise, goal modeling covers a completely different but yet very important aspect: it captures the motivation and strategy behind organizational practices (Yu et al. 2006). Goal-oriented modeling became increasingly popular among requirements engineering techniques (Matulevičius et al. 2007). It is primarily used to support early IS development stages as it models the rationale and scope of the IS and helps in resolving conflicts and making sure that all the stakeholders are heard by the development team (Matulevicius et al. 2015). Goal models are capable of representing functional and non-functional requirements (Santos et al. 2010), and shift from traditional modeling focus on *what* and *how* to innovative *who* and *why* (who are the actors, what they want to achieve and what is their motivation) (Moody et al. 2010). Moreover, by utilizing goal modeling, organizations can express their choices behind multiple alternatives, and investigate other possibilities (Jonkers et al. 2004).

Given such a great importance of both conceptual modeling languages, and their different points of focus, we believe that choosing those two options adds rigor to the demonstration of our approach. If it works for such diverse modeling types, then we can claim its generality. Demonstrating it for all (types of) conceptual modeling languages is obviously not possible within the scope of a doctoral thesis.

1.1.3.1 Business process modeling

A basic definition of business process is provided by Hammer, who defines it as ‘‘a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer’’ (Hammer 1990). Aguilar – Saven adds an “enterprise flavor” to the previous definition: “Business process is a partially ordered set of Enterprise Activities which can be executed to realize a given objective of an enterprise or a part of an enterprise to achieve some desired end-result” (Aguilar-Savén 2004). By aggregating both definitions, business process model receives an input, and generates an output by applying a partially ordered set of activities to satisfy business needs of an enterprise. (Lin et al. 2002) identified five essential elements in a business process: (1) a business process has a customer, (2) it constitutes of activities, (3) those activities aim at creating value to the customer, (4) those activities are executed either by people or by agents and (5) a business process often involves several units of the enterprise.

There are a plethora of research efforts concerning business processes modeling. (Aguilar-Savén 2004) perform a review of business process modeling literature. The author concludes that business processes are modeled for three main purposes: 1) to learn about the process, 2) to make decisions about the process and 3) to develop business process software. Based on which need is addressed, the type of process modeling technique is selected. The same author classifies business processes into "core" and "supportive". "Core" is a primary business process which is initiated externally to the enterprise. The "supportive" business process is secondary and it facilitates the execution of the core process. The "supportive" business process can serve as management process which controls overall strategy and objectives of the enterprise.

Many process modeling approaches are proposed in the literature. Kueng et al (1996) classify those process modeling approaches into four main categories:

1. **Activity-oriented approach:** it primarily concentrates on activities (sometimes referred to as tasks). Using this approach, a business process is presented as a set of ordered activities. This representation is suitable for the high-level view on the process, but it might underestimate the true complexity of the work. It disregards the information flow and the involved enterprise units.

2. **Object-oriented approach**: this presents encapsulation, specialization, composition, and other aspect of object – oriented methods (Booch 1994). However, according to the author, this approach is not adequate for business process modeling.

3. **Role-oriented approach**: this modeling approach rotates around "role" concept, which is defined as a position played in a process by an individual, team, or unit. This is a rather broad definition, as it can imply a complete job description, such as school teacher, or a part of the job, or it can be limited to a specific task. The advantage of this modeling approach is the possibility to group activities and assign them to a particular actor. However, this approach does not allow expressing an intricate sequencing logic.

4. **Speech-act oriented approach**: the main concept behind this approach is "ActionWorkflow Loop": every communication process (actionWorkflow) incorporates customer and performer. The communication process itself constitutes of four phases (loop): proposal, agreement, performance, satisfaction. Even though the idea itself is innovative, the approach is not practical in analyzing existing processes or in creating new ones.

More recently, a new paradigm of process modeling has emerged: declarative process modeling. The difference between imperative (traditional) and declarative process modeling is rooted in computer programming (Pichler et al. 2011). *Imperative process modeling* focuses on a "inside-to-outside" approach; it prescribes how exactly the work needs to be executed. All the possible alternatives need to be specified in advance before the execution starts. *Declarative process modeling* follows the "outside-to-inside" approach. It only describes the essential characteristics, and does not prescribe the precise execution procedure.

BPMN

The Business Process Modeling Notation (Object Management Group (OMG) 2008) or Business Process Model And Notation (OMG 2011) (BPMN) was developed by the Business Process Management Initiative (BPMI) Notation Working Group after two years of effort. The main purpose was to develop a notation which is understandable by both business users and developers. BPMN allows various departments within an enterprise to understand their processes, and enables an enterprise as a whole to communicate its processes in a standardized manner. BPMN is designed to cover various types of models and as a result, it enables the creation of end to end business process. As described in (OMG 2006a), there are three basic types of sub models within BPMN 2.0:

1. Processes, which include private non executable business process, private executable business process, and public process. *Private* processes are internal to the organization, while *public* process represent interaction between the organization and external participants.
2. Choreographies: while a regular process is contained in a pool (see Appendix A), choreography exists between pools and represents a definition of an expected behavior.
3. Collaborations: represents interaction between two or more business entities by means of public processes

To learn more about BPMN, the reader can refer to (Silver 2009). BPMN is a relatively young notation dating to 2004. Nevertheless, it underwent several updates. The latest updates (at the time of submitting this thesis) can be found at (Allweyer 2016).

BPMN is capable of conveying a wide variety of information to a broad audience. White identifies two main types of business process diagrams: Collaborative (public) B2B process, and Internal (private) business process (White 2004). The collaborative B2B process captures an interaction between two or more business entities in a neutral way. This process does not take a point of view of any of the organizations. It simply depicts the sequence of activities, and the communication among those organizations. The second type,

Internal business process, acts from the perspective of a particular organization, and depicts its interactions with others (external organizations).

For an overview of BPMN constructs the reader is referred to Appendix A.

1.1.3.2 Goal modeling

Zave and Jackson define a goal as an objective the system under consideration should achieve (Zave & Jackson 1997). Goals cover different types of objectives: functional and non-functional. Functional are associated with the services to be provided, while non-functional deal with the quality of service such as performance and accuracy (Van Lamsweerde 2001).

There are many reasons why goals are important in requirements engineering. Here are some of them:

- Goals provide rationale for requirements, thereby making it easier to explain to stakeholders (Mostow 1985)
- Goal refinement assists in structuring complex requirements documents thereby increasing readability (Van Lamsweerde 2001)
- Modeling goals explicitly helps in detecting conflicts in requirements (van Lamsweerde et al. 1998)

Goal models represent business objectives for stakeholders of an enterprise. Goal modeling is a promising and important methodology for eliciting requirements (Matulevicius et al. 2015; Shibaoka et al. 2007)

There is an established connection in the literature between goal modeling and business process modeling. Here are a few examples: Koliadis et al propose a framework which allows translating changes occurring in one model type (goal or process model) to the other model type (Koliadis et al. 2006). Decreus and Poels automatically derive BPMN models from requirements models which incorporate goal models (Decreus & Poels 2011). Finally, (Santos et al. 2010) proposes applying variability analyses on BPMN models using a goal oriented approach.

I* (I-star)

I* is an agent and goal-oriented modeling framework which analyzes intentional relationships among social actors (Yu et al. 2011). It is one of the most widespread goal-modeling approaches (Franch 2010) Unlike traditional systems and modeling methods which abstract from the social aspect, i* modeling revolves around social actors. Those actors are perceived as intentional; they have goals, beliefs, and commitments. The i* framework raised interest in socially-aware approaches to systems modeling, and led to several extensions and adaptations (Yu 2009). I* is mainly applied in requirements

engineering (especially for early requirements). Additionally, it is used in enterprise architecture, business process redesign, information security and privacy, digital asset protection, and software development among others (Yu et al. 2013). The main constructs of the i* notation are actor, goal, task, resource, and softgoal. The relationships are modeled through Strategic Dependency (SD) and Strategic Rationale (SR) models.

According to Franch, the i* framework reached a high level of maturity from an academic perspective, which resulted in a significant body of knowledge available through literature. Additionally, there are workshops, tutorial and scientific conferences for i*, and an i* wiki which offers explanation of the modeling language and supplies the reader with the initial literature. Despite all the research efforts, i* adoption by practitioners is unfortunately very limited (Franch 2010). For a detailed description of the i* notation the reader is referred to (Yu et al. 2011).

An overview of i* constructs is provided in Appendix B.

1.2 Conceptual Model vs Ontologies

Within the conceptual modelling research community there is no clear agreement with respect to the distinction between ontology and conceptual model. Some researcher consider ontologies as disguised conceptual models (Winter 2001). The literature presents several attempts to differentiate between ontologies and conceptual models which should help to set the boundaries for both. The confusion in literature between conceptual models and ontologies also has its origin in the fact that ontologies can support the conceptual modeling process. More details on how ontologies contribute to the semantic alignment of conceptual models are presented in Chapter 2.

A distinction that has been regularly mentioned is based on whether the model or ontology is descriptive or prescriptive in nature. A conceptual model that is developed within an enterprise forms the template according to which a particular enterprise system is developed (Aßmann & Zschaler 2006). For instance, a business process model prescribes the business process that is implemented by a process-aware information system. In contrast, ontologies are descriptive models, and therefore follow an open-world assumption (Horrocks et al. 2003) (anything which is not stated explicitly is unknown), rather than a closed-world as conceptual models. Ontologies portrait reality but reality is not constructed from them. If an ontology is used in a prescriptive manner, then it is better referred to as specification model (Seidewitz 2003). Most ontologies correspond to structural models whereas conceptual models can give a behavioral view in addition to the structural view. The second distinction proposed by the same author is related to sharedness (Aßmann & Zschaler 2006). An ontology is shared between different enterprises or between various

units and projects within one enterprise. A conceptual model does not need to be shared and can be developed and used by a small set of people within a specific context or project. The notion of sharedness is relative and therefore less useful to make a clear distinction between ontologies and conceptual models.

Within the context of this thesis we adopt the distinction between ontologies and conceptual models proposed by Fonseca and Martin (2007). The concept of *Ontology* described in this thesis is equivalent to the concept of *Computational ontology* in (Fonseca & Martin 2007), which implies an ontology used to build information systems (IS) within the Ontology-Driven IS development approach described by (Guarino 1998). The term *Conceptual model* appearing in this thesis, corresponds to Fonseca and Martin's *Conceptual schema*. Conceptual schema represents the results of the modeling process, merely a set of diagrams constructed in a particular modeling language to express data structures of an application to be developed. In their definition, Fonseca and Martin focus on the data models, however, this thesis incorporates different kinds of models, namely process and goal models. Hence, we borrow their definition and extend it with other kinds of models, not limited to data models. To avoid confusion, throughout this thesis we opt for using the term enterprise-specific ontology ESO (or simply, ontology), and conceptual modeling.

(Fonseca & Martin 2007) differentiates between ontologies and conceptual models based on the *objectives* they aim to satisfy, and the *objects* they incorporate. Those differences are summarized in Figure 2 below. With respect to the objectives:

- **Ontology** aims to offer explanation and information integration based on assumptions about invariant conditions defining the domain of interest. Ontologies supply the users of the IS with common assumptions about the whole, within which the facts about IS arise. Ontology aims to explain the domain by describing it as a coherent whole. It might support the development of an IS, or be used by an IS, but it is not entirely coupled to a specific IS. Ontology is able to serve a broad range of purposes and it supports the modeler during model creation and analyses. An ontology offers a common reference to which various artefacts related to particular projects within the enterprise are connected.

- **The Conceptual model**'s objective is to classify the observed facts, and provide measurement (the linking of general categories with particulars) for the objects along dimensions of possible variation, and within the context of assumptions supplied by the corresponding ontology. Hence, models focus on linking general ontological categories with particular observations within the IS. Conceptual models are usually constructed with a specific IS in mind. Conceptual model aims to structure a set of

instances to facilitate querying the IS. Models from different projects within an enterprise are linked to one enterprise-specific ontology.

With respect to the incorporated objects:

- **Ontology** lays focus on reality and the real world. It represents the general and assumed categories defining the domain of interest. Ontology captures a domain which is intended to be shared by multiple projects within the enterprise. It encapsulates concepts and relationships which are meaningful within a context of a particular enterprise, but are not captured by the models. An ontology has a broader scope as it incorporates more than a conceptual model is able to cover. The ontology engineer is free to incorporate extra information to facilitate understanding of the concepts presented in the ontology.

- **Conceptual model** aims to link the ontology with the "facts". It establishes the relationship between categories presented in the ontology and the range of possible variations of facts related to those ontological categories. For example, if an ontology contains a category of "Automobile", a conceptual model will provide a machinery to link the category of automobile to a specific auto, owned by a particular person. The conceptual model represents a particular enterprise application, and the modeler is restricted by describing terms which are a part of the IS. Within this research project, even though all models are linked to the ontology, they are not restricted by the terminology used within the ontology. The modelers are free in selecting the terminology which suites the project for which the model is created.

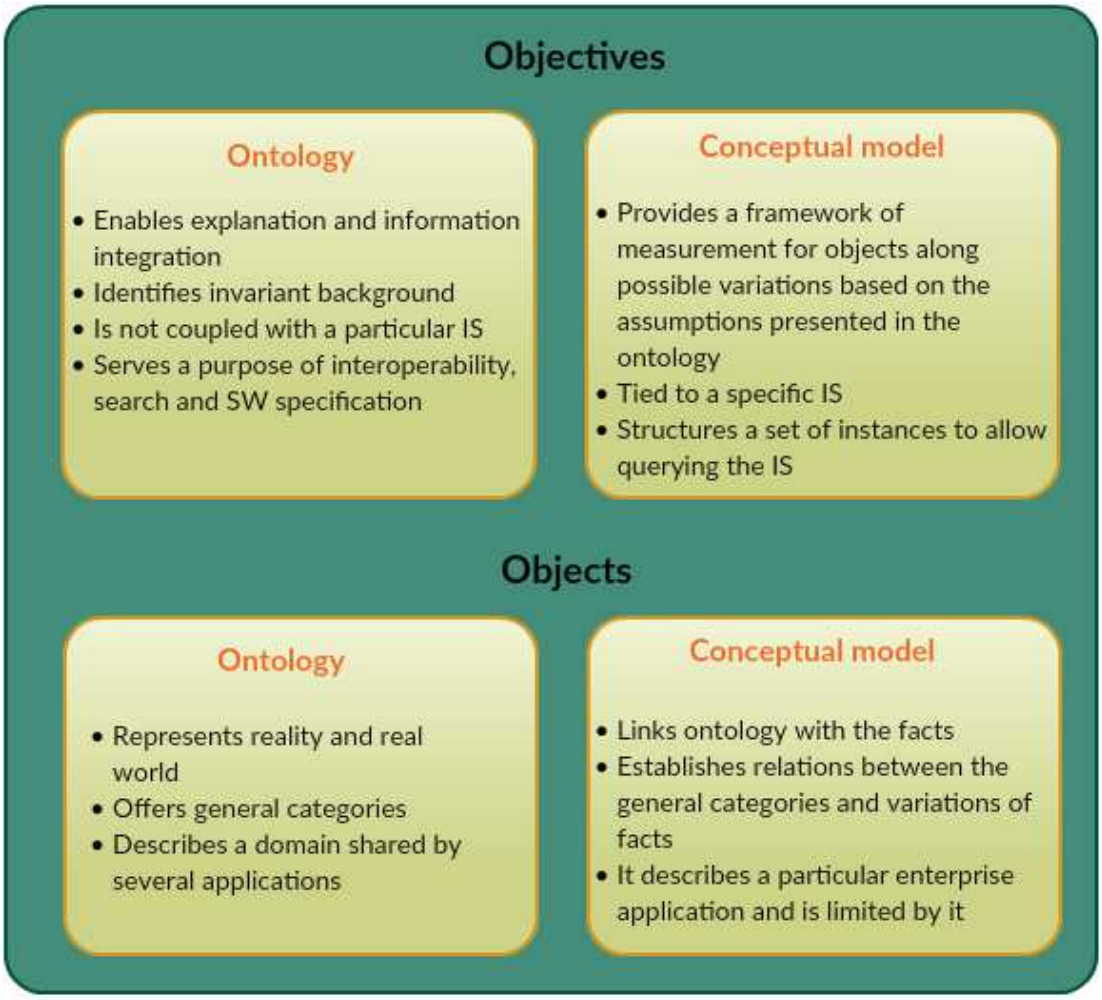

Figure 2: Differences between ontology and conceptual model taken from (Fonseca & Martin 2007)

1.3 Problem Description

Within an enterprise various types of conceptual models are created. Every model captures a particular concern, such as representing processes, intentions and goals, data, etc. All those models are represented in different modeling languages and created by different modelers. This diversity has a negative impact on the semantic alignment of those models. If an enterprise dedicates effort for model creation, it is important that those models are usable within different aspects of the enterprise. Models represented by different modeling languages are naturally lacking semantic alignment as every modeling language uses its own set of constructs, and mapping corresponding constructs manually can be a tedious task. Without additional effort for aligning those models, they will be hard to integrate and perform search. Additionally, models lacking semantic alignment will result in implementation of non-interoperable software systems. Models created by different people suffer label inconsistency and other signs of semantic heterogeneity as people tend to use different terminology to describe the same term. Furthermore, they can use different modeling constructs within one modeling language to model the same semantic concept.

Consequently, enterprise's knowledge base piles a big amount of semantically heterogynous models which are challenging to integrate and reuse.

This research work aims to improve semantic alignment of conceptual models created within one enterprise. This includes models of the same type created using the same modeling language, models of the same type created using different modeling languages, and models of different types. Within the context of this thesis, S*emantic Alignment* of conceptual models implies that overlapping elements of those models are consistent on the model and the meta-model levels. On the model level, model elements can be annotated with ESO concepts. Overlapping model elements are annotated with the same ESO concept, which ensures consistency of model elements. On the meta-model level, corresponding modeling elements are represented by equivalent modeling constructs in different modeling languages. The meta-model level is only applicable to models created in different modeling languages. Models lacking semantic alignment cause problems in communication among stakeholders, result in creation of non-interoperable software systems, and in additional effort and money wasted by the enterprise on integrating its different models, or systems built based on those models.

The following issues may arise when models are not aligned semantically:

- **Synonyms in labels of modeling elements carrying the same semantics**. Different people tend to use different words to depict the same concept as this depends on personal background and profession specific jargon. Models within the enterprise are created by different stakeholders working within different occupations, which can result in using occupation specific jargon. For example, a medical specialist may use "script" while a person without medical background will use the word "prescription" to depict the same meaning. Similarly, one modeler can use a word "client", and another will use the word "customer". Those words have the same meaning which is recognized by humans, but not by machines.

- **Abbreviations used in labelling modeling elements**. Humans tend to use abbreviations which are not recognised by machines even though they sound familiar to humans, such as using "dr" for "doctor", "dept" for "department" and "h" for "hour". Some abbreviations are not even recognized by the majority of people. For example, a stakeholder with a military background can use a label "NVS", and other stakeholders without military background will not understand that it stands for "Night Vision System". Other type of abbreviations can make a particular term ambiguous, such as using "report" to depict "business proposal report". This is clear only in a very limited context. On a broader view "report" can stand for "research report", "technical report" or any other type of report. Hence, using "report" is not precise and causes interoperability problems.

- **Inability to reuse knowledge captured in existing models**. Various conceptual models capture different aspects of the enterprise. Nevertheless, some models intersect and capture redundant information. This is particularly prominent in models of the same type representing overlapping concerns in different modeling languages, such as creating business process models using BPMN and UML activity diagrams. But this situation may also expand to models of different types; for example, process models may overlap with goal models while modeling a series of tasks to be executed to achieve a particular goal.

- **Inability to reuse general knowledge of the domain.** For example, if a Customer acquired a Loan from a Bank, then this relationship between the Customer and the Bank might be useful within the context of an enterprise operating in a financial domain. Some models might capture this relationship explicitly, while others will not. Independently, maintaining such a relationship as a common knowledge within the enterprise is important. It is best to define such a relationship at a higher level, not tight to a particular model. As a result, information about the relationship can be access without the need to search for models capturing it.

A possible solution for this model alignment problem is providing modelers with a shared vocabulary formalized in an ontology (Bera et al. 2011; Francescomarino & Tonella 2009). Enterprise-specific ontologies are gaining importance in the context of Data Governance and knowledge representation (Bera et al. 2011). Supporting tools, such as IBM InfoSphere[1] or Collibra Enterprise Glossary[2] allow enterprises to specify their own enterprise glossary/ontology. Such an enterprise-specific ontology, once available, can subsequently be deployed to help enterprise modelers in creating compatible, conceptual models, such as requirements, data or process models. If an enterprise decides to formalise its knowledge for reuse, it will be faced with various challenges regarding updating, retrieving and querying of the formalised knowledge. Therefore, developing effective practices in this regard is crucial. Another challenge is actually using this knowledge in model creation. How to select the right concepts? Where to look for them? What to do if the right concept is not found?

However, even if such an ontology is available, it might not be used by modelers during the modeling process. There are many reasons for such an underutilization of enterprise knowledge. For example, an interface to the ontology is too complex for a modeler without ontology training, and the absence of supporting tools which assist the modeler to navigate through the ontology.

[1] http://www-01.ibm.com/software/data/infosphere/
[2] http://www.collibra.com/

Another challenge surrounding ESO is how to maintain such an ontology to remain relevant to the enterprise it was created for. An enterprise is an evolving entity which occasionally undergoes changes. Completely new and innovative concepts are emerging, other concepts are becoming obsolete. It is essential that the ontology formalizing knowledge of an enterprise is also updated accordingly. Knowledge which became obsolete has to be removed, while new pieces of knowledge are added. The ontology is created to be actively utilized within the enterprise, and if it is outdated, it cannot be of a great benefit anymore.

To summarize, three research problems are highlighted to be addressed in this thesis:

- Semantic alignment of different types of conceptual models created within an enterprise
- Utilization of ESO during the modeling process
- Keeping ESO up to date with the evolving nature of the enterprise

1.4 Research Goals

The research work presented in this thesis is a part of a bigger research project with the overall goal of establishing and demonstrating a symbiotic relationship between ontology engineering and conceptual modeling. This PhD project addresses the issues related to the conceptual modeling part, while the community-based ontology engineering part will be explored as part of another project. Hence, this PhD aims to establish requirements for a framework which guides modelers through creation of semantically aligned conceptual models, and facilitates ESO maintenance and usage. This framework must operate independently of a particular conceptual modeling type or language to cover all the needs of an enterprise. After establishing the general requirements and translating these requirements to a framework with different cycles and phases, we explore in detail the parts of the framework focusing on the conceptual modeling efforts. Next, those parts (phases) are instantiated and thoroughly evaluated. This results in the following goals for this PhD project:

1. **GOAL1**: Setting overall requirements for establishing the symbiotic relationship between conceptual modeling and ontology engineering, and establishing a framework which will deliver those requirements.

2. **GOAL2**: Providing more detailed explanation regarding those phases of the framework which are related to conceptual modelling. This includes pointing out what can be reused from the literature, and offering practical guidance regarding every phase.

3. **GOAL3**: Evaluating the phases responsible for the conceptual modeling side of the symbiosis by demonstrating the framework for BPMN process modeling language. This implies instantiating the relevant phases of the framework for BPMN, and implementing a tool in order to perform an experimental evaluation. This will represent the first step towards proving framework's generalizability. Next, the same phases will be instantiated for i* goal modeling.

4. **GOAL4**: Exploring the ability of the framework in aligning i* goal models with BPMN process models. After instantiating the relevant phases of the framework for BPMN and i*, we want to investigate the possibility of creating BPMN models which are semantically aligned with previously created and stored i* models. This will demonstrate the operation of the framework across modeling languages.

1.5 Methodology

This research contributes to solving a specific business problem, i.e. semantic alignment of conceptual model, and to create artefacts that can be directly used by enterprise workers. For this purpose the Design Science Research Methodology (DSRM) is perfectly suited (Hevner et al. 2004). The main goal of a design science project is to solve a practical problem. The problem at hand here is twofold: 1) improving semantic alignment of conceptual models, and (2) enriching, evolving and keeping the ESO up to date. Following (Wieringa & Heerkens 2006), a DSRM project is divided into *Engineering Cycles* (EC) that provide the necessary steps to design and evaluate an artefact, and associated *Research Cycles* (RC), responsible for resolving research related issues, such as establishing the state of the art for a specific problem, finding and adapting related techniques, etc. The main problem is thus decomposed into nested problems represented by ECs supported by RC(s).

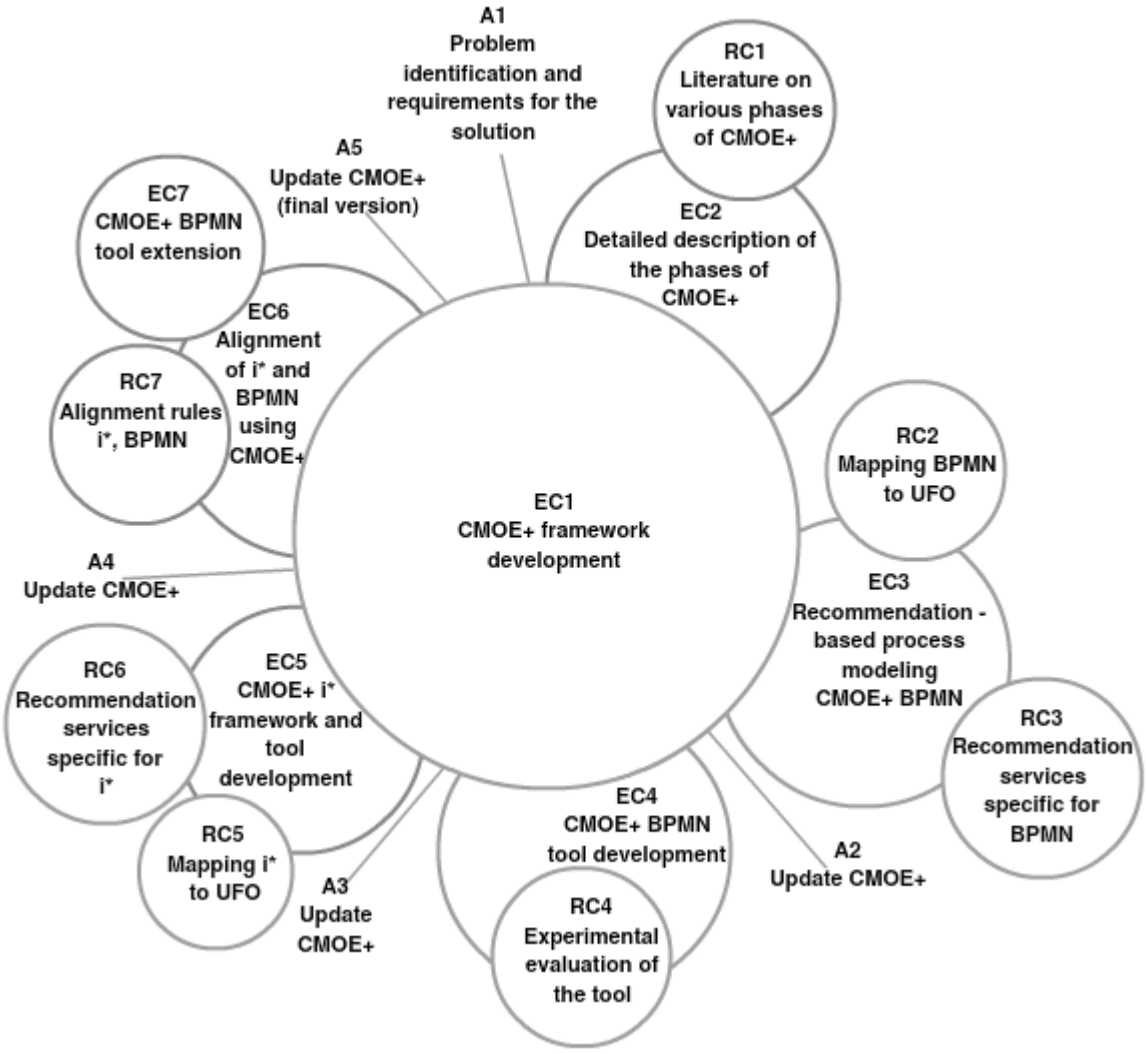

Figure 3: Regulative cycles of the PhD inspired by (Wieringa & Heerkens 2006)

This research project in Figure 3 commences with Problem identification and requirements specification activity **A1**. It focuses on analyzing the problem and defining requirements necessary for the solution. Based on the requirements obtained from A1 the main design science artefact, the Conceptual Modeling and Ontology Evolution framework (CMOE+), was developed in the subsequent engineering cycle **EC2**. Detailed description and ideas of EC2 were supported by RC1. This research cycle incorporated literature on various topics including semantic alignment of conceptual models, ontological analyses, community-based ontology engineering, and more. The CMOE+ framework consists of two cycles: an *Ontology Evolution cycle* which is responsible for ontology maintenance and amelioration, and a *Conceptual Modeling cycle* which facilitates creation of different conceptual models based on the ontology. The framework is explained in detail in Chapter 2. After designing the framework, it was applied to the process modeling (EC3 and EC4) and requirements engineering fields (EC5). EC1 was performed in iterations, and the first version was drafted in **EC2** based on the requirements resulting from A1.

EC3 represents the first instantiation of the CMOE+ framework. An ontology in the financial domain was selected as the starting point for the ontology engineering cycle of the framework. Process modeling using the BPMN modeling language was chosen for the conceptual modeling cycle. EC3 makes use of a broad knowledge base, therefore a thorough literature review was needed. In **RC2** we investigated the possibility of using various core ontologies, and how to map BPMN constructs to the selected core ontology. **RC3** is about finding the optimal combination of matching algorithms that will reduce to the minimum the effort required by the modeler to locate the right concept in the ontology. Some matching algorithms were reused, some specifically designed. Upon completion of EC3, the first modifications were made to the framework (**A2**). All findings of RC2 and RC3 were implemented in a tool (**EC4**) that allowed us to execute experiments with students and evaluate this first instantiation of the framework (**RC4**). Two experiments were conducted with the tool: one at the Vrije Universiteit Brussel, and one at Ghent University. Based on the findings, CMOE+ was enhanced again in **A3.**

To ensure that CMOE+ is suited for different modeling languages and is not biased to one type of conceptual models, the framework was tested in the goal modeling field using the i* modeling language. This resulted in implementing CMOE+ i* tool in **EC5**. The CMOE+ framework was updated accordingly in **A4**. After presenting the performance of CMOE+ for individual modeling languages, it was necessary to demonstrate that the framework is able to operate across different modeling languages. **EC6** establishes the alignment between goal models in i* and process models in BPMN. This is achieved with the help of **RC7** which explores the possibilities of various alignment rules. **EC7** extends the CMOE+ BPMN, implemented within EC4, to offer evidence on the alignment. After completing this last engineering cycle, CMOE+ framework was altered again (**A5**), which resulted in its final version.

1.6 Publications

This section presents the published and submitted papers written during the course of this PhD. the section gives a brief overview of the content of the paper, and the chapter(s) where it was used within this thesis.

- F. Gailly, S. Casteleyn, and N. Alkhaldi. (2013). On the Symbiosis between Enterprise Modeling and Ontology Engineering. *Proceedings of 32nd International Conference on Conceptual Modeling*. pp. 487 –494
  This paper is our first publication and it gives an overview of the initial version of CMOE+ framework. This framework was modified and enhanced in iterations, with every instantiation and evaluation. Hence, this first paper does not include all the details, and uses a slightly different terminology. Some terminology was changed

with the advancement of this research; while presenting our achievements in conferences, we noticed the confusion caused by some terminology, and tried to adapt the terminology to achieve the common understanding with other researchers in the field. This paper is incorporated in Chapter 2.

- N. Alkhaldi, S. Casteleyn, and F. Gailly. (2014). Supporting Process Model Development with Enterprise-Specific Ontologies. *Proceedings of the 16th International Conference on Enterprise Information Systems.* pp. 236–248
  This publication presents an enhanced version of CMOE+, and its instantiation for process modeling using BPMN with a basic demonstration of the instantiation process. This paper is not included in the thesis as it presents only the basic idea behind recommendation – based process modeling.

- Alkhaldi, N., Casteleyn, S. and Gailly, F. (2015). Enterprise-specific Ontology-driven Process Modeling. In J. Cordeiro, S. Hammoudi, L. Maciaszek, O. Camp, & J. Filipe (Eds*.), Lecture Notes in Business Information Processing*. pp. 472-488 Springer
  This work provides a detailed overview of recommendation – based process modeling, which is the first instantiation of CMOE+ framework. This paper is a part of Chapter 2.

- F. Gailly, N. Alkhaldi, S. Casteleyn, and W. Verbeke. (2017). Recommendation-based Conceptual Modeling and Ontology Evolution Framework (CMOE+). *Business & Information Systems Engineering*
  It presents an evaluation of CMOE+ for process modeling. It describes the developed process modeling tool, and the experiment conducted with university students to evaluate the feasibility of such a modeling tool. This paper is presented in Chapter 3.

1.7 Thesis Structure

This PhD thesis constitutes of five chapters. Chapters' division is inspired by the engineering cycles presented in Figure 3 (Section 1.5).

CHAPTER 1: Introduction

This Chapter defines relevant topics, methodologies, and concepts used within this thesis namely "ontology", "ontology engineering" and "conceptual modeling" (Section 1.1). Further, it clarifies the difference between ontology and conceptual models within the context of this thesis in Section 1.2. Next, this chapter describes the problem to be addressed (Section 1.3) and states the goals to be accomplished (Section 1.4). Section 1.5 highlights Design Science Research Methodology, the