Generating Ontologies through Organizational Modeling

Karen Najera
INFOTEC
Mexico D.F., Mexico
Karen.najera@infotec.com.mx

Alicia Martinez
CENIDET
Cuernavaca, Mexico
amartinez@cenidet.edu.mx

Anna Perini
FBK
Trento, Italy
perini@.fbk.eu

Hugo Estrada
INFOTEC
Mexico D.F., Mexico
hugo.estrada@infotec.com.mx

Abstract— Ontologies are recognized as important component of information systems supporting business processes within and across organizations. At modeling time, they contribute identifying key elements from business processes; at development time, their structure can be translated automatically into information system's source code; finally, at run time, through queries and reasoning, they provide proper data for decision making. Additionally, ontologies provide the basis for sharing and publishing organizational information through the Semantic Web. However, representing organizational information directly into an ontology requires specialized expertise in the ontology engineering domain, thus, ontologies are not generated using a domain language easily for the organizational domain experts. In this work, we propose the use of specialized organizational modeling techniques as starting point to model the organization, thereby, ensuring the proper definition of the organizational knowledge. Then, a mechanism is provided to automatically transform the organizational knowledge in its corresponding ontological representation. Our proposed approach is based on Model Driven Engineering ideas and it involves: a) the development of an ontology representing the metamodel of two widely used organizational modeling techniques *i and Tropos and b) the systematic transformation of *i** based modeling primitives into instances of the ontology.**

Keywords-Organizational modeling; ontologies; organizational knowledge base; Model-Driven Engineering

I. Introduction

Ontologies are becoming popular to be an important component of information systems that supports business processes within and across organizations. Ontologies can give support to the system lifecycle: at modeling time, ontologies can be used to identify and describe key elements from business processes such as data, activities and profiles involved in the process itself (e.g. [1]); at development time, the structure of an ontology can be automatically translated into the source code of an information system by using an appropriate development support environment, as described for instance in [2] where business knowledge represented as OWL ontology is automatically translated into an information system implemented in the Mercury programming language. Hence, if changes of a business process are reflected in the ontology, the information system will also automatically reflect those changes. Finally, at run time, ontologies can add semantics to specify the behavior of business process [3], for instance, by using queries and reasoning to retrieve proper data for decision making or process validation (e.g., [4, 5, 6]).

Additionally, ontologies provide the basis for sharing and publishing organizational information through the Semantic Web [7].

However, representing organizational information directly into an ontology is not an easy task. A crucial step in building good quality ontologies is the right involvement of domain experts. As argued in [8, 9], traditional methodologies and tools are based on the idea that knowledge engineers drive the modeling process. This often creates an extra layer of indirectness which makes the task of producing and revising ontologies too rigid and complex, e.g., for the needs of business enterprises. Therefore, in this work we propose the use of specialized organizational modeling techniques as starting point for the capture and representation of organizational knowledge, thereby ensuring their proper definition. Then, we aim to provide a mechanism to automatically generate the corresponding ontological representation from the organizational model.

The *i** visual modeling language [10] is one of the most widely used organizational modeling techniques [11]. It supports the description of networks made up of social actors of an enterprise and the social intentional relationships and dependencies among them together with the representation of the internal behaviors required to satisfy actor dependencies. *i** provides the modeling basis to software engineering methodologies that support the early requirements elicitation stage such Tropos [12]. Therefore, we propose to start capturing the needed organizational knowledge with the *i** or Tropos modeling languages and automatically generate the corresponding ontological representation in the standard semantic web language Web Ontology Language (OWL) [13]. This is done to enable enterprise domain experts, with low knowledge engineering skills, to effectively represent organizational knowledge such as: strategy, structure, processes, and behavior, information and system requirements, in terms of ontologies.

In this paper, we present an approach based on Model Driven Engineering ideas that involves: a) the development of an ontology representing the metamodel of two widely used organizational modeling techniques *i** and Tropos and b) the systematic transformation of the knowledge represented in a specific *i** based model into instances of the ontology. Following this approach, we provide the automatic generation of an Organizational Knowledge Base (that we called Organizational KB), where OntoiStar embody the terminological knowledge (Tbox), that is, the knowledge about the terminology of the organizational domain, and OntoiStar instances represent the assertional knowledge (Abox), which is the knowledge coming from a specific organization description represented in an *i** based model.

As a result, and according to the organizational knowledge represented into an organizational model, we can provide: 1) organizational knowledge available to be exploited and consumed in the Semantic Web; 2) system requirements captured in the ontology for software development, such as agent systems; 3) proper data for decision making through ontology reasoning.

This paper is structured as follows: Section 2 presents the background in the organizational modeling domain. Section 3 provides the overview of our proposal. Section 4 describes the development of the proposed approach. Section 5 describes related work and finally, Section 6 concludes this work and summarizes our ongoing and future work.

II. Organizational Modeling

We have as goal to represent organizational knowledge in terms of ontologies. Therefore, we propose, as starting point, the use of specialized organizational modeling techniques to model the organization, thereby, ensuring the proper definition of the organizational knowledge. For this purpose, we have selected *i** [10] and Tropos [12].

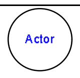
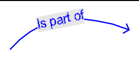
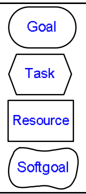
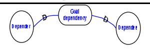
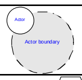
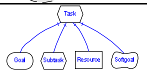
*i** supports the description of organizational networks made up of social actors who have freedom of action, but also depend on other actors to achieve their objectives and goals. It provides a visual language which includes two models: the *strategic dependency model*, a graph to represent social and intentional relationships (dependencies that describe an ‘agreement’) among the network of actors of an enterprise; and the *strategic rationale model*, a graph to describe and to support the internal behavior of each actor required to satisfy their dependencies on other actors. Examples of *i** primitives are presented in Table 1.

Tropos is an agent-oriented software methodology based on *i**. it provides a development process that is organized into five phases: *Early requirements*, to produce a model of the organization; *Late requirements*, to introduce the system-to-be in the model analyzing its impact in the organization; *Architectural design*, to obtain a representation of the architecture of the system in terms of subcomponents and the relationships among them; *Detailed design*, to define the software agent rationale, including capabilities and interaction specifications; and *Implementation*, which involves the production of code. The Tropos visual language uses the core concepts of *i** presented in Table 1 (with minimal differences omitted due to space).

Due to the growing interest around *i** [11], variants based on the original framework have been defined (such as Tropos and several more). Therefore, approaches for dealing the heterogeneity of the *i** variants have been proposed. We have analyzed two of these approaches [14, 15] as we aim to support the ontological representation of organizational knowledge represented not only with *i** and Tropos but also with other *i** variants. In [14] a metamodel is proposed where following a union approach the constructs of *i** and Tropos were included in the metamodel; in [15] a metamodel focused in *i**, Tropos and GRL is proposed where following an intersection approach the common constructs of the three variants were included in the metamodel; in [15] the iStarML specification language is proposed. iStarML is an XML interchange format which provides a common representational framework for *i** variants diagrams. It includes a set of tags corresponding to the core constructs of different *i** variants and a definition of attributes in each tag to represent particularities of the constructs (see Table 1, where attributes have been omitted due to space).

We have implemented the approach presented in this work, based on the metamodels and the iStarML format proposed in [14, 15].

TABLE I. *i** and Tropos constructs

***i** and Tropos core constructs**	**Modeling primitive**	**Types**	**iStarML tag**
Actor		None Role Position Agent	<actor>
Actor link		Is_a Is_part_of Occupies Covers Instance Plays	<actorLink>
Intentional element		Goal Softgoal Resource Task (*i**) Plan (Tropos)	<ielement>
Dependency (depender, dependum, dependee)		Goal Softgoal Resource Task (*i**) Plan (Tropos)	<dependency> <depender> <dependee>
Boundary			<boundary>
Intentional element link		Decomposition Means-End Contribution	<ielementLink>

III. Overview of the Proposal

In this section, we present the overview of our proposal, which is presented in Fig. 1. The phase 1 corresponds to the development of an OWL ontology, called OntoiStar, for the ontological representation of the metamodel of *i** and Tropos.

The phase 2 consists of the automatic generation of an Organizational KB by transforming the knowledge represented in a specific *i** based model into instances of the ontology OntoiStar. *Phase 1* has been divided in two processes. Process 1 is related to the analysis of *i** based metamodels [14, 15], which was addressed to determine the constructs to be represented in OntoiStar. Process 2 refers to the generation of OntoiStar, where constructs identified in the process 1 were manually mapped into OWL constructs following MDE ideas. *Phase 2* was also divided in two processes. Process 3 is related to the graphical representation of the organization with any of the organizational modeling technics *i** or Tropos, generating an *i** based model. This process can be realized with *i** modelers or editors that enables producing a model specified in iStarML [15], for instance jUCMNav[1] or HiME[2]. Process 4 refers to the automatic transformation of the *i** based model specified in iStarML to instances of the ontology OntoiStar. In order to support this process, we have developed a tool that implements transformation rules between the iStarML format and the ontology OntoiStar according to the MDE approach. The output of the tool corresponds to the ontology OntoiStar instantiated with the knowledge described in the *i** based model, namely, the Organizational KB.

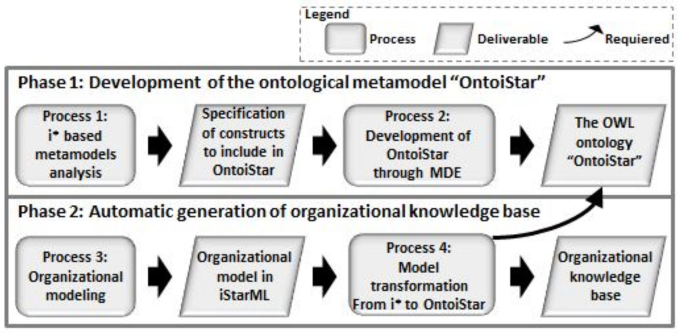

Legend
Process
Deliverable
Requiered
Phase 1: Development of the ontological metamodel "OntoiStar"
Process 1: i* based metamodels analysis
Specification of constructs to include in OntoiStar
Process 2: Development of OntoiStar through MDE
The OWL ontology "OntoiStar"
Phase 2: Automatic generation of organizational knowledge base
Process 3: Organizational modeling
Organizational model in iStarML
Process 4: Model transformation From i* to OntoiStar
Organizational knowledge base

Figure 1. Overview of the proposed approach

IV. ONTOLOGY GENERATION APPROACH

In this section, we describe our proposed approach to represent the organizational knowledge in terms of ontologies. Our approach starts from models described with the organizational modeling techniques *i** and Tropos to generate an Organizational KB in the ontology domain. The approach is based on MDE ideas. MDE is a methodology which focuses on creating and exploiting domain models for the software development. It is based on layered architectures, where models, metamodels and metametamodels correspond to the M1, M2 and M3 layers, respectively. Fig. 2 shows the layered architectures of the domains that we are addressing. On the left side, it is found the layered architecture of *i** based modeling languages, and on the right side, it is found the layered architecture that we have proposed for the ontology domain. Transformation bridges [16] can be defined to move from a layered architecture to another. A transformation bridge comprises a set of transformation rules which together describe how a model conforming to the source metamodel can be transformed into a model conforming to the target metamodel. A transformation rule defines how one or more constructs in the source metamodel can be transformed into one or more constructs in the target metamodel. A transformation bridge is defined in two steps:

1) Constructs in each metamodel are identified.
2) The relationships between the constructs of both metamodels are analyzed and specified, i.e. transformation rules are defined.

A transformation bridge can be defined at the level of metamodels (M2) as well as the level of metametamodels (M3). Thus, a transformation bridge defined at level Mn can then be used to automate model to model transformation at level Mn-1.

In our approach, we have defined two transformation bridges:

M_3 transformation bridge, which has been defined in M3 layer to generate the OWL ontology OntoiStar. Therefore, it contains the transformation rules between concepts from the *i** metametamodel (represented in the Unified Modeling Language, such classes and associations) in the *i** layered architecture and concepts from the OWL metamodel (such classes and properties) in the OWL ontology architecture.

M_2 transformation bridge, which has been defined in M2 layer to transform any *i** based model into instances of the ontology OntoiStar. It contains the transformation rules between concepts from the *i** metamodel (represented in iStarML) in the *i** layered architecture and concepts from the ontology OntoiStar in the OWL ontology architecture. We have developed a tool that implements the transformation rules of this bridge in order to automate the transformation process of any *i** based models.

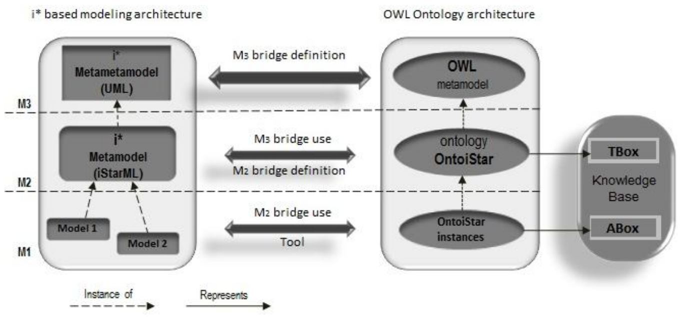


Figure 2. Arquitectural solution of the approach

The implementation of the proposed approach is described in the following subsections.

A. *Ontological metamodel development phase*

In this subsection, we describe the phase 1 of our proposed approach, that is, the development of the ontology OntoiStar which represent the ontological metamodel of *i** based modeling languages.

Process 1 was carried out in order to determine the constructs of the *i** based modeling languages to be included into OntoiStar. It consisted of an analysis of two metamodels that address the integration of several *i** variants [14, 15], as we mentioned in section II. The metamodel proposed in [14] includes all the elements of *i** and Tropos; the metamodel proposed in [15] includes only the common concepts of the

1 The jUCMNav website. http://jucmnav.softwareengineering.ca/ucm/bin/view/ProjetSEG/
2 The HiME website. http://www.lsi.upc.edu/ llopez/hime/

variants *i**, Tropos and GRL. The main differences between these metamodels lie in concepts not common, the representation of concepts relationships, the class hierarchy and the class properties. For the development of OntoiStar, we have adopted the common characteristics of both metamodels including concepts, relations and attributes, but also some specific characteristics of each one. From [14] we have adopted: a) super classes to define a class hierarchy between constructs (see *i** and Tropos core concepts in Table 1); b) enumeration classes to represent specific attributes such as contribution types: positive and negative; c) the representation of all concept relationships in terms of classes and associations. From [15] we have adopted the most classes' properties, such as, disjoint and complete. As a result of the analysis we have generated an *i** metamodel including the adopted characteristics. The resultant *i** metamodel is the basis to generate the ontology OntoiStar.

Process 2 was carried out in order to generate the ontology OntoiStar. OntoiStar has been generated by applying the *M_3 transformation bridge* (from i* metametamodel to OWL metamodel), which is defined as follows:

(1) Constructs from the *i** metametamodel and from the OWL metamodel are identified.
The *i** metametamodel is described with UML therefore, the relevant constructs to consider for the transformation are: class, attribute, association and class property.

The significant constructs of the OWL language to develop OntoiStar are: OWL Class, Object property and Data property; the axioms: subClassOf, ObjectPropertyDomain, ObjectPropertyRange, DataPropertyDomain, disjointWith and unionOf.

(2) Relationships between constructs from the *i** metametamodel and the OWL metamodel are analyzed and specified. The following transformation rules were proposed:
Rule 1: Classes and class associations from *i** metametamodel as classes in OWL.
Rule 2: Associations from *i** metametamodel as object properties in OWL.
Rule 3: Class properties from *i** metametamodel as axiom class properties in OWL.
Rule 4: Enumeration elements from *i** metametamodel as class instances in OWL.
Rule 5: Attributes (we define two kinds of attribute representation):
(a) Class attributes from *i** metametamodel as data properties in OWL.
(b) Attributes (of enumeration class) from *i** metametamodel as object properties in OWL.

The *M_3 transformation bridge* is manually applied in layer M2, transforming the *i** metamodel into its ontological representation: OntoiStar. Table 2 presents partially results of the application of Rule 1, the OWL class representation of classes and class associations of the *i** metamodel.

An example of application of Rule 2 is the actor is_a link. It represents the relationship between two actors. Therefore, the actor class "Actor" and a class to represent the is_a link "isALink" have been generated following Rule 1. Then two object properties have been defined to represent the source and target associations between the isALink class and the Actor class.

```
<owl:ObjectProperty rdf:ID="has_Actor_IsALink_source_ref">
  <rdfs:range rdf:resource="#Actor"/>
  <rdfs:domain  rdf:resource="#IsALink"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="has_Actor_IsALink_target_ref">
  <rdfs:domain rdf:resource="#IsALink"/>
  <rdfs:range rdf:resource="#Actor"/>
</owl:ObjectProperty>
```

After applying the transformation rules to all the elements of our proposed *i** metamodel, we obtain the ontological metamodel OntoiStar. Thus, we obtain the Tbox part of the Organizational KB.

TABLE II. M3 TRANSFORMATION BRIDGE

***i** and Tropos core constructs**	**Types**	**OWL construct**
Actor	None	<owl:Class rdf:about="#Actor"/>
	Role	<owl:Class rdf:ID="Role"/>
	Position	<owl:Class rdf:ID="Position"/>
	Agent	<owl:Class rdf:ID="Agent"/>
Actor link	Is_a	<owl:Class rdf:about="#IsALink"/>
	Is_part_of	<owl:Class rdf:about="#IsPartOfLink"/>
	Occupies	<owl:Class rdf:about="#OccupiesLink"/>
Intentional element	Goal	<owl:Class rdf:ID="Goal"/>
	Softgoal	<owl:Class rdf:ID="SoftGoal"/>
	Resource	<owl:Class rdf:ID="Resource"/>
	Task (*i**)	<owl:Class rdf:ID="Task"/>
	Plan (Tropos)	<owl:Class rdf:ID="Plan"/>
Dependency (depender, dependum, dependee)	Goal Softgoal Resource Task (*i**) Plan (Tropos)	<owl:Class rdf:ID="Dependency"/> <owl:Class rdf:ID="DependeeLink"/> <owl:Class rdf:ID="DependumLink"/> <owl:Class rdf:ID="DependerLink"/>
Boundary		<owl:Class rdf:ID="ActorBoundary"/>
Intentional element link	Decomposition	<owl:Class rdf:ID="DecompositionLink"/>
	Means-End	<owl:Class rdf:ID="MeansEndLink"/>
	Contribution	<owl:Class rdf:ID="ContributionLink"/>

B. *Organizational Knowledge Base generation phase*

In this subsection, we describe the phase 2 of our proposed approach, that is, the automatic generation of the Abox part of the Organizational KB. First, we present a tool that supports the automatic transformation of the knowledge represented in a specific *i** based model into instances of the ontology OntoiStar. Then, we describe the processes of this phase, which must be performed whenever that it is desired to represent the organizational knowledge of a specific organization in terms of ontologies.

The tool is called TAGOOn – (**T**ool for the **A**utomatic **G**eneration of **O**rganizational **O**ntologies). It is based on MDE ideas. Therefore, through a set of transformation rules implemented in the tool, it automatically populates the ontology OntoiStar with instances that represent the *i** elements belonging to a specific organizational model. Thus, generating the Abox part of the Organizational KB.

OntoiStar instantiation is carried out by implementing in TAGOOn the *M_2 transformation bridge* (from i* metamodel, described in the iStarML specification, to OntoiStar) as shown in Fig. 3. Transformation rules between iStarML constructs (see Table 1) and the ontology OntoiStar has been defined. The transformation rules are automatically applied to *i** based models on layer M1. In this way, TAGOOn supports the automatic transformation of any *i** based model into instances of the ontology OntoiStar.

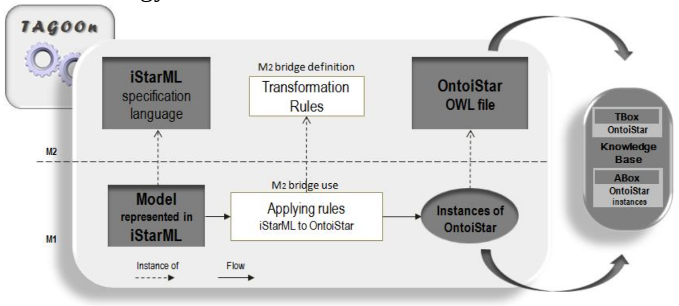


Figure 3. *M_2 transformation bridge* implemented in TAGOOn

Following we describe the Organizational KB generation process flow (Fig. 4). Process 3 consists in modeling the organization by using the visual *i** or Tropos modeling languages. This model can be realized with *i** modelers or editors that enables producing a model specified in the iStarML format. The model in iStarML is the input for the process 4, which is performed by the tool TAGOOn. The tool then parses the iStarML file, and according to the defined transformation rules, instantiate the corresponding classes and properties in the ontology OntoiStar. The output of the tool is the ontology OntoiStar with instances that represent the knowledge content in the *i** based model. The ontology OntoiStar with their instances shapes an Organizational KB in which is possible to apply services offered by the ontology technology such as reasoning and querying. The output can be edited with an ontology editor, for modifying the ontology or its instances or it can be the input of development or reasoning platforms supported by ontologies.

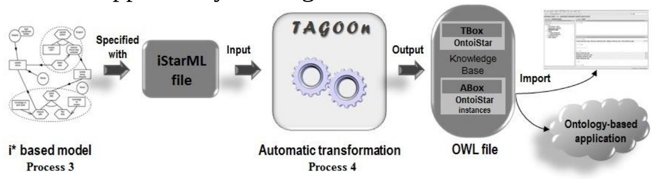


Figure 4. Organizational KB generation process flow

TAGOOn has been validated with a case study carried out in [17]. It consists of a real project to model the processes of a postgraduate institution (www.cenidet.edu.mx) that offers Master and PhD programs. We present a fragment of the process to register students in the academic semesters. The registration process involves: 14 actors, 43 actor dependencies, 117 intentional elements and 44 intentional element links. The fragment is related with the actors "Student" and "Thesis advisor" and their Softgoal dependency "Choose appropriated courses" and their goal dependency "Choose courses" (Fig. 5). Table III presents parts of the iStarML file and the resultant OWL file.

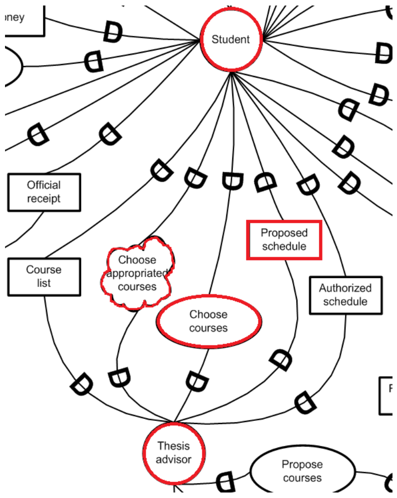


Figure 5. Fragment of a *i** model and its iStarML

TABLE III. FRAGMENTS OF THE ISTARML AND THE OWL FILES

Fragments of the iStarML file	Fragments of the OWL file
<actor id="05" name="Student"/> <actor id="06" name="Thesis advisor"/> <ielement id="01" name="Choose appropriated courses" type="Softgoal"> <dependency> <depender aref="05"/> <dependee aref="06"/> </dependency> </ielement> <ielement id="02" name="Choose courses" type="Goal">	<OntoiStar:Actor rdf:ID="Student"/> <OntoiStar:Actor rdf:ID="Thesis_advisor"/> <OntoiStar:Softgoal rdf:ID="Choose appropriated courses"/> <OntoiStar:Goal rdf:ID="Choose courses"/> <OntoiStar:Resource rdf:ID="Proposed schedule"/>

V. RELATED WORK

Solutions to the problem of modeling in various aspects of an enterprise were proposed in several works, both in terms of definition of the metamodel and in terms of methodologies to support the creation of the model itself. Concerning the metamodel the TOVE Ontology Project [18] proposes a set of integrated ontologies for the modeling of an enterprise which spans several aspects of an enterprise, such as activities, states, resources, time, and so on. The Common KADS model set [19] is a collection of models (organization, task, agent) for structuring knowledge in an organization. The organizational ontology [7] is a core ontology to represent organizational structures developed with the objective of supporting linked data publishing of organizational information across different domains. In [20], a set of ontologies are proposed to support business process integration. Focusing on methodologies for ontology / model creation, we can notice that most of them - e.g., TOVE Enterprise methodology [18], CommonKADS [19], Methontology [21] and the Enterprise ontology [22] - are built around the knowledge engineer, who executes and coordinates all the different phases of the knowledge acquisition and formalization process. The novelty of our approach w.r.t. these proposals lies in using a Model Driven

Engineering ideas to represent organizational knowledge in terms of ontologies from specialized techniques for modeling organizations, where concepts are familiar to enterprise domain experts such as a) the representation of social and intentional relationships among the network of actors of an enterprise, and b) the representation of the internal behaviors required to satisfy actor dependencies. The approach is also complemented by an intuitive visual representation language which can facilitate the involvement of enterprise domain experts in the modeling process. The ontology generation is managed in a transparent manner for the user.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented a semi-automated approach to generate ontologies from an organizational model described with *i** or Tropos modeling languages. Specifically, it describes how the ontological metamodel of *i** based modeling languages (OntoiStar) has been developed, and it also explains the transformation process that can be applied to automatically populate OntoiStar with instances of *i** elements belonging to a specific organizational model. Thus, providing an Organizational KB where OntoiStar represent the Tbox and OntoiStar instances, represent the Abox.

Services offered by the ontology technology such as reasoning and querying can be applied to this Organizational KB. Furthermore, it can be opened and edited with an ontology editor, or it can be the input of development or reasoning platforms supported by ontologies. As the organizational knowledge is represented in the standard Semantic Web language OWL, it could be available to be exploited and consumed in the Semantic Web by paradigms such as Linked Data.

Although, we considered the case of *i** notation, the approach can be generalized to other organizational modeling frameworks, since it follows MDE ideas.

In our ongoing work, we are currently addressing the integration of other *i** variants to OntoiStar, thereby the ontology will be useful for any *i** variant. Moreover, we are consolidating the tool to support the automated generation of an Organizational KB through an *i** based model described with any of the variants integrated in OntoiStar.

REFERENCES

[1] A. Prieto and A. Lozano-Tello. Use of ontologies as representation support of workflows oriented to administrative management. Journal of Network and Systems Management, 17(3):309–325, 2009.

[2] M. V, E. Bossche, P. Ross, I. Maclarty, B. V. Nuffelen, and N. Pelov. Ontology driven software engineering for real life applications, 2007.

[3] I. Weber, J. Hoffmann, and J. Mendling. Beyond soundness: on the verification of semantic business process models. Distributed and Parallel Databases, 27(3):271–343, 2010.

[4] M. Dimitrov, A. Simov, S. Stein, and M. Konstantinov. A bpmo based semantic business process modelling environment. In Semantic Business Process and Product Lifecycle Management, volume 251 of CEUR, 2007.

[5] G. Gröner and S. Staab. Modeling and query patterns for process retrieval in owl. In 8th International Semantic Web Conference (ISWC 2009), volume 5823 of LNCS, pages 243–259, Springer, Washington, DC, 2009.

[6] C. Ghidini, C. D. Francescomarino, M. Rospocher, P. Tonella, and L. Serafini. Semantics based aspect oriented management of exceptional flows in business processes. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2011.

[7] D. Reynolds. An organizational ontology. W3C, https://dvcs.w3.org/hg/gld/raw-file/default/org/index.html, March, 2013.

[8] V. Dimitrova, R. Denaux, G. Hart, C. Dolbear, I. Holt, and A. G. Cohn. Involving domain experts in authoring owl ontologies. In Proceedings of the 7th Int. Semantic Web Conference (ISWC 2008), volume 5318/2010 of LNCS, pages 1–16. Springer Berlin / Heidelberg, Karlsruhe, Germany, 2008.

[9] C. Ghidini, M. Rospocher, and L. Serafini. Moki: a wiki-based conceptual modeling tool. In ISWC 2010 Posters & Demonstrations Track: Collected Abstracts, volume 658 of CEUR Workshop Proceedings (CEUR-WS.org), pages 77– 80, Shanghai, China, 2010.

[10] E. S.-K. Yu. Modelling strategic relationships for process reengineering. PhD thesis, Toronto, Ont., Canada, Canada, 1996.

[11] X. Franch. The *i** framework: The way ahead. In Sixth International Conference on Research Challenges in Information Science RCIS'12, pages 1–3, Paris, France, 2012.

[12] P. Giorgini, J. Mylopoulos, A. Perini, and A. Susi. The Tropos Methodology and Software Development Environment. In E. Yu, P. Giorgini, N. Maiden, and J. M. eds., editors, Social Modeling for Requirements Engineering, pages 405–423, Chapter 11, MIT Press, Cambridge, MA, 2010.

[13] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL Web Ontology Language Reference. Technical report, W3C, http://www.w3.org/TR/owl-ref/, February 2004.

[14] M. Lucena, E. Santos, C. T. L. L. Silva, F. M. R. Alencar, M. J. Silva, and J. Castro. Towards a unified metamodel for i*. In Second International Conference on Research Challenges in Information Science RCIS'08, pages 237–246, Marrakech, Morocco, 2008.

[15] C. Cares, X. Franch, A. Perini, and A. Susi. Towards interoperability of *i** models using istarml. Computer Standards & Interfaces, 33(1):69{79, 2011.

[16] S. Staab, T. Walter, G. Gr¨oner, and F. S. Parreiras. Model driven engineering with ontology technologies. In Reasoning Web, pages 62–98, 2010.

[17] H. Estrada. A service-oriented approach for the *i** framework. PhD thesis, Valencia University of Technology, Valencia, Spain, 2008.

[18] M. S. Fox and M. Gr¨uninger. Enterprise modeling. AI Magazine, 19(3):109–121, 1998.

[19] G. Schreiber, H. Akkermans, A. Anjewierden, R. Dehoog, N. Shadbolt, W. Vandevelde, and B. Wielinga. Knowledge Engineering and Management: The CommonKADS Methodology. The MIT Press, December 1999.

[20] M. Grüninger, K. Atefi and M. Fox. Ontologies to Support Process Integration in Enterprise Engineering. In: Computational & Mathematical Organization Theory vol 6, n 4, pp. 381-394, 2000.

[21] M. Fernandez-Lopez, A. Gomez-Perez, and N. Juristo. Methontology: from ontological art towards ontological engineering. In Proc. Symp. on Ontological Eng. Of AAAI, Providence, Rhode Island, 1997.

[22] J. L.G. Dietz. Enterprise Ontology. Springer, Berlin / Heidelberg, 2006.