Contents lists available at ScienceDirect





journal homepage: www.elsevier.com/locate/infosys



A software engineering approach to ontology building

Antonio De Nicola^{a,*}, Michele Missikoff^a, Roberto Navigli^b

^a Istituto di Analisi dei Sistemi ed Informatica, Consiglio Nazionale delle Ricerche, Viale Manzoni, 30-00185 Roma, Italy
^b Dipartimento di Informatica, Università di Roma "La Sapienza", Via Salaria, 113-00198 Roma, Italy

ARTICLE INFO

Article history: Received 15 October 2007 Received in revised form 11 June 2008 Accepted 21 July 2008 Recommended by: P. Loucopoulos

Keywords: Ontology building Knowledge representation formalism Knowledge representation method

ABSTRACT

Ontologies are the backbone of the Semantic Web, a semantic-aware version of the World Wide Web. The availability of large-scale high quality domain ontologies depends on effective and usable methodologies aimed at supporting the crucial process of ontology building. Ontology building exhibits a structural and logical complexity that is comparable to the production of software artefacts. This paper proposes an ontology building methodology that capitalizes the large experience drawn from a widely used standard in software engineering: the Unified Software Development Process or Unified Process (UP). In particular, we propose UP for ONtology (UPON) building, a methodology for ontology building derived from the UP. UPON is presented with the support of a practical example in the eBusiness domain. A comparative evaluation with other methodologies and the results of its adoption in the context of the Athena EU Integrated Project are also discussed.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

Ontologies, i.e., semantic structures encoding concepts, relations and axioms, providing a model of a given domain, are the backbone of the Semantic Web [1], a semantic-aware version of the World Wide Web. Ontologies allow web resources to be semantically enriched. This is a pre-condition to provide new, advanced services over the web, such as semantic search and retrieval of web resources.

Ontology building is a task that pertains to ontology engineers, an emerging expert profile that requires the expertise of knowledge engineers (KEs) and domain experts (DEs). Even though automatic ontology learning methods (such as text mining and knowledge extraction [2]) significantly support ontology engineers by speeding up their task, there is still the need of a significant manual effort, in the completion, consolidation, and validation of the automatically generated ontology.

Today, there are several methods for building ontologies, but no one is yet emerging as a clear reference. The aim of this paper is to present a method, namely the Unified Process for ONtology (UPON) building, based on a well-established and widely used software engineering process, the Unified Process (UP) [3].

UPON is a novel approach to large-scale ontology building that takes advantage of the UP and the Unified Modeling Language (UML) [WR1]. This paper illustrates the phases, the steps, and intermediate outcomes of the UPON method, designed in accordance with the UP method, aimed at guiding ontology engineers in the production of an effective and valuable domain ontology [4]. UPON has been used in the context of the Athena Integrated Project [WR2], to build four different ontologies in as many different pilots, established in the domain of automotive, aerospace, kanban logistics, and furniture. Throughout the paper, we discuss a running example concerning the latter: an eProcurement ontology for the Spanish woodworking and furniture sector. The DEs were coming from AIDIMA [WR3], an Athena partner dedicated

^{*} Corresponding author. Tel.: +39067716433; fax: +39067716461. *E-mail addresses*: denicola@iasi.cnr.it (A. De Nicola),

missikoff@iasi.cnr.it (M. Missikoff), navigli@di.uniroma1.it (R. Navigli).

^{0306-4379/\$ -} see front matter \circledast 2008 Elsevier B.V. All rights reserved. doi:10.1016/j.is.2008.07.002

to technology and innovation transfer to the SMEs of the sector.

The main objectives of UPON are:

- reduction of time and costs in the production of largescale domain ontologies (providing also useful guidelines for small ontologies);
- enhancement of the quality of the produced ontology, by progressive validation of the intermediate results;
- creation of a methodological setting where the two kinds of expertise, KE and DE expertise, are explicitly identified and used at best;
- clear identification of the activities, with roles and responsibilities of the different experts;
- production of intermediate results that can be readily available to the users of the ontology-based applications (e.g., semantic search).

The paper is organized as follows. Section 2 presents an overview of the UPON approach to ontology building. Section 3 describes UPON in detail with the support of a running example. Section 4 discusses previous work in this area and provides a first comparative evaluation of UPON, against other ontology building methodologies. Finally, in Section 5, we provide conclusions and future research directions.

2. An overview of UPON: Unified Process for ONtology building

In this section, UPON, an incremental methodology for ontology building, is presented. As anticipated, UPON stems its characteristics from the UP, one of the most widespread and accepted methods in the software engineering community, and uses the UML to support the preparation of all the blueprints of the ontology development. UML has been already shown to be useful in building ontologies [5], confirming its nature of a rich and extensible language.

What distinguishes the UP and UPON approach from other methods, respectively, for software and ontology engineering, is the *use-case driven*, *iterative*, and *incremental* nature.

UPON is *use-case driven* since it does not aim at building generic domain ontologies, but ontologies that serve its users, both humans and automated systems (e.g., Semantic Web services, intelligent agents, etc.), in a welldefined application area and with accurately defined objectives. *Use cases* are the first diagrams that drive the exploration of the application area, at the beginning of the ontology building process.

The nature of the process is *iterative* because each activity is not only cyclically repeated, typically concentrating on different parts of the ontology being developed, but also *incremental*, since at each cycle the ontology is further detailed and extended.

Following the UP approach, in UPON there are *cycles*, *phases*, *iterations*, and *workflows*. Each cycle consists of four phases (*inception*, *elaboration*, *construction*, and *transition*) and eventually results in the release of a new



Fig. 1. A sketchy representation of the UPON process.

version of the ontology. Each phase is further subdivided into iterations. During each iteration, five workflows (described in the next subsections) take place: *requirements, analysis, design, implementation,* and *test.* A sketchy representation of the UPON process is presented in Fig. 1. Please note that, for the sake of space, workflows' names are only expanded in the construction phase.

Workflows and phases are orthogonal since the contribution of each workflow to an iteration of a phase can be more or less relevant: early phases are mostly concerned with establishing the requirements (identifying the domain, scoping the ontology, etc.), whereas later phases result in additive increments that eventually bring to the final release of the ontology (Fig. 2). Notice that, as illustrated in the figure, more than one iteration may be required to complete each of the four phases. Cycles produce major versions of the ontology that must be constantly aligned with the modeled reality. This scheme faithfully follows the UP. Fig. 2 reports the well-known diagram, drawn from the original proposal of the UP. On the right we added a diagram that suggests the degree of involvement of the two key experts necessary to build a sound ontology. DEs provide their contribution in the early workflows and partially during the *test* while KEs are mostly focused on the design and implementation workflows.

The first iterations (inception phase) are mostly concerned with capturing requirements and partly performing some conceptual analysis. Neither implementation nor test is involved. During subsequent iterations (belonging to the elaboration phase) analysis is performed and fundamental concepts are identified and sketchily structured. This may require some design effort and it is also possible that modelers provide a preliminary implementation in order to have a small skeletal blueprint of the ontology, but most of the design and implementation workflows pervade the iterations in the construction phase. Here some additional analysis could be required, aiming at identifying concepts overlooked in the previous phases to be further added to the ontology. During final iterations (transition phase), testing is the key activity performed and the final ontology (resulting from the current cycle) is eventually released. In parallel, the material necessary to start the new cycle, that will produce the next version of the ontology, is collected. As shown in Fig. 2, at each iteration different workflows come into play and a richer and more complete version of the target ontology is produced.



Fig. 2. The UPON framework and ontology engineers' involvement.

The incremental nature of UPON is reflected also by the outcomes produced in the different phases. First the relevant terms in the domain are identified and gathered in a *lexicon*; then the latter is progressively enriched with definitions, yielding a *glossary*; populating it with the basic ontological relationships allows a *semantic network* (*SN*) to be produced, until further enrichments and a final formalization produces the sought *domain ontology*.

A characterizing issue of UPON is the distinction of the application knowledge vs. the domain knowledge. The latter represents the context in which the application is placed. The former takes into consideration the latter but is focused on the application issues. Domain knowledge, having a general scope, is mainly acquired through preexisting resources (domain glossaries, technical documents, standards, manuals and technical guides, etc.) while application knowledge is provided ad hoc by the application experts and application-related documents.

In the following, we support the methodology with a running example concerning eProcurement in the furniture domain. In our example, the ontology chiefly concerns activities and interactions taking place when a client buys some products or services from a supplier (e.g., exchange of business documents like an *invoice* or a *purchase order*).

3. The UPON methodology

In the following subsections the UPON workflows, as well as the steps which compose them, are reported in detail, by providing specific information about each individual step.

3.1. The requirements workflow

Capturing requirements consists in specifying the semantic needs and user view of the knowledge to be

encoded in the ontology. This activity requires an agreement between application modelers, KEs, and end users [3].

During the first meetings, KEs and DEs establish the guidelines for building the ontology. The first task consists in identifying the objectives of the ontology from a user's point of view. To this end, it is necessary to: (i) determine the domain of interest and the scope, and (ii) define business purpose. These objectives are achieved by: (iii) writing one or more storyboards [6], (iv) creating an application lexicon (AL), (v) identifying the competency questions (CQs) [7], and (vi) modeling the related use cases. Each point will be elaborated below in detail. In Fig. 3, the sequencing of the activities of this workflow is reported with the related degree of involvement of the two key experts. The main contribution is given by DE since, as already mentioned, this workflow is devoted to specify semantic needs and user views of the knowledge to be encoded in the ontology. However, KE plays an important role in the activity (vi) where conceptual modeling expertise is required.

Below we describe the six above points in detail.

(i) Determining the domain of interest and the scope. Circumscribing the domain of interest is a fundamental step [8], to focus on the appropriate fragment of reality to be modeled. If the domain is large, two or more subdomains may also be determined.

As anticipated, in this work the domain addressed to validate the UPON methodology is eBusiness. Specifically, the eProcurement subdomain, i.e., the *business-to-business* (B2B) purchasing and selling goods and services over the Internet [WR4], has been addressed.

Defining the scope of the ontology consists in the identification of the most important concepts to be represented, with their characteristics, thus pushing the refinement to the suitable granularity. To this end, a set of ontological commitments [9] is required, bringing some part of the domain into focus at the (required and expected) expense of other parts that will be less represented or neglected.



Fig. 3. Requirements workflow: an overview.

Following Guarino et al. [9], the ontological commitment can be seen as "a mapping between a language and something which can be called an ontology". In this preliminary stage, it can be seen as a set of statements that allows one to identify a first set of terms as representatives of ontology concepts.

Usually at this stage modelers have only a vague idea of the position that each concept will assume within the ontology, i.e., the semantic interconnections between pairs of concepts. If necessary, each term can be informally annotated by DE for further development during subsequent iterations.

(ii) Defining business purpose (or motivating scenario, with users and their objectives). The reason for having an ontology, its intended uses, and kinds of target users must be established. In the eProcurement application, the goal of the ontology is to provide a better understanding of the domain of interest and to support a number of semantic services for ontology users (i.e., clients and suppliers in the purchasing processes). In particular, three basic uses of the ontology can be envisaged:

- resource discovery and retrieval, e.g., for semantically enriched documents or web services;
- ontology-based reconciliation of data messages exchanged between sender and receiver business actors in business transactions [10];
- ontology-based mediation of business processes belonging to two different business partners (e.g., the steps in a purchasing activity).

In this workflow we have also specific activities, articulated in the following four points.

(iii) *Writing one or more storyboards*. In this step the DE is asked to write a panel or series of panels of rough sketches outlining the sequence of the activities that take place in a particular scenario. Storyboards (modeling

contexts and situations in a narrative way) can be used in this activity. In particular, a storyboard is related to a particular scenario of the realization of one or more business purposes. In the considered eProcurement application, a storyboard sounds as follows:

"The ACME manufacturer (client) sends a request for quotation (RFQ) to the EMCA provider (supplier). The supplier processes the RFQ and sends back his quotation to the client. The client evaluates the quotation and, possibly, makes the purchase order (PO) that is sent to the supplier. The supplier accepts the purchase order, fulfills it, delivers the goods, and sends the invoice to the client. Finally, if the delivered goods correspond to the order, the client pays the invoice."

(iv) *Creating the AL*, by collecting the terminology from DE and application-specific documents. The storyboard is used to extract the terminology. Then a preliminary version of the *AL* is built. This task can be supported by using some automatic tools to extract knowledge from textual documents, such as OntoLearn [2], Text-To-Onto [WR12], or similar tools. In the eProcurement example, the AL contains 212 terms, mainly extracted from documents representing the reference XML schemas of AIDIMA business documents. An excerpt of the *AL* is reported in Table 1.

(v) Identifying the Competency Questions (CQs). CQs are questions at a conceptual level¹ an ontology must be able to answer [7]. They are essentially identified through interviews with DEs and end-users brainstorming. The questions are used during the test workflow to evaluate the ontological commitments that have been made and, more in general, the coverage (related to domain

¹ Please note that CQs are inherently different from a database query that aims at retrieving instances, not concepts.

Table 1

| An excerpt of the eProcuremen | t application lexicon |
|-------------------------------|-----------------------|
|-------------------------------|-----------------------|

| Address | Due date | Phone |
|----------------------|-------------------|-----------------------|
| Amount | Due payment | Postal code |
| Bank account details | e-mail | Quantity |
| Bank account number | Fax | Quotation |
| Client info | Invoice | Quotation date |
| Client order number | Invoice date | Quotation number |
| City | Invoice number | Request for quotation |
| Country | Order | Street |
| Currency | Order date | Supplier info |
| Delivery date | Order number | Terms of payment |
| Delivery note | Payment condition | Total price |
| | | |

Examples of competency questions for the eProcurement application

| CQ1 | What are the documents that a client sends to a supplier during an eProcurement process? |
|------|---|
| CQ2 | What are the documents that a client receives from a supplier during an eProcurement process? |
| CQ3 | What are the activities composing an eProcurement process? |
| CQ4 | What is the sequencing of documents exchanged in an |
| | eProcurement transaction? |
| CQ5 | Who sends a purchase order? |
| CQ6 | Who evaluates a quotation? |
| CQ7 | Who issues a purchase order? |
| CQ8 | What are the transactions achieved by the cooperating |
| | companies A and B before issuing a purchase order? |
| CQ9 | Is the purchase order schema of A semantically equivalent to |
| | the selling order schema of B? |
| CQ10 | Is the "address" in the purchase order schema of A reconcilable |
| | with the "location" in the selling order schema of B? |

circumscription) and the depth (level of detail of the ontology to be built). According to the main objectives of an ontology, CQs can be of two different kinds: oriented either to *resource discovery* and *retrieval* or to *reconciliation* (within a semantic interoperability solution). Some examples of CQs in the eProcurement application are presented in Table 2.

(vi) Use-case identification and prioritization. UPON proposes to address COs by using use-case models. According to UML, a use-case model contains a number of use cases that serve as a basis to specify the expected use of the ontology. This model is a result of an agreement among different users (i.e., who will use the ontology for a business purpose) achieved with the support of ontology modelers. In the context of ontologies, use cases correspond to knowledge paths through the ontology, to be followed for achieving business operations and answering CQs. Use cases will be detailed during the analysis and design workflows; here, in the requirements workflow, they are identified and prioritized. The result will help indicate which use cases should be addressed during early iterations, and which ones can be postponed. An example of a use-case model with the corresponding CQ is reported in Fig. 4.

The output of the requirements workflow is a set of documents (Fig. 3), namely:

- use-case models,
- the application lexicon.

The latter can be more precisely specified as follows:

Definition 1. Lexicon

A lexicon is defined as the set of terms

$$L = \{t_i\}, \text{ being } i \in N \text{ and } i \leq i_{\max}$$

where

 $i_{\text{max}} = |L|$

Definition 2. Application lexicon

An *application lexicon*, AL, is a lexicon where the terms pertain to a given application *APP*, as stated by a community of application experts.

 $AL = \{at_i \in L | pertains(at_i, APP)\}, being i \in N$

3.2. The analysis workflow

The conceptual analysis concerns the refinement and structuring of the ontology requirements identified in the previous workflow. The ontological commitments derived from the definition of the ontology scope are extended, by reusing existing resources and through concept refinement. The *AL* is enriched through a more general *domain lexicon* (DL), to build the *reference lexicon* (RL). The latter is then used to produce the *reference glossary* (RG) (Fig. 5), by adding definitions to the terms. The expert involvement schema in Fig. 5 shows that the main role is played by DE providing his knowledge on the domain of interest and on the given application. Nevertheless KE gives an important contribution during UML modeling of the application scenario due to his expertise on conceptual modeling.

(i) Acquiring domain resources and building a domain lexicon. The DL is built by gathering the terminology used in the domain of interest, mainly extracted by analyzing existing documental resources, such as reports, technical manuals, standards, glossaries, thesauri, legacy computational lexicons, and available ontologies. This step, like in the case of the AL, can be supported by automatic tools for text mining. This activity inherently adheres to the view of a linguistic ontology [11] in which concepts are anchored to textual descriptions, i.e. they have a counterpart in natural language.

In our project, to build the eBusiness *DL*, an important role has been played by standards. The DEs considered the following eBusiness standards: ebXML [WR5], RosettaNET [WR6], and OAGIS [WR7]. The analysis of these standards allowed 2613 elements to be extracted (140 from ebXML, 1873 from RosettaNET, 600 from OAGIS)²; then a consistent filtering has been necessary. In fact, since many elements from these standards pertain to specific industry sectors (e.g., elements coming from *RosettaNet* mainly refer to the high-tech electronic industry), a first manual pruning was performed. Then terminological analysis was done against the corpus of documents of reference, to identify specific (frequently used in eProcurement) terms to be included in the DL. Furthermore, DEs decided to include in the DL, all the terms present in at least two

[•] competency questions,

² The analysis was manually performed by the domain experts.

Competency Question:

CQ8 - What are the transactions achieved by the cooperating companies

A and B before issuing a purchase order?

Use-case model:



Fig. 4. A competency question and an excerpt of the corresponding use-case model.



Fig. 5. An overview of the analysis workflow.

standards. Other terms were included only after approval from a wider panel of experts. After this activity, the *DL* contained 114 terms (not considering synonyms). (An excerpt of the *DL* is reported in Table 3.) In a simplified vision, useful for our example, these 114 terms represent the core of the initial 2613 terms. Fig. 6 sketchily reports the composition of the *DL*, with respect to the originating resources.

(ii) Building the reference lexicon. The *RL* is built by selectively merging the *AL* (from application DEs) and the *DL* (from existing external resources). During the merge of the two lexicons, the terms are grouped into three major areas: one intersection area and two disjoint areas, application specific and domain specific (see Fig. 7). To

build the *RL* the following "inclusion policy" is used: the RL should include all the terms coming from the intersection area and, after the users and DEs approval, some terms belonging to the disjoint areas. Therefore the intersection area will be extended, on the one hand, with *domain terms*, considered useful for a better specification of the application at hand, and, on the other hand, with a part of the remaining *application terms* that are considered relevant, even if not extensively used by other application, it contains 139 terms.

(iii) Modeling the application scenario using UML diagrams. The goal of this activity is to model the application scenario, adding to the use-case diagrams,

drawn in the Requirements Workflow, the activity and class diagrams. UML diagrams represent a model of the application and will be used for the *validation* of the ontology. All classes, actors, and activities modeled in

Table 3

An excerpt of the eProcurement domain lexicon

| Account name | Organisation name | Seller |
|-------------------|-------------------------|---------------------|
| Account number | Payment method | Street name |
| Amount | Person name | Tax amount |
| Client | Person surname | Tax description |
| City | Product name | Telephone number |
| Contact name | Purchase order | Total price |
| Country name text | Purchase order date | Transportation term |
| Currency | Quantity | Transport mode |
| Date | Quote | Unit price |
| Fax number | Region | |
| Line item amount | Requested delivery date | |
| | | |

UML must have a corresponding concept in the ontology. In Fig. 8, three representative UML diagrams, drawn for the eProcurement application, are reported. Please note, in the Class Diagram (Fig. 8c), the use of the *«*Object*»* profile. UML profiles are used to properly categorize the terms, according to the OPAL ontology representation methodology [12] (see below).

(iv) Building the reference glossary (RG). A first version of a glossary is built by using the RL and by adding informal definitions (i.e., natural language sentences) to the terms. In essence, the RL evolves into a *reference* glossary (Fig. 9) by associating one or more definitions to each term. The definitions should be selected from knowledgeable sources and agreed among DEs and users. An excerpt from the *reference* glossary for the AIDIMA eProcurement domain is reported in Table 4. In this application the *reference* glossary contains 139 glossary entries.



Fig. 6. Sources of the terms in the eProcurement domain lexicon.



Fig. 7. Activity of reference lexicon building.



Fig. 8. An excerpt of the UML diagrams of the application scenario: (a) use-case diagram; (b) activity diagram; and (c) class diagram.

In summary, the analysis workflow releases a *reference* glossary that will be used in the next workflows. Below a specification of *DL*, *RL*, glossary, and *reference* glossary is reported.

Definition 3. Domain lexicon

The *domain lexicon* is a lexicon validated by a community of domain experts.

 $DL = \{ dt_i \in L | pertains(dt_i, Dom) \}, being i \in N \}$

where Dom is the domain of interest.

Definition 4. Reference lexicon

The *reference lexicon* is a lexicon validated by a community of application and domain experts with the coordination of knowledge engineers.

 $RL = (AL \cap DL) \cup \delta AL \cup \delta DL$

where

 $\delta AL = \{t_i \in AL \mid DL \mid approved(t_i)\}, \text{ being } i \in N$

and

 $\delta DL = \{t_i \in DL \mid AL \mid approved(t_i)\}, \text{ being } i \in N$

 δAL and δDL are, respectively, subsets of AL and DL. They are disjoint. Their terms are included in the RL since considered relevant by users and domain experts. Please note that *approved*(*x*), as well as *validated*(*x*,*y*) (see below) are "human judgment" [13] predicates that are evaluated by a board of domain experts.

Definition 5. Glossary and reference glossary

A glossary G is defined as the finite set of terms belonging to a lexicon L paired with the corresponding descriptions validated by the reference community. The couple term and description is defined as the glossary entry g_i . The reference glossary, RG, is a glossary validated by a community of application and domain experts with the coordination of knowledge engineers.

 $G = \{ \langle t_i, des_i \rangle | t_i \in L \land des_i \in DES \land validated(t_i, des_i) \},\$



Fig. 9. Activity of reference glossary building.

An excerpt of the reference glossary

| Term | Description |
|---------------------------|---|
| Client | An employee or organisation that buys products for a partner type in the supply chain (RosettaNet) |
| Issuing purchase order | The process that enables a client to issue a purchase order (RosettaNet+) |
| Postal address | The collection of information which locates and identifies a specific address as defined by postal services (ebXML) |
| Purchase order | A printed or typed document, issued by the Client Purchasing Unit as a firm and formal request to a specific Manufacturer/Supplier to produce and supply goods/services according to Price, Terms and Conditions previously agreed and approved (Merriam Webster+AIDIMA application experts) |

 $g_i = \langle t_i, des_i \rangle$, being $i \in N$

 $RG \subseteq RL \times DES$

 $RL = \{rt_i\}, \text{ being } i \in N$

 $DES = \{des_i\}, being i \in N,$

where des_i is the textual description of term t_i .

3.3. The design workflow

The main goal of this workflow (Fig. 10) is to give an ontological structure to the set of glossary entries gathered in the *reference glossary*. In brief, we leave the linguistic dimension to enter in the conceptual dimension. This is essentially achieved by organizing the terms according to conceptual hierarchies and structuring them with attributes and axioms. The refinement of entities, actors and processes identified in the analysis workflow, as well as identification of their relationships, is performed during the design workflow. This structuring process is carried out according to OPAL [12,14], an ontology representation methodology developed at LEKS, IASI-CNR, based on UML and OWL [15]. The phases of the design workflow, presented below, concern first structural modeling of concepts and then construction of hierarchies and domain relations. The involvement schema in Fig. 10 shows that an important role is played by KE in modeling concepts and building relationships. However, a minor support is provided by DE for the required application and domain knowledge.

(i) *Modeling concepts.* Each concept is categorized by associating a "kind" to it. Here we adopt the categories proposed by OPAL. These include the major ontological categories, according to proposals of top ontologies, such as [16], or meta-ontologies [8,10,14], and they are deeply inspired by the primary modeling constructs of UML. OPAL organizes concepts in three primary and some complementary categories. The primary categories are:

- *business actor*: gathering active elements of a business domain, able to activate, perform, or monitor a business process. DEs, in analyzing the reality, are asked to identify relevant actors that operate producing, updating or consuming business objects (BOs). In the eProcurement example, 15 business actors have been identified (Table 5).
- business object: an entity on which a business process operates. A business object document (BOD) is a further refinement of a BO that represents a category of documents in the business domain. In the AIDIMA scenario 14 BODs, belonging to the client and the supplier, have been identified (Table 6).
- business process: a business activity or operation aimed at the satisfaction of a business goal, operating on a set of BOs (e.g., purchase order issuing, requesting quotation). It can be rather simple, with a limited



KE: Knowledge Engineer

Fig. 10. An overview of the design workflow.

Table 5 Business actors

| Broker Exporter | Payee |
|---------------------------|--------------|
| Business partner Financer | Payer |
| Carrier Freight forwarder | Person |
| Client Importer | Organisation |
| Consignee Manufacturer | Seller |

Table 6

Business object documents

| Delivery receipt | Purchase order | Request for quote |
|-------------------|-----------------------------|-----------------------|
| Invoice | Purchase order acceptance | Quote |
| Order change | Purchase order cancellation | Sale order |
| Planning schedule | Purchase order rejection | Shipment confirmation |
| Price list | Purchase order update | |

duration in time, or complex, with parallel branches and phases³ that last for a long time span. In the eProcurement example 19 business processes have been identified (Table 7).

Having presented the primary concept categories of OPAL, we now introduce the complementary categories, necessary to complete a rich ontological representation of the observed reality.

Message: A message represents the information exchanged during an interaction (e.g., request, response) between processes (someone prefers to say "among actors", but we prefer to focus on the processes that need to exchange info). A message is characterized by a

| Table 7 | |
|----------|-----------|
| Business | processes |

| Cancelling PO Changing PO Despatching Distributing price list Invoicing Issuing delivery receipt | • | Requesting PO changes Requesting quote |
|---|-------------------------|---|
| Issuing delivery receipt Issuing invoice | Ordering Paying bill | |
| issuing involce | Faying Din | |

content that is typically a BOD (e.g., a RFQ-message, carrying a request for quotation). In OPAL we adopted the FIPA [WR8] approach, based on 23 message types, related to different kinds of communicative acts. Specifically, we selected 8 message kinds for which the payload matches a BOD in the eProcurement ontology. Table 8 shows the 8 selected message kinds matched with their corresponding BODs in the eProcurement application.

Attribute: Attributes characterize the information structure of a concept. In OPAL there are atomic attributes, modeling elementary information (e.g., street name), and complex attributes, modeling structured information (e.g., address). Essentially, a complex attribute is defined as an aggregation of lower level complex and/or atomic attributes. Table 9 reports the list of 83 attributes identified in the eProcurement example.

Having categorized the concepts identified in the eProcurement example, the resulting ontology consists of 139 concepts: 15 business actors, 14 BODs, 19 business processes, 8 messages and 83 attributes, complex and atomic.

(ii) Modeling concept hierarchies and domain-specific relationships. At this stage, concepts are hierarchically organized and formal relations are introduced. A first step consists in organizing the concepts in a taxonomic

³ Notice that *process* is the category term. Such a category, from an ontological point of view, is representative of other concepts, such as phases, activities, and operations.

| OPAL Message semantics | Description | Example of associated BOD |
|------------------------------|---|---|
| Agree | The action of agreeing to a previous assertion or to perform some action | Purchase order acceptance |
| Call for Proposals | The action of calling for proposals to perform a given action | Request for quote |
| Cancel | The action of cancelling some previously requested action which has temporal extent (i.e. is not instantaneous) | Purchase order cancellation |
| Confirm | The sender informs the receiver that a given proposition is true, where the receiver is known to be uncertain about the proposition | Delivery receipt, purchase order ack |
| Inform | The sender informs the receiver that a given proposition is true | Price list, shipment confirmation |
| Propose | The action of submitting a proposal to perform a certain action, given certain preconditions | Quote, purchase order update |
| Refuse | The action of refusing to perform a given action and explaining the reason for the refusal | Purchase order rejection |
| Request | The sender requests the receiver to perform some actions. One important class of uses of the request act is to request the receiver to perform another communicative act | Purchase order, purchase order change, invoice, sale order |

List of attributes

| Account name | Identifier authority | Post office box number |
|-------------------------|-----------------------|---------------------------|
| Account number | Language | Price description |
| Account type | Line ID | Product attribute |
| Address type | Line item amount | Product attribute |
| Amount | Line status | Description |
| Building number | Location | Product code |
| City | Location description | Product name |
| Code | Location type | Product quantity |
| Completion date | Measure unit | Proprietary product ID |
| Contact name | Name | Quantity |
| Contact role ID | Number of item | Region |
| | packages | |
| Country code | Package material | State |
| Country name | Package type | Street name |
| County | Packaging description | Sub total price |
| Credit card expiry date | Partner information | Taxable amount |
| Currency | Party description | Tax amount |
| Currency exchange | Party ID | Tax category |
| rate | | |
| Date | Party name | Tax description |
| Delivery date | Party Tax ID | Tax rate |
| Discount percent | Payment method | Telephone number |
| Document date | Payment term | Text |
| Document ID | Percentage | Time |
| Document reference | Person first name | Total price |
| Effective date | Person full name | Transportation term |
| Email address | Person middle name | Transport mode |
| Exchange rate date | Person surname | Unit price |
| Fax number | Postal address | URI |
| ID | Postal code | Version ID |

hierarchy according to the *generalization* (i.e., *IsA*) relation. To this end, three main approaches are known in the literature [17]: *top-down* (from general to particular), *bottom-up* (from particular to general) and *middle-out* (or combined). The combined approach consists in finding first the salient concepts (typically placed in a middle area) and then generalizing and specializing them. This approach is considered to be the most effective because concepts "in the middle" tend to be more informative about the domain.

The resulting taxonomy can be extended with other relations, i.e., *part-of* and domain-specific relationships. The outcome of this step is a *SN*,⁴ that is represented according to UML class diagram, in particular using *generalization* (IsA), *aggregation* (part-of) and *association*. Therefore the *SN* can be represented in a diagrammatic format (see Figs. 11 and 12 for examples). Finally, this workflow eventually provides a diagrammatic representation of the ontology in the form of a set of UML class diagrams.

Below we formally define a SN, according to our methodology.

Definition 6. Given a finite set *C* of concepts and a finite set *R* of relationships established among concepts, a *SN* is defined as

SN = (C, R)

 $C = \{c_i\}, \text{ being } i \in N$

 $R = IsA \cup De \cup DR = \{ \langle c_i, c_k \rangle \} \subseteq C \times C \text{ and } i, k \in N$

where

 $IsA = \{ \langle c_i, c_k \rangle | gen(c_i, c_k) \},\$

 $De = \{ \langle c_i, c_k \rangle | partOf(c_i, c_k) \},\$

 $DR = \{ \langle c_i, c_k \rangle | rel(c_i, c_k) \}.$

where *gen*, *partOf*, and *rel* are, respectively, the *generalization*, the *aggregation*, and the *association* relationships.

3.4. The implementation workflow

The purpose of this workflow (Fig. 13) is to encode the ontology in a rigorous, formal language. For this reason, the activities of this workflow are mainly performed by KEs as shown in the expert involvement schema of Fig. 13.

When selecting a formal language to encode the ontology it is necessary to consider the expressive power, the computational complexity of the associated reasoning method, and level of acceptance within the community. As a result of a long standardization effort, the Web Ontology Language (OWL) [15] [WR9] is currently the main candidate for encoding an ontology to be used on the Semantic Web.

The output of this workflow is therefore the OWL implementation of the ontology (in particular, we adopted OWL-DL).

⁴ Notice that we consider a semantic network as the informal counterpart of the ontology that will be produced in the next workflow.



Fig. 11. Examples of ontological relationships in the eProcurement application.

In the eProcurement example the OWL code, reported in Fig. 14, has been produced, in the context of the Athena Integrated Project [WR2], by using the ontology management system ATHOS [WR10]. KEs and DEs adopted ATHOS since it is based on OPAL and it is implemented as a web application, remotely accessible by using a web browser as a client. In addition to concepts and relationships, an ontology is characterized by a set of axioms. An axiom is essentially a constraint imposed by the modeled reality. It can be associated to the structural component (e.g., cardinality constraints on an attribute), to the hierarchy (e.g., disjunction of two subconcepts), or stand-alone, expressed by a specific language (e.g., as a boolean expression).

Please note that in OWL-DL only two levels of abstractions are allowed. Consequently, in order to represent concepts categories (e.g., actor, BOD, process) in OWL-DL, we use top level superclasses and the *subClassOf* construct.

Below the formalization of an ontology is presented.

Definition 7. Given a finite set *C* of concepts, a finite set *R* of relationships established among concepts, and a finite set of semantic axioms *Ax*, an *Ontology O* is defined as

O = (C, R, Ax)

 $C = \{c_i\}, \text{ being } i \in N$

where

 $R = IsA \cup De \cup DR = \{ \langle c_i, c_k \rangle \} \subseteq C \times C, \text{ and } i, k \in N$

 $Ax = \{boolExp_i\}, being j \in N$

$$R = IsA \cup De \cup DR$$

3.5. The test workflow

According to [4], the quality of an ontology is a multidimensional feature. In particular, an ontology should be evaluated with respect to four different characteristics:

- syntactic quality: that measures the quality of the ontology according to its formal style, the way it is written;
- semantic quality: where the primary concern is the absence of contradictory concepts;
- pragmatic quality: that refers to the ontology content and usefulness for users, irrespectively of its syntax and semantics;
- social quality: that reflects more general criteria, for instance the numbers of other ontologies that link to it, e.g., by defining their terms using its definitions, and the number of times it is accessed (when public) from within and outside of the community that manages it.

The test workflow (Fig. 15) is conceived to verify the semantic and pragmatic quality of the ontology, since syntactic quality is checked in the previous workflow and social quality can be checked only after its publication. In particular, syntactic quality is guaranteed during the OWL



Fig. 12. Processes identified for the eProcurement application, with PartOf structure.



KE: Knowledge Engineer

Fig. 13. An overview of the implementation workflow.



Fig. 14. An OWL fragment of the eProcurement ontology.



Fig. 15. An overview of the test workflow.

coding and social quality is assured by the interaction with different teams of KEs and cross-DEs.

Semantic quality can be mainly verified by checking the consistency of the ontology, achieved by using a reasoner, such as *Racer* [18] or *Pellet* [WR11]. Consistency, besides absence of contradiction, requires also that the modeling

constructs are correctly used (e.g., absence of cycles in the specialization hierarchy or the fact that classes and properties are disjoint [19]). The activity concerning consistency checking is mainly performed by KEs, as shown in the expert involvement schema of Fig. 15, since the use of a reasoner requires technical skills.

Pragmatic quality is related to three features: (i) *fidelity*, (ii) *relevance*, and (iii) *completeness*.

(i) Fidelity can be measured by checking if the claims an ontology makes are true in the target domain. This task can be achieved, for instance, by verifying the references to the sources used in the descriptions of terms. (ii) *Relevance* is checked in conjunction with completeness, verifying the correct implementation of ontology's requirements, gathered in the first workflow. This can be achieved, for instance, by performing the following two kinds of test. The first concerns the coverage of the ontology over the application domain. For instance, a DE is asked to semantically annotate the UML diagrams. modeling a software application, with the ontology concepts. The second test concerns the CQs and the possibility to answer them by using the ontology content. For instance, in the eProcurement application such a test gives a positive result, since it is possible to answer all the CQs listed in the example. For instance, CQ1: "what are the documents that a company receives before a purchase order?" is elaborated using the ontology concepts "Request For Quotation", "Processing RFQ", "Sending RFQ". This test is particularly useful for ontologies supporting the semantic search and discovery of web resources. (iii) Completeness: according to [20] a design artefact, like an ontology, is complete and effective when it satisfies the requirements and the constraints of the problem it was meant to solve. For this reason, the completeness of an ontology can be assessed by using the domain goals and again the coverage and CQs. Consequently an ontology is complete if the objectives, defined in the requirements workflow, are reached (e.g., semantic search and retrieval, reconciliations of data messages and processes in the eProcurement application).

As shown in the involvement schema of Fig. 15, the activities, coverage verification and answering CQs, are performed by DE with a strong support of KE when his technical expertise is needed (e.g., during semantic annotation of UML diagrams).

Finally, Fig. 16 reports a complete overview of the UPON cycle. Table 10 illustrates, for the eProcurement example, the evolution from lexicons to ontology, indicating the corresponding output sizes during the first cycle of activities.

4. Related work and assessment

The first contributions to ontology building methods are due to [7,8,17,21], representing the basis for many subsequent proposals. Gruber's seminal work [21] discusses some basic ontology design criteria: some related to the quality of ontology building methodology (clarity and ontological commitment) and some related to the quality of the built ontology (coherence, extendibility, and minimal encoding bias). Gruninger and Fox [7] provide a skeletal methodology for ontology building based on CQs, while Uschold and King [8] present a method based on four main activities: identification of the purpose of the ontology, building activity, evaluation, and documentation. A complete ontology development process, *METHON-TOLOGY*, is proposed by Fernández et al. [22]. The process is composed by the following phases: *specification, conceptualization, formalization, integration, implementa-tion,* and *maintenance*. Its life cycle is based on evolving prototypes and specific techniques peculiar to each activity. Other activities, like knowledge acquisition, control, evaluation, documentation, and quality assurance are carried out simultaneously with the ontology development activities.

With a strong emphasis on knowledge maintenance and management, Sure et al. [23] propose *On-To-Knowledge*, an ontology development process consisting of six main phases: *feasibility study*, *kick-off*, *evaluation*, *refinement*, *application*, and *evolution*. Other approaches, often tied to industry or research projects, include the methods used for building *CyC*, *SENSUS*, and *KAKTUS* [24]. A complete overview of ontology building methods is provided by Corcho et al. [25].

In the following, a two-fold evaluation of UPON is provided. Firstly, a comparative evaluation with respect to the methodologies introduced above is presented. Secondly, the experience in using the UPON process in building an ontology of eProcurement for the Athena Integrated Project is briefly reported.

In order to evaluate a number of different ontology building processes, Fernández and Gómez-Pérez [26] presented a framework based on the comparison with respect to the IEEE 1074-1995 standard for software development life cycle. This framework is used to assess UPON with respect to the other proposals.

The IEEE 1074-1995 standard, applied to ontologies, distinguishes three kinds of processes:

- project management processes: concerning the creation of a project management framework for the entire ontology life cycle;
- ontology development processes, including a predevelopment process (an environment study and a feasibility study), a development process (requirements, design, implementation), and a post-development process (installation, operation, support, maintenance, and retirement of an ontology);
- integral processes, including processes of knowledge acquisition, evaluation, configuration management, documentation, and training.

Table 11 compares the support provided by each methodology to the processes indicated by the IEEE 1074-1995 standard. Features are marked as *supported*, *unsupported*, or *partially supported*; the latter is used when a feature is mentioned in the methodology literature but not fully supported in practice. UPON provides full support to the ontology development process, but also to the knowledge acquisition and evaluation (through use-case testing), and the production of documentation (a feature intrinsic to the nature of our methodology).

The UPON methodology is focused on the core activities of ontology building, therefore project management and many pre/post-development activities, like feasibility study, and maintenance, fall outside of its



Fig. 16. UPON process.

scope. Conversely, such activities are mainly addressed by the On-To-Knowledge approach. An important feature is that the adoption of UPON does not require any learning curve for DE familiar with UML and the UP, because it is just an adaptation of the UP to ontology building. This is an advantage also over the METHONTOLOGY, that roughly covers the same development processes as UPON. There is an extension of the UP, the *Enterprise Unified Process* (EUP), proposed by Ambler et al. [27], developed with the aim of taking project management and all the other

eProcurement application: evolution from lexicons to ontology

| Application lexicon | Domain lexicon | Reference lexicon | Reference glossary | Semantic network | Ontology |
|------------------------|-------------------|----------------------|------------------------|--|--------------------------------------|
| 213 terms | 114 terms | 139 terms | 139 terms+descriptions | 139 terms+descriptions+411 relations | 139 concepts+411 relations+10 axioms |

Table 11

A comparison of different processes with respect to the IEEE 1074-1995 standard

| IEEE 1074-1995 standard | d processes | | Uschold and King | Grüninger and Fox | METHONTOLOGY | On-To- Knowledge | UPON |
|---|---|---|----------------------------|------------------------|-------------------------------------|--------------------------------|-----------------------------------|
| Project management processes | Project initiation Monitoring and control Quality management | | | Р Р — | — P P | + + - | - - - |
| Ontology development-oriented processes | Pre-development Development Post-development | Environment study Feasibility study Requirements Design Implementation Installation Operation Support Maintenance Retirement | - P - + - - | + + | + + + - - - P | + + - - - P | P + + - P |
| Integral processes | Knowledge acquisition Evaluation Configuration managen Documentation Training | nent | + + - + - | P | + + + P | - + - - | + + - + P |

-, unsupported; +, supported; P, partially supported.

pre/post-development activities into account. In the future we will consider the integration of UPON according to EUP, to cover these other aspects.

Another significant advantage of UPON over the other methodologies is that, given its strong rooting in UML, diagramming, documentation, and versioning can be performed with the aid of a variety of UML tools, like Rational Rose, Microsoft Visio, etc.

UPON has been adopted in the context of the Athena, a European IST Integrated Project, for building an ontology of eProcurement. In this first experimentation, both DEs and modelers, after an extensive use of UPON expressed their appreciation. As reported in the section related to the description of the workflows, the developed ontology consists of 139 concepts: 15 business actors, 19 business processes, 14 BODs, 8 messages, and 83 attributes, complex or atomic. Though it may seem a "small ontology", it is appropriate for the purposes, essentially the achievement of business messages interoperability. In particular, it allows the semantic annotation of the main business documents (e.g. the purchase order and the invoice) used in a purchasing transaction. Thanks to the characteristics of the methodology and of the ATHOS ontology management system, its scalability is also ensured.

5. Conclusions and future work

In this paper, an ontology building methodology, UPON, based on the software engineering UP, has been presented. Building an ontology is different from developing a software system, but the underlying principles and the basic phases are the same; furthermore, diagrammatic specifications can be used for each phase of the lifecycle of both software systems and ontologies.

The strength of the proposed approach lies in the UP being a highly scalable and customizable methodology. It can indeed be tailored to fit a number of variables: the ontology size, the domain of interest, the complexity of the ontology to be built, the experience and skill of the project experts and their organization. Furthermore, modelers can decide to adapt the scheme here presented to one of the methodologies derived from the UP (like the Rational Unified Process).

We compared UPON against the main ontology building methodologies, showing that, within the scope, its features are aligned, and sometimes outperform the best solutions. Furthermore, UPON has been experimented by several European projects (Athena being the most noticeable) and some industrial contexts. The experiments demonstrated its applicability and high degree of acceptance by both business people and ontology engineers. In the future we intend to proceed along two lines in parallel. On the one hand, to adopt UPON in new industrial applications (we are applying it to the construction of an ontology to support the automation of software project management) to acquire additional validation cases. A second line is represented by automatic support of the methodology. With this respect, we will carefully consider the case of embedding (part of) it in the new functionalities of ATHOS, the Ontology Management System developed in our laboratory.

Acknowledgments

This work is partially supported by the Interop NoE and Athena IP, both are *6th European Union Framework Programme*. Furthermore, we wish to thank Athena partners and, in particular, María José Núñez (AIDIMA), for their support and their constructive objections that helped us in improving our work.

References

- T. Berners-Lee, J. Hendler, O. Lassila, The Semantic Web, Scientific American, 2001 May, pp. 28–37.
- [2] R. Navigli, P. Velardi, Learning domain ontologies from document warehouses and dedicated websites, Computat. Linguist. 30 (2) (2004) MIT Press.
- [3] I. Jacobson, G. Booch, J. Rumbaugh, The Unified Software Development Process, Addison Wesley, USA, 1999.
- [4] A. Burton-Jones, et al., A semiotic metrics suite for assessing the quality of ontologies, Data Knowl. Eng. 55 (2005).
- [5] G. Guizzardi, H. Herre, G. Wagner, Towards ontological foundations for UML conceptual models, in: First International Conference on Ontologies, Databases and Application of Semantics, Irvine, CA, USA, 2002.
- [6] J. Krebs, Form feeds function: The role of storyboards in requirements elicitation, 2005 < http://www-128.ibm.com/developerworks/rational/ library/dec05/krebs/index.htm >.
- [7] M. Gruninger, M.S. Fox, Methodology for the design and evaluation of ontologies, in: Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing in IJCAI 95, Montreal, Canada, 1995.
- [8] M. Uschold, M. King, Towards a methodology for building ontologies. Workshop on Basic Ontological Issues in Knowledge Sharing in IJCAI 1995, Montreal, Canada, 1995.
- [9] N. Guarino, M. Carrara, P. Giaretta, Formalizing Ontological Commitments, in: Proceedings of AAAI 94, vol. 1, 1994, pp. 560–567.
- [10] M. Missikoff, F. Taglino, An ontology-based platform for semantic interoperability, in: S. Staab, R. Studer (Eds.), Handbook on Ontologies, Springer, 2004, pp. 617–633.
- [11] A. Gómez-Pérez, M. Fernández-Lopez, O. Corcho, Ontological Engineering, Springer, London, 2004.
- [12] F. D'Antonio, M. Missikoff, F. Taglino, Formalizing the OPAL eBusiness ontology design patterns with OWL, in: Third International Conference on Interoperability for Enterprise Applications and Software, I-ESA 2007.

- [13] S. Plous, The Psychology of Judgment and Decision Making, McGraw-Hill, New York, 1993.
- [14] M. Missikoff, F. Taglino, Business and enterprise management with SymOntoX, in: First International Semantic Web Conference, Sardinia, Italy, 2002.
- [15] G. Antoniou, F. van Harmelen, Web ontology language: Owl, in: S. Staab, R. Studer (Eds.), Handbook on Ontologies, Springer, 2004, pp. 67–92.
- [16] J. Sowa, Knowledge Representation: Logical, Philosophical and Computational Foundations, Brooks/Cole, USA, 1999.
- [17] M. Uschold, M. Gruninger, Ontologies: principles, methods and applications, Knowl. Eng. Rev. 11 (2) (1996).
- [18] V. Haarslev, R. Möller, Description of the RACER system and its applications, in: Proceedings of the International Workshop on Description Logics (DL-2001), Stanford, USA, 2001.
- [19] N. Ide, J. Véronis, Extracting Knowledge Bases from Machine-Readable Dictionaries: Have We Wasted Our Time?, KB&KS '93 Workshop, Tokyo, 1993, pp. 257–266.
- [20] A. Hevner, S. March, J. Park, S. Ram, Design science in information systems research, MIS Q. 28 (1) (2004).
- [21] T.R. Gruber, A translation approach to portable ontology specification, Knowl. Acquis. 5 (1993) 199–220.
- [22] M. Fernández, A. Gómez-Pérez, N. Juristo, METHONTOLOGY: from ontological art towards ontological engineering, in: Symposium on Ontological Engineering of AAAI. Stanford, CA, 1997.
- [23] Y. Sure, S. Staab, R. Studer, On-To-Knowledge Methodology (OTKM). Handbook on Ontologies, Springer, 2004, pp. 117–132.
- [24] OntoWeb Deliverable 1.4: A Survey on Methodologies for Developing, Maintaining, Evaluating and Reengineering Ontologies (2002). < http://ontoweb.aifb.uni-karlsruhe.de/About/Deliverables/ D1.4-v1.0.pdf>.
- [25] O. Corcho, M. Fernández, A. Gómez-Pérez, Methodologies, tools and languages for building ontologies. Where is the meeting point?, Data Knowl. Eng. 46 (2003).
- [26] M. Fernández, A. Gómez-Pérez, Overview and analysis of methodologies for building ontologies, Knowl. Eng. Rev. 17 (2) (2002).
- [27] S.W. Ambler, J. Nalbone, M.J. Vizdos, The Enterprise Unified Process, Prentice-Hall, Englewood Cliffs, NJ, 2005.

Web references

- [WR1] Unified Modeling Language (UML), version 2.0: http://www.uml.org>.
- [WR2] Advanced Technologies for Interoperability of Heterogeneous Networks and their Application, Integrated Project 507849, 6th EU FP: < http://www.athena-ip.org >.
- [WR3] <http://www.aidima.es/aidima>.
- [WR4] < http://dictionary.bnet.com/definition/E-procurement.html >.
- [WR5] Electronic Business using eXtensible Markup Language (ebXML): <http://www.ebxml.org>.
- [WR6] RosettaNet: < http://www.rosettanet.org >.
- [WR7] Open Applications Group (OAGIS): < http://www.openapplications. org >.
- [WR8] Foundation for Intelligent Physical Agents (FIPA): <http://www.fipa.org>.
- [WR9] Ontology Web Language (OWL): <http://www.w3.org/TR/ owl-features>.
- [WR10] ATHOS ontology management system: <http://leks-pub.iasi.cnr. it/Athos>.
- [WR11] Pellet: < http://www.mindswap.org/2003/pellet >.
- [WR12] Text-To-Onto: < http://sourceforge.net/projects/texttoonto >.