EDON : A Method for Building an Ontology as Software Artefact

Emiliano Reynares[1], Ma. Laura Caliusco[1], and Ma. Rosa Galli[1,2]

[1] CIDISI Research Center, UTN Santa Fe, B. Lavaysse 610, S3004EWB Santa Fe, Argentina
{ereynares,mcaliusc}@frsf.utn.edu.ar
[2] INGAR-UTN-CONICET, Avellaneda 3657, S3002GJC Santa Fe, Argentina
{mrgalli}@santafe-conicet.gov.ar

Abstract. The current dynamics of organizations produces frequently changing business rules, involving changes into the software applications that embed them. The use of ontologies as software artefacts intended to encapsulate business rules is a mean to raise the flexibility, extensibility and ease of maintenance of the software applications. Such ontologies should be developed and maintained in conjunction with other software components, so an ontology building methodology must be considered in the context of the software development process. This paper presents an evolutionary method for building ontologies intended to be used as a structural conceptual model of an information system, encoding business rules in a declarative way and enabling the intertwining of ontology and software development processes.

Keywords: ontology building, software artefact, conceptual dynamics, business rule, evolutionary development, software development

1 Introduction

In general, the business rules of an organization are embedded in the procedural part of a software application, spreading them across all the software system. Then, changes in business rules involve an intensive task of software maintenance, which requires the discovering and modification of the part of the source code relevant to each change. Since organizations are facing frequently changing business environments, there is a need for technologies supporting a quick propagation of new business rules into the software systems as a mean to raise its flexibility, extensibility and ease of maintenance.
Encapsulating the business rules in a specific system component is an approach to solve the previous problem. Rules bases can be used in such a way, but in this approach the use of different attributes to represent the same thing, and the definition of rules based on incompatible notions of a concept are issues that could affect the rule interaction [?]. Ontologies, as artefacts intended to explicitly represent the data semantics, could tackle both issues: encapsulating the

business rules while avoiding the concept misunderstanding [?] [?].
To develop an ontology intended to be used as a structural conceptual model of an information system, the ontology building process must be considered in the context of the software development project as a way to enable the assessment and integration of ontology and software code. Ontology building methodologies can be grouped in two main approaches. The former involves the best practices from the Knowledge Engineering field [?]. But such practices usually do not form part of the toolbox involved in the development of software applications. The second group of methodologies is based on practices from the Software Engineering field. In this group, some proposals stems their characteristics from traditional software development processes, taking advantage of the large experience drawn from widely used standards in software engineering [?] [?]. However, this proposals remains conceptual dynamics as an open issue. Conceptual dynamics [?], i.e., new conceptual elements arise as some old ones becomes irrelevant, is specially relevant for organizations operating in frequently changing business environments. Approaches applying the main principles of Extreme Programming (XP), a widely used agile software methodology [?], have emerged as a response to this issue [?] [?] [?] [?]. Such approaches stress the importance of applying a change when it is necessary, without distinguishing a set of phases during the performing of development activities. Although it allows tackling the conceptual dynamics issue, the lack of a phase model difficult its application because it does not provide a descriptive path to quickly and smoothly evolve through further extended versions of the ontology.
The contribution of this paper is a method, named Evolutionary Development of ONtologies (EDON), which pursues three goals:

1. To develop from scratch an ontology intended to encapsulate the declarative specification of business rules, supporting accurate and well-defined requirements of a software system.
2. To enable the intertwining of ontology and software development process, facilitating a quick integration of both artefacts.
3. To allow the ontology readily reflect the changes in quickly evolving domains by providing a phase model to evolve through successive versions of the ontology.

This paper is organized as follows. Section 2 presents an overview of EDON. Section 3 describes EDON in detail. Section 4 presents discussions and future research directions.

2 EDON: Evolutionary Development of ONtologies

2.1 The EDON approach

The main objective of EDON is to develop from scratch an ontology intended to be used as a structural conceptual model of an information system, encoding business rules in a declarative way. EDON adopts a requirement driven, iterative, and incremental approach. Requirement driven because it allows the gradual evolution from the identification and prioritization of the functional requirements to be supported toward a computable ontology. Iterative and incremental because the ontology is developed through cycles, each one producing a further extended ontology. The iterations are short time-boxed, enabling the produced ontology to rapidly reflect the changes in requirements and conceptual dynamics, reducing the maintenance bottleneck.

2.2 The roles involved in EDON

EDON is collaboratively performed by Domain Experts (DEs) and Knowledge Engineers (KEs). DEs interact daily with the problem domain, so they have the knowledge to be modelled. KEs have the know-how for representing the reality in a way that supports the requirements of a software application. To facilitate the communication between these profiles, the modelling activities are performed at a high level of abstraction. Furthermore, short development iterations allow KEs and DEs to take advantage of what was learned during the building of earlier versions of the ontology. Learning comes from both the building and the use of the ontology, iteratively improving those versions.

2.3 EDON in the context of software development methodology

EDON has been conceived to be intertwined with a iterative and incremental software development process, as shown in Figure 1.

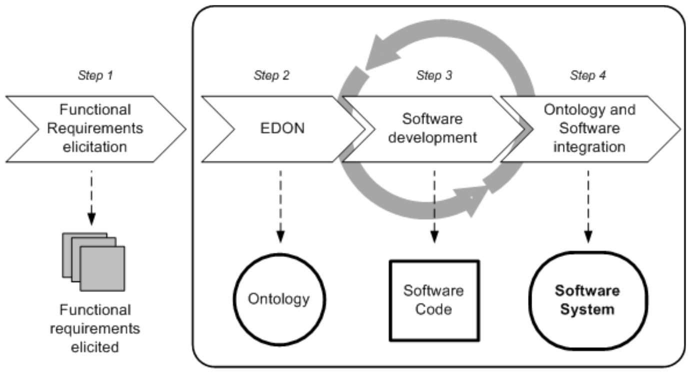
Step 1
Functional Requirements elicitation
Functional requirements elicited
Step 2
EDON
Ontology
Step 3
Software development
Software Code
Step 4
Ontology and Software integration
Software System

Fig. 1. EDON in the context of software development process

Using an ontology to the declarative specification of business rules is an architectural decision that falls outside the scope of EDON. EDON performing starts when the technical people responsible for the overall software project have taken the decision of using an ontology in such a way. Hence, EDON starts after the requirements elicitation activities have been performed and the core architectural decisions have been taken (Step 1). Each performing of the EDON process produces an ontology (Step 2) which is followed by the implementation of the subset of functional requirements supported for such ontology (Step 3). Finally, the ontology and the software code are integrated to generate a specific version of the software system (Step 4). Due to the iterative and incremental nature of EDON, each version of the ontology and software is integrated cyclically.

2.4 The EDON Processes

The overall approach is shown in Figure 2, and depicted as follows:

1. Selecting a subset of functional requirements of the software system not yet supported by the ontology.
2. Implementing the ontology that supports the selected requirements.
3. Interrelating the produced ontology with the ontology obtained in the previous iteration.

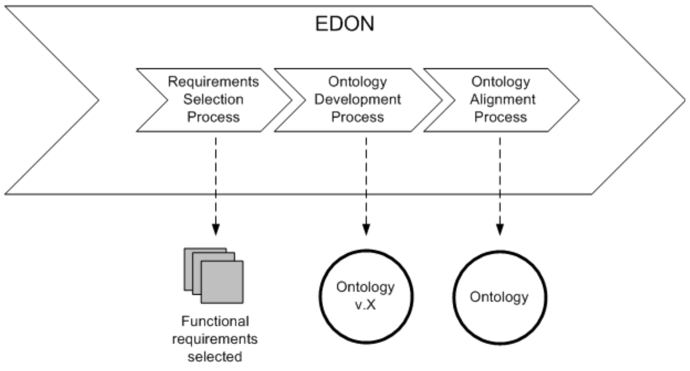
EDON
Requirements Selection Process
Ontology Development Process
Ontology Alignment Process
Functional requirements selected
Ontology v.X
Ontology

Fig. 2. EDON processes

Requirements Selection Three activities are performed on the requirements selection process (Figure 3): (1) *requirements identification*, (2) *domain entities identification and prioritization* and (3) *requirements grouping and selection.* The former involves the identification of the functional requirements that involves business rules in their meeting. Such requirements will be supported by the ontology. The second one consists in the identification and prioritization of the domain entities involved in the meeting of the requirements identified before. The main goal of this activity is to identify the small set of entities that usually form the core of the domain. The third activity consists in grouping and selecting the requirements according to the importance of the entities involved. The aim is to select for further development the group of requirements that involves the core entities of the domain. These activities are mainly performed by KEs but requires the agree of DEs to trace the guidelines for building the ontology.

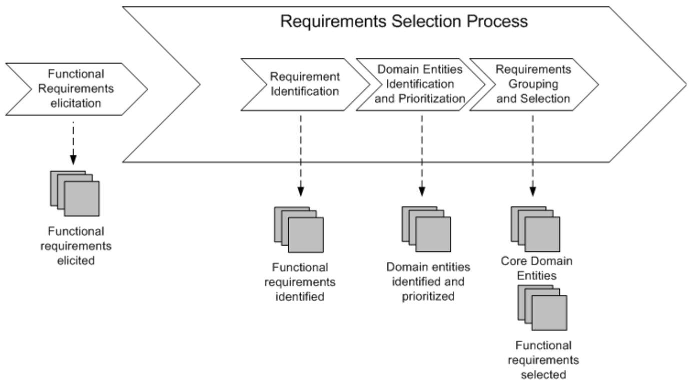
Requirements Selection Process
Functional Requirements elicitation
Requirement Identification
Domain Entities Identification and Prioritization
Requirements Grouping and Selection
Functional requirements elicited
Functional requirements identified
Domain entities identified and prioritized
Core Domain Entities
Functional requirements selected

Fig. 3. Requirements Selection process

Ontology Development This process involves Development Activities - *specification, conceptualization, formalization, refinement, implementation and alignment* - and Support Activities - *knowledge elicitation and evaluation.* This activity classification is based on the Methontology Framework and the techniques to carry out them are based on the different methodologies and good practices for building ontologies that have been developed since mid-1990 [**?**]. However, EDON considers the performing of the refinement activity with the aim of extending the ontology by focusing on the declarative formulation of business rules. Figure 4 depicts the overall process.
The performing of Development Activities allows to evolve from an abstract model toward an computable ontology. Support Activities are carried out along the whole development process and its importance depend on the nature of the activity being performed, i.e., the knowledge elicitation is especially relevant in the conceptualization and refinement activities while the importance of evaluation increases along the development. Ontology evaluation comprises: (1) ontology verification that deals with making sure that its definitions implement the requirements correctly, and (2) ontology validation that refers to whether the meaning of the definitions really models the real world for which the ontology was created [**?**]. At the end of the building process, ontology verification is stressed through a systematic testing process.

Ontology Alignment This process is needed due to the EDON incremental nature and consists of: (1) *alignment* and (2) *alignment evaluation* (Fig. 4).

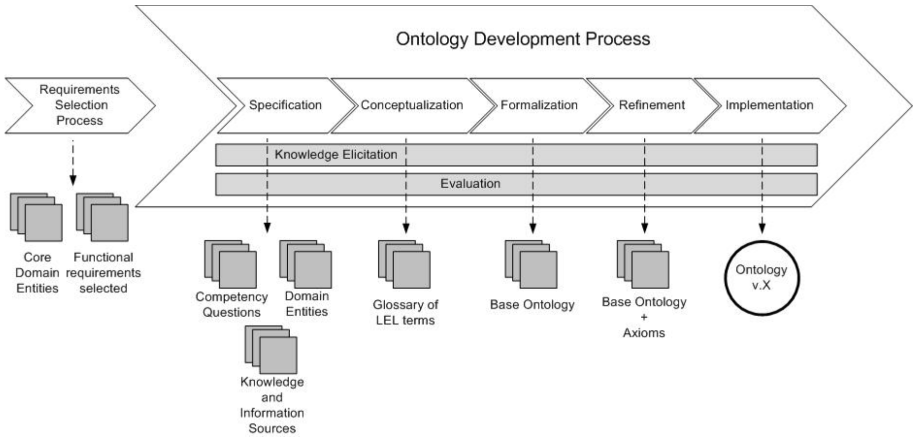

Fig. 4. Ontology Development process

Each application of EDON produces an ontology that supports a disjoint set of functional requirements, i.e., those selected on the specification activity of the iteration. Therefore, the alignment of current and previous version of the ontology is needed as a way to support both set of requirements. Ontology alignment is the process of determining the different types of (inter-ontology) relationships among their terms [**?**] [**?**]. As a result, a new ontology composed by sub-ontologies is created. DEs are responsible for stablish the most adequate relationships, given they have such kind of knowledge.
Alignment evaluation activity is performed on the aligned ontology with the aim of testing its quality. In the case the required quality cannot be reached, the relationships established in the alignment activity will need to be revised and reformulated.

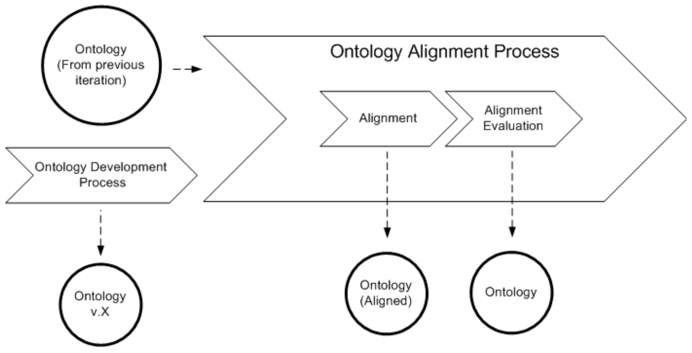
Ontology (From previous iteration)
Ontology Alignment Process
Alignment
Alignment Evaluation
Ontology Development Process
Ontology v.X
Ontology (Aligned)
Ontology

Fig. 5. Ontology Alignment process

3 EDON Method

The processes that compose the EDON method are depicted by supporting the development of a case study, as shown in the following subsections. EDON has been used to build an ontology intended to meet the requirements of a software application named Research Project Management Solution (RPMS). RPMS is used at the CIDISI research center to support management activities taking place when the administrative staff interacts with projects, researchers, and funding agencies. Examples of this activities: loading a research project pending approval, managing evaluation reports of research projects and requesting a progress report for an approved research project. Two roles were involved in the activities performing: DEs came from the CIDISI administrative staff while the KEs were the authors of this work. The software development was performed by an independent team using the SCRUM[3] agile software development framework.

3.1 Requirements Selection

The functional requirements of RPMS were selected considering two issues: (1) the developed ontology should provide ontology-based reasoning over the business rules of the application domain and (2) human users of the RPMS should not interact directly with the underlying ontology but through the software application. Following, the *project*,*researcher* and *funding agency* were considered the core entities of the domain as result of the domain entity identification and prioritization activity performing. Finally, the subset of functional requirements mainly involved with such entities were selected for further development. A storyboard exposing a functional requirement belonging to the selected subset is:
The software system should allow a researcher to upload a research project pending approval. During this upload, the software system must validate the input data against the rules defined by the agency intended to funding the project.

3.2 Ontology Development

Ontology Specification Once selected the functional requirements to be supported, EDON proposes to stem competency questions from such requirements. Competency questions (CQs) are questions at a conceptual level, informally written in natural language, that the ontology should be able to answer [?]. CQs allow identifying remaining entities - less important than the core entities identified in the previous process - that are required to support the functional requirements. CQs will be also used to evaluate the developed ontology. An excerpt of CQs and entities stemmed from preceding storyboard is shown in Table 1 and Table 2, respectively.
Knowledge and information sources need to be identified for further development of the ontology. The sources can be grouped into three categories: (1)

[3] Scrum overview in http://www.scrumalliance.org/pages/what_is_scrum

Table 1. An excerpt of CQs the ontology should be able to answer

CQ1	What are the requirements a research project must satisfy to be accepted for a funding agency?
CQ2	What are the requirements the manager of a research project must satisfy?
CQ3	What are the requirements a member of a research project must satisfy?
CQ4	What are the documents evaluated for a funding agency?

Table 2. An excerpt of the entities required for supporting a functional requirement

Research Project	Researcher	Funding Agency
Project Member	Project Manager	Evaluation Documents

structured, like databases, thesauri and ontologies; (2) *semi-structured*, such as technical reports, forms and XML documents, and (3) *non-structured*, like manuals, standards, and DEs. Such grouping allows the identification of techniques and tools that can be used for knowledge mining. The availability and need for specific knowledge sources are strongly related to the application domain and the software project. In the study case, five knowledge resources were identified. Two taxonomies as structured information sources, one of them detailing the established hierarchy of scientific disciplines and themes, and the other one depicting the cost constraints that a research project must meet. Two documents as non-structured information sources, stating policies and constraints for projects proposals and researchers. The most important source was the CIDISI administrative staff - the project DEs - who had the expertise to be aware of the implicit policies that governed their work.

Ontology Conceptualization This activity refers to the representation of the knowledge associated to the domain entities. The knowledge is collected from the information sources and its representation is done independently of the modelling paradigm and the implementation language of the target ontology, by using the Lexicon Extended Language (LEL) [?]. LEL is a representation of the terminology in the application language, which is classified in four categories: *object*, *subject*, *verb*, and *state*. Each term is described in two ways. The former, *notion*, is the denotation of the term indicating its meaning, who is, when it occurs, which process involves. The latter, *behavioural response*, describes the connotation of the term indicating the effects that such term generates on others terms and the effects that other terms generate on it. This activity generates a glossary of LEL terms, which should be built, validated and committed by both DEs and KEs.
Interviews and brainstorming with DEs were the main elicitation techniques used in the study case, supplemented by an analytical review of the documents stating policies and constraints for project proposals and researchers. An excerpt of a glossary entry is shown in Table 3.

Ontology Formalization The formalization activity generates a base ontology by performing a systematic process detailed in [?]. The process gives an ontology structure to the terminology described in the conceptualization activity. Such structure is independent of any ontology implementation language but can be mapped into most of them, and it is defined as follows [?]:

Definition 1. *An ontology is a 5-uple O:= $\{\mathcal{C}, \mathcal{R}, \mathcal{H}, rel, \mathcal{A}\}$ where:*

- *Two disjoint sets, $\mathcal{C}$ (concepts) and $\mathcal{R}$ (relations).*
- *A concept hierarchy, a directed relation $\mathcal{H} \subseteq \mathcal{C}\ x\ \mathcal{C}$ which is called concept hierarchy or taxonomy. So, $\mathcal{H}(\mathcal{C}1, \mathcal{C}2)$ means $\mathcal{C}1$ is a sub-concept of $\mathcal{C}2$.*
- *A function rel: $\mathcal{R} \rightarrow \mathcal{C}\ x\ \mathcal{C}$ that relates the concepts non taxonomically.*
- *A set of ontology axioms $\mathcal{A}$ expressed in appropriate logical language.*

Axioms and business rules have been distinguished in some works, i.e., axioms as logical sentences expressing model properties that are always true while business rules as logical sentences used to express the characteristics of a domain [?]. For the sake of simplicity, this article will use the term *axiom* to refer both concepts. An excerpt of the base ontology produced in the study case is shown in Table 4. For example, concept properties have been represented by means of relations, e.g., *HasFullName*, *HasCategory*, *WorkIn*, *HasCode*, etc. Some axioms, e.g., the manager's category, its minimal dedication to a project are also shown.

Table 3. The Project Manager glossary entry

Project Manager *(subject LEL type)*
Notion
Itis a Researcher at Category I or II, working in an Execution Unit and managing, at most, two Research Projects. Each Research Project requires, at least, twenty hours-month dedication.
Behavioural response
To develop research projects
To raise funds for the project execution
To generate reports on project status

Ontology Refinement The refinement activity consists in further extending the base ontology by focusing on the formulation of axioms, which are obtained from the knowledge and information sources identified in the specification activity. Insights of KEs and DEs are specially important in this stage. DEs collaborate with the aim of producing the tightest possible model of reality while KEs are focused on supporting a set of functional requirements by means of an ontology.
Regarding to the study case, some axioms identified trough the refinement activity performing are depicted in Table 5.

Table 4. An excerpt of the base ontology

Concepts	**Relations** *(Related Concepts)*	**Axioms**
Research Project	HasCode HasTitle HasTheme *(Theme)* HasManager *(Researcher)* HasManagerDedication HasReport *(Report)*	The manager is a researcher at category I or II. The manager must dedicate, at least, twenty hours-month at each managed project.
Researcher	HasFullName HasCategory *(Category)* WorkIn *(Execution Unit)* ResponsibleFor *(Research Project)*	It can be the manager of, at most, two research projects.

Table 5. Some axioms extending the base ontology

Axiom
If a project report has a 30 day delay, the research project is cancelled
If a project report has a 30 day delay, the project manager is suspended
If a project report is rejected, the research project is cancelled

Ontology Implementation This activity encodes the ontology in an implementation language, producing a computable ontology which should support the functional requirements selected in the specification. The expressive power, computational complexity of the reasoning method, functional requirements to be supported, and software application execution environment should be considered to select the implementation language. A description of the most common languages and the selection criteria can be found in [?]. KEs are responsible for the activity performing given its technical character.
The ontology of the study case was implemented using Protégé - a free and open source ontology editor - and Pellet - an inference engine that provides sound-and-complete OWL-DL reasoning services -[4]. The ontology was written in the OWL-DL 1.0 ontology language and serialized in OWL/RDF format[5]. The axioms overcoming the expressive power of OWL-DL were modelled by SWRL language[6]. Two axioms on this language are shown in Table 6. The former express that if a project report is rejected, the research project is cancelled. The latter express that a manager must dedicate, at least, twenty hours-month at each managed project.
As stated in Section 2.4, evaluation is a Support Activity whose importance increases along development stages, stressing ontology verification at the end of the building process. Several works have stated the quality of an ontology is a multidimensional feature [?] [?]. We consider three dimensions must be evaluated:

[4] Support, downloads and documentation about the integration of Protégé editor and Pellet inference engine can be found in http://protege.stanford.edu/
[5] Specifications and applications can be found in http://www.w3.org/2004/OWL/
[6] Specifications can be found in http://www.w3.org/Submission/SWRL/

Table 6. Axioms expressed by SWRL

Axiom
Research_Project(?x) $\wedge$ Project_Report(?y) $\wedge$ hasReport(?x, ?y) $\wedge$ hasReportState(?y, "rejected") $\rightarrow$ hasProjectState(?x, "cancelled")
Manager(?x) $\wedge$ hasMonthlyDedication(?x, ?y) $\rightarrow$ swrlb:greaterThanOrEqual(?y, 20)

(1) *syntactic dimension*, regarding the way the ontology is written according to a particular implementation language, (2) *semantic dimension*, concerning the consistent modelling of the ontology, and (3) *functional dimension*, related to the intended use of the ontology.
In the study case, syntactic quality was reached during the OWL coding while semantic quality was verified by checking the consistency using the Pellet inference engine. The functional quality of the ontology was reached by testing its ability to answer the CQs formulated in the specification activity. Although such testing can be performed by means of an analytical review of the ontology, the implementation of CQs in the SPARQL language[7] allows a more systematic testing process by querying the ontology. Table 7 shows two CQs implemented in SPARQL. The former ask for the researcher category required to be the manager of a research project. The latter ask for the types of documents evaluated for a funding agency.

Table 7. SPARQL queries

Queries
SELECT ?object WHERE { rpms:Manager rpms:has_Category ?object }
SELECT ?object WHERE { rpms:agency_evaluate rdfs:range ?object }

3.3 Ontology Alignment

The alignment activity consists in establishing a set of correspondences between entities belonging to two different ontologies. As a result, a new ontology composed by sub-ontologies is created. Each correspondence is defined as follows [**?**]:

Definition 2. *A correspondence Co is a 4-uple {id , e1, e2, r} where:*

- *id is the identifier of a given correspondence.*
- *e1 and e2 are entities, e.g., classes and properties of the first and the second ontology, respectively.*
- *r is the relation, e.g., equivalence (=), more general ($\subseteq$), disjointness ($\perp$), holding between e1 and e2.*

[7] Specifications can be found in http://www.w3.org/TR/rdf-sparql-query/

So, the correspondence Co:= $\{id\ ,\ e1,\ e2,\ r\}$ asserts that the relation r holds between the ontologies entities *e1* and *e2*.
Subsections above have depicted the development of the first version of the ontology of the study case, which does not involve the performing of alignment activities. A brief description of the changes included in the second iteration is depicted to follow, with the aim of illustrating the alignment process.
Funding agencies were the main source of business changes in the study case because the requirements to be met by a research project to be funded by an agency are constantly changing. Finished the first iteration, a particular funding agency begun to provide financial support to different kinds of research projects, grouped in three disjoint categories. With the aim of reflecting this change, a second iteration of EDON was initiated. Some terms identified in the specification activity are shown in Table 8.

Table 8. Some new terms identified

One-Man Research Project	Co-Led Research Project	Promoted Research Project
Project Co-Manager	Promoted Project Team	PhD Fellow

DEs were mainly involved in establishing the correspondences between the entities of the two ontologies, by using the alignment services provided by the Protégé-based PROMPT tool [**?**]. PROMPT tool provides a semi-automatic approach to ontology alignment, suggesting specific correspondences between entities and showing the reasons that motivated such suggestions. An excerpt of the correspondences established is shown in Table 9.

Table 9. Some correspondences established in the alignment activity

e1 *(Entity of Ontology 1)*	**r** *(Relation)*	**e2** *(Entity of Ontology 2)*
Research Project	$\supseteq$	One-Man Research Project
Research Project	$\supseteq$	Co-Led Research Project
Research Project	$\supseteq$	Promoted Research Project
Researcher	$\supseteq$	PhD Fellow
Manager	$\perp$	Co-Manager
Manager	$\perp$	PhD Fellow

Alignment evaluation activity is performed over the aligned ontology with the aim of testing the three dimensions - syntactic, semantic and functional - that define its quality. In the study case, syntactic and semantic dimensions were respectively tested by means of the editor and inference engine. Functional testing was facilitated by the availability of SPARQL implementations of CQs that allowed to test the ontology by querying it. Minor reformulations of correspondences established in the alignment activity were necessary because some CQs were not reached, although they are not shown here for the sake of space.

3.4 Experiences and Learned Lessons

In the study case, EDON was intertwined with the SCRUM agile software development framework [8]. The RPMS software application was implemented in Java[9] and the produced ontology was integrated by using the Jena Java API[10], which provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine. Each iteration of both software and ontology development processes was time-boxed in 30 days, enabling the quickly integration and release of successive versions of the software system. The ontology development was done in three iterations. As expected, the most time-consuming activities were those requiring deeper knowledge elicitation efforts, i.e., conceptualization, refinement and alignment consumed around 75% of the iteration time.

Development of the study case have given us some feedback about relevance of using an ontology as a software artefact intended to encapsulate business rules, and the aptness of EDON to be used as a methodology for build such ontologies. First, the requirements grouping activity has shown a deep influence in the efficiency of the overall ontology development. The correct grouping of requirements facilitate a modular ontology design as a way to achieve maintainability and evolution. A better modularization results on clearer correspondences to be defined in the alignment activity. Second, although the ontology conceptualization by using LEL has proven to be useful to facilitate the communication between the DEs and KEs, we consider that a more powerful formalism will improve the way complex business rules are expressed.

The study case has also shown the applicability and acceptance of EDON by the development team of the software application and the DEs involved in the ontology development. Development team highlighted the ability to adapt the overall software system to changing business requirements with minor modifications on the software procedural code, by encapsulating the evolving features of the business in a specific software artefact. Meanwhile, DEs showed a high degree of interest on the possibility of embed its knowledge in the software system, participating in the process in a intuitive manner and evolving from knowledge to ontology in a smooth way.

4 Discussion and Future Research Directions

EDON partly draws on some of the best practices from the existing methodologies for ontology building. The use of competency questions to scope and evaluate the conceptualization was originally proposed in a very formal methodology based on the TOVE project, inspired on the development of knowledge-based systems and the use of first order logic [?]. The iterative life cycle, definition of orthogonal activities and collaborative work of DEs and KEs was depicted

[8] More information in http://www.scrumalliance.org/pages/what_is_scrum

[9] More information in http://download.oracle.com/javase/6/docs/technotes/guides/language/

[10] Documentation, downloads and support can be found in http://jena.sourceforge.net/

by Methontology [?]. Like EDON, approaches applying the main principles of XP agile software methodology also encourages the active involvement of DEs, development of evolving versions of the ontology through short-time boxed iterations, and production of as reduced as possible amount of intermediate deliverables along the process. Moreover, EDON is strongly rooted on the use of application languages, lexicons and the systematic process defined in [?].
EDON takes advantage of semantic richness of Application Languages [?] to bridge the gap between the knowledge domain and the ontology, representing the raised lexicon by means of LEL [?]. The use of LEL has several advantages over common glossaries since it provides additional information to the meaning of terms in the format of a list of relations to other lexicon terms. Furthermore, classification of lexicon terms in pre-defined categories provides a starting point to the ontology modelling of concepts.
Several distinctive features of EDON has been already highlighted. First, the development of an ontology intended to be used as a structural conceptual model of an information system, encoding business rules in a declarative way. Second, the aptness of EDON to intertwine the software and ontology development process, enabling the production of readily available evolving versions of the software system. In this way, although most of the modern Software and Knowledge Engineering approaches in ontology building are evolutionary and based on iterative development, EDON differs significantly with them in the duration of iterations and use of each version of the ontology. The third distinctive feature of EDON is the proposal of a phase model that takes advantage of the benefits of agile approaches. By means of the phase model, EDON provides a descriptive path to quickly and smoothly evolve through further extended versions of the ontology while stems from agile approaches its ability to adapt to changes in the business domain.
Based on the experience obtained from the study case, future research directions will be focused on the following areas. First, tracing a set of guidelines for the correct grouping of requirements will help the experts to improve the requirements selection process. Second, researching for a more powerful formalism for ontology conceptualization will enable the expression of complex business rules in a high level of abstraction. As a consequence, will be necessary to formulate a systematic process to generate the ontology structure from such high level expressions. In addition, we will intend to acquire additional validation cases for EDON.

References

1. Auer, S.: RapidOWL - an Agile Knowledge Engineering Methodology. In Proc. Perspectives of System Informatics. (2006)
2. Beck, K.: Extreme Programming Explained: Embrace Change. A. Wesley. (2000)
3. Breitman, K. and Leite, J.C.S.P.: Lexicon based ontology construction. In Software Engineering for Multi-Agent Systems II.. Springer/Berlin/Heidelberg **LNCS 2940** (2004) 41–45

4. Burton-Jones, A. and Storey, V.C. and Sugumaran, V. and Ahluwalia, P.: A semiotic metrics suite for assessing the quality of ontologies. In Data Knowledge Engineering **55-1** Elsevier Science. Amsterdam, The Netherlands. (2005) 84–102
5. Carnap, R.: The Methodological Character of Theoretical Concepts. In Minnesota Studies in the Philosophy of Science **1** University of Minnesota Press. Minneapolis, USA. (1956)
6. Euzenat, J. and Shvaiko, P.: Ontology Matching. Springer/Berlin (2007)
7. Fernández-López, M. and Gómez-Pérez, A. and Juristo, N.: METHONTOLOGY: from Ontological Art towards Ontological Engineering. In Proc. AAAI97 Spring Symposium Series on Ontological Engineering (1997) 33–40
8. Gangemi, A. and Catenacci, C. and Ciaramita, M. and Lehmann, J.: Modelling ontology evaluation and validation. In Proc. 3rd European Semantic Web Conf. (ESWC2006) **LNCS 4011** Springer. (2006)
9. Gómez-Pérez, A. and Fernández-López, M. and Corcho, O.: Ontological Engineering. Springer/Heidelberg. (2004)
10. Grüninger, M. and Fox, M.S.: Methodology for the design and evaluation of ontologies. In Proc. Workshop on Basic Ontological Issues in Knowledge Sharing in IJCAI 95. Montreal, Canada. (1995)
11. Hepp, M.: Possible Ontologies: How Reality Constrains the Development of Relevant Ontologies In Internet Computing, IEEE **11** (2007) 90–96
12. Hristozova, M. and Sterling, L: An eXtreme method for developing lightweight ontologies. In Workshop on Ontologies in Agent Systems, 1st Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems (2002)
13. Knublauch, H.: An Agile Development Methodology for Knowledge-Based Systems Including a Java Framework for Knowledge Modeling and Appropriate Tool Support. University of Ulm (2002)
14. Leite, J.C.S.P. and Franco, A.P.M.: A Strategy for Conceptual Model Acquisition. In Proc. IEEE International Symposium on Requirements Engineering. IEEE Computer Society Press. (1993) 243–246
15. Nicola, A.D. and Missikoff, M. and Navigli, R.: A software engineering approach to ontology building. In Information Systems **2-34** Elsevier Science. (2009) 258–275
16. Noy, N.F. and Musen, M.A.: PROMPT: Algorithm and tool for automated ontology merging and alignment. In Proc. 17th National Conf. on Artificial Intelligence AAAI-00 MIT Press/AAAI Press. Austin, Texas. (2000)
17. Pavel, S and Euzenat, J.: Ontology Matching: State of the Art and Future Challenges In IEEE Transactions on Knowledge and Data Engineering, **PP-99** (2011).
18. Ruotsal, T.: Methods and Applications for Ontology-Based Recommender Systems (PhD. Thesis) Aalto University School of Science and Technology. Finland. (2010)
19. Sharifloo, A.A. and Shamsfard, M.: Using Agility in Ontology Construction. In Proc. 2008 Conf. on Formal Ontologies Meet Industry. IOS Press. Amsterdam, The Netherlands. (2008) 109–119
20. Vieira, T.A.S.C. and Casanova, Marco A. and Ferrão, L.G.: An Ontology-Driven Architecture for Flexible Workflow Execution. In Proc. WebMedia and LA-WEB. IEEE Computer Society. (2004) 70–77
21. Vrandeĉić, D.: Ontology Evaluation. In Handbook on Ontologies (2nd edn) International Handbook on Information Systems. Springer. (2009) 293–313
22. Zacharias, V.: Rules as simple way to model knowledge: Closing the gap between promise and reality In Proc. 10th Int. Conf. on Enterprise Information Systems (2008).