An ontology engineering methodology for DOGMA

STAR Lab Technical Report

2008

Peter Spyns, Yan Tang & Robert Meersman

affiliation: 
corresponding author: Peter Spyns
keywords: ontology engineering methodology
number: STAR-2008-x1
date: 05/12/2008
status: final
An ontology engineering methodology for DOGMA

Peter Spyns*, Yan Tang and Robert Meersman

STAR Lab., Vrije Universiteit Brussel, B-1050 Brussel, Belgium
E-mail: {first_name.second_name}@vub.ac.be

Abstract. Although ontologies occupy a central place in the Semantic Web and related research domains, there are currently not many fully fledged ontology engineering methodologies available. In this paper, we want to present an integrated methodology for ontology engineering from scratch, inspired by various scientific disciplines, in particular database semantics and natural language processing.

Keywords: Methodology, ontology engineering, semantics, knowledge modelling

1. Introduction

1.1. Preliminary remarks

The main goal of this paper is to describe an integrated ontology engineering methodology based on the research efforts as performed at VUB STAR Laboratory over the past ten years. Depending on application domains, project contexts and scientific inspirations, several methods and tools for ontology engineering have been explored. We felt it has become worthwhile to consolidate them into an overarching methodological framework. In addition, we want to present the ontology engineering methodology in a didactic way that any skilled computer scientist is able to learn and apply it him/her self. Therefore, we want the methodology to be teachable and repeatable (Meersman, 2001b).

1.2. DOGMA: What’s in a name

DOGMA\(^1\) (Developing Ontology-Grounded Methods and Applications) as an ontology engineering framework has its roots in database semantics and model theory. Some of its important underlying assumptions coincide with some of Guarino’s insights on formal ontology engineering – cf., e.g. Guarino & Giaretta (1995). The roots of DOGMA can be retraced to five initial key publications that already at the moment of publication contain the core vision of the research programme of the following

\(^*\)Corresponding author: Peter Spyns, STAR Lab., Vrije Universiteit Brussel, Pleinlaan 2, Gebouw G-10, B-1050 Brussel, Belgium. E-mail: peter.spyns@vub.ac.be.

\(^1\)It goes without saying that as a researcher one cannot be dogmatic in the religious sense of the word – even if we distinguish seven dogmas in this section. Any researcher of the Free University of Brussels has subscribed literally to the fundamental principle of free inquiry and research, namely that science should be based only on facts, thus rejecting any preconceived idea, passion, interest or dogma – although at STAR Laboratory we make an exception for DOGMA.

1570-5838/08/$17.00 © 2008 – IOS Press and the authors. All rights reserved
decade (Meersman, 1996, 1999a, b, 2001a, b). The fundamental ideas put forward in these papers still
stand although they underwent refinements and extensions as a result of research results and the suc-
cession of (senior) researchers with various interdisciplinary backgrounds in the laboratory over the
past 10 years. A recent formalisation of DOGMA can be found in (De Leenheer & Meersman, 2005).
A short-list of the most important DOGMA principles is given below.

1. The meaning of data is to be decoupled from the data itself, stored in a specific repository separated
from the data repository, and managed by proper methods. This is the most important “dogma”,
based on the transposition of the principle of data independence (as applied in modern databases)
into the principle of meaning independence (to be applied for ontologies) (Meersman, 1999b). An
application commits its local vocabulary to the meaning of the ontology vocabulary.
A similar three-tier structure (as used for database design) holds for ontology management: a
conceptual level to model the application domain or domain of discourse, a logical level that
fits the conceptual model into some modelling paradigm (e.g., TopicMaps, the triple-model, ...)
and an implementation level that contains the actual implementation in some specific language
and/or product (e.g., KIF, KL-ONE, OWL languages, F-Logic, DOGMA (meta-)lexons, Conceptu-

al Graphs, ...).

2. An ontology is based on many interpretation-independent plausible fact types about a universe
of discourse. They constitute the DOGMA ontology base. In DOGMA, lexons are a more formal
but still linguistically determined representation of propositions about a domain to be modelled.
They can result from different – even conflicting – situations and points of view on the universe
of discourse. A lexon can be added to the DOGMA ontology base at any time. Hence, except for
an intuitive linguistic interpretation, lexons have no formal interpretation or semantics (Meersman,
2001b). “Contexts” have been introduced in the ontology base as an organising principle, grouping
related lexons. A context can be considered as an identifier of a possible world, leading to specific
possible world semantics (e.g. Kripke models) and mappings between them (see also Sowa, 2000).

3. It is possible to have multiple views on and uses of the same stored conceptualisation. The “mean-
ing space” can be subdivided in different ways according to the needs of a specific application that
will add its particular semantic restrictions according to its intended usage (= an instance of a first
order interpretation) (Meersman, 1999a). A DOGMA ontology consists of an ontology base layer
and a commitment layer (Jarrar et al., 2003). Fuzzy intuitive meaning is separated from precise
formal meaning.
This ontology engineering principle allows for scalability in representing and reasoning about for-
mal semantics, as a generative approach can be adopted. With a reduced set of meaning building
blocks, a large space of meaning combinations can be generated. In analogy with a principle stem-
mimg from the linguistics fields, this has been dubbed the double articulation principle (Spyns,
2005b; Spyns et al., 2002). This practical distinction between an ontology base layer and a com-
mmitment layer corresponds to a certain extent to Guarino’s theoretical distinction between the con-
ceptual and ontological levels. The ontological level is the level of commitments, i.e. formal restric-
tions on the semantics of the primitives while the definite cognitive interpretation of a primitive is
situates on the conceptual level (Guarino, 1995; p. 633).

4. An ontology is language neutral. We concur with Guarino’s definition that:

An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e.
its ontological commitment to a particular conceptualisation of the world. The intended models
of a logical language using such a vocabulary are constrained by its ontological commitment.
An ontology indirectly reflects this commitment (and the underlying conceptualisation) by approximating these intended models (Guarino, 1998; p. 7).

However, we state that an ontology is language neutral – as put forward by Nirenburg & Raskin (2001) – rather than language dependent as stated by Guarino. The logical (and hence unambiguous) vocabulary of a formal KR language should be clearly distinguished from potentially ambiguous natural language vocabulary expressing the intended meaning (Spyns & De Bo, 2004). In the DOGMA framework, together with a context identifier (Meersman, 1999a), the language identifier plays a disambiguating role in determining which concept is linked with which word or natural language term (De Bo et al., 2003). Meta-lexons are the language-neutral and disambiguated counterparts of the language dependent lexons.

5. As foundation of an ontology modelling methodology, we choose the Object Role Modelling (ORM) (Halpin, 2001) method complemented with Nijssen’s Information Analysis Method (NIAM – the precursor of ORM) (Wintraecken, 1990) because of the latter’s emphasis on natural language as starting point and elicitation vehicle of the modelling exercise (Meersman, 2001b). The preference for an analysis method that starts from natural language is a consequence of the point of view that all meaning (semantics) must be described and rooted in natural language; otherwise humans cannot agree on common semantics (Meersman, 2001b). In addition, domain experts should not be bothered with how to think in a new formal language: adequately capturing and organising domain knowledge is a task sufficiently demanding as it is.

6. We focus on intensional semantics instead of extensional semantics. An ontology is not affected by a change on the instance level as long as the corresponding concept and conceptual relationships are not modified. As a corollary, value types (as used in ORM (Halpin, 2001)) or lexical object types (LOTs in NIAM parlance (Wintraecken, 1990)) are not part of an ontology. Said in another way, an ontology does not contain instances. An ontology is in principle application independent and rather represents a domain (Meersman, 1999b). An ontology can be described as a “fat” domain model compared to a parsimonious and optimised application model of a database schema (Spyns, 2005b). Hence, applications first make a selection of the ontology vocabulary they want to commit their local vocabulary to. Note that ORM and NIAM had to be adapted at some points to serve as ontology modelling methodology (Spyns, 2005a).

7. Meaning will be negotiated between the relevant stakeholders (or imposed by an authoritative party and accepted as such by the other stakeholders) before any meaning exchange can happen. Absolute meaning does not exist. Hence, research on negotiation (de Moor, 2005a), interaction patterns (de Moor & Weigand, 2005), pragmatics (de Moor, 2005b) and the like (mostly stemming from social sciences) have been taken into account. A community layer has been included in DOGMA. It focuses on the efficient negotiation of the meaning of concepts relevant to a community of stakeholders (de Moor et al., 2006). Experience has shown that it is easier to agree on simple concepts and conceptual relationships rather than on semantic constraints or other more complex structures (Meersman, 1996). Constraints are much more context and application dependent than the concepts.

The above mentioned principles have been orthogonally combined and applied to modelling domains such as financial fraud, European privacy and VAT directives, customer complaints management, administrative and health record patient data, vocational competencies and job announcement matching, professional bakery education, innovation information look-up, medical and biomedical data mediation to mention the most important ones.
One can deduce from the paragraphs above that the theoretical basis (and related practical experiences) of DOGMA is too broad to be extensively described in this paper alone. The focus on the remainder of this paper consequently lies on the ontology engineering methodology and work flow (Section 2). The related work and discussion Section (3), for the same reason of space limitations, will be rather general.2 A conclusion (Section 4) ends the paper.

2. A collaborative ontology engineering methodology

2.1. Preliminary remarks

In the following sections, we present the outline of our ontology engineering methodology. The section is structured along the work flow an ontology engineer has to adopt when modelling in the DOGMA-style. As it is impossible to describe in full detail all the ins and outs of this methodology in the limited space of this paper, a (seemingly arbitrary) selection has been made of certain parts of the methodology that are described more extensively. The methodology consists of two main steps:

1. Preparatory stages
   - Section 2.2 (formulate vision statement): define the overall purpose,
   - Section 2.3 (conduct feasibility study): find out if the overall effort is worthwhile,
   - Section 2.4 (project management): organise and plan the effort,
   - Section 2.5 (preparation and scoping): collect the required “materials”.

2. Ontology engineering stages (in accordance with dogma 3)
   - Section 2.6 (domain conceptualisation): collect, define and describe (cf. dogma 4) in agreement (cf. dogma 7) the set of plausible domain fact types (cf. dogma 2),
   - Section 2.7 (application specification): select and formally constrain plausible fact types (cf. dogma 5) as needed by an application committing to their agreed meaning (cf. dogma 1).

The bulk of the description will be devoted to preparatory activities (Section 2.5), the actual domain conceptualisation methods (Section 2.6) and the application specification techniques (Section 2.7) as these constitute the core of the ontology engineering effort. The project management (Section 2.4) is more a continuous meta-activity. The vision statement formulation (Section 2.2) and feasibility study (Section 2.3) create and investigate the possibilities, necessities and enabling conditions. The latter three activities are only succinctly mentioned to complete the picture. As the arrows in the various figures below show, the process is not a mere linear one.

2.2. Formulation of the vision statement

A vision is a compelling and inspiring view of a desired and possible future. The stakeholders develop a shared vision and common values. The vision statement is drawn on a high “executive” level. It shows the expectations from the stakeholders and provides a picture of the long-range results of the planned process and the future accomplishments when the strategies put forward are actually implemented. Several questions can be involved when drawing the vision statement. Table 1 shows the (empty) form that should contain the result of the ontology vision statement phase.

2A comparison of our methodology with other ontology modelling methodologies will be the subject of a follow-up paper.
Fig. 1. High level decomposition of the ontology engineering methodology.

Table 1
Empty ontology vision statement form

<table>
<thead>
<tr>
<th>Title</th>
<th>&lt;Title of this document&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>&lt;Name of the author of this document&gt;</td>
</tr>
<tr>
<td>Creation date</td>
<td>&lt;Creation date and time of this document&gt;</td>
</tr>
<tr>
<td>Last update date</td>
<td>&lt;Date and time of the last modification&gt;</td>
</tr>
<tr>
<td>Responsible person(s)</td>
<td>&lt;List of responsible persons involved&gt;</td>
</tr>
<tr>
<td>Coverage time</td>
<td>&lt;Estimate time needed for the whole project&gt;</td>
</tr>
<tr>
<td>Requirements</td>
<td>&lt;List of stakeholder requirements&gt;</td>
</tr>
<tr>
<td>Domain</td>
<td>&lt;Description of the domain of the ontology&gt;</td>
</tr>
<tr>
<td>Ontologies</td>
<td>&lt;Name and short description of the ontologies involved&gt;</td>
</tr>
<tr>
<td>Applications</td>
<td>&lt;Name, type and description of applications involved&gt;</td>
</tr>
<tr>
<td>Technologies</td>
<td>&lt;Technologies that have to be applied&gt;</td>
</tr>
</tbody>
</table>

- Why do we want to build an ontology?
- What is the domain of the ontology? What will be its purpose, its context?
- Which are the applications involved?
- What is the necessary/basic technology (technologies) that shall we consider?
- Who are the (responsible) people involved in the whole project?
- What is the timeframe foreseen to create an ontology?
2.3. Feasibility study

The feasibility study is a mechanism to refine the vision statement and to find out whether a project is actually worthwhile in terms of expected costs and benefits, technological feasibility and needed (and committed) resources (Schreiber et al., 1999; p. 26). Finally, a go or no-go decision is to be taken. A feasibility report not only provides recommendations of how to refine the vision statement, but also the material and rationale underpinning it.

2.4. Project management

This stage is actually the initialisation of management activities. Project management activities will be on-going during the whole period of the ontology capture and development process. A project plan (including a Gantt chart) should be produced during this stage detailing at least the following items:

- Time frame of every activity;
- Task resource assignment for every activity;
- Project team and responsible person(s) of every activity;
- Planning management.

2.5. Preparing and scoping

The preparation and scoping activity delineates more sharply the domain of the future ontology. It scales down the problem domain in order to reach the final goal more easily. As it has a strong impact on the quality of the final ontology, we should pay sufficient attention to this activity. The steps shown in Fig. 2 result in a completed ontology scoping form (see Table 2).

![Fig. 2. Steps of the knowledge preparation and scoping stage.](image)

<table>
<thead>
<tr>
<th>Table 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empty ontology scoping form</td>
</tr>
<tr>
<td><strong>Ontology scoping form</strong></td>
</tr>
<tr>
<td><strong>Title</strong></td>
</tr>
<tr>
<td><strong>Theme</strong></td>
</tr>
<tr>
<td><strong>Author</strong></td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
</tr>
<tr>
<td><strong>Requirements</strong></td>
</tr>
<tr>
<td><strong>Experts</strong></td>
</tr>
<tr>
<td><strong>Sources</strong></td>
</tr>
</tbody>
</table>
Of the steps displayed by Fig. 2, only the knowledge resource scoping step will be presented in more detail. It consists of two subprocesses, namely the selection of relevant passages (Section 2.5.1) and the definition of scenarios (Section 2.5.2).

It is possible that, once the domain experts have been identified, “classical” brainstorming and knowledge elicitation techniques (e.g., see Schreiber et al., 1999, Chapter 6) are preferred during the domain conceptualisation phase (see Section 2.6). In this case the knowledge scoping step can be skipped.

The way how this step is executed greatly depends on the type of material or resources that are available to the modellers. E.g., a collection of texts will be handled in a different way than when a database schema constitutes the working material. Especially, if the material will be processed automatically (e.g., by means of natural language processing technologies), this step is meant to prevent or at least reduce the generation of irrelevant information. In the following sections, we will mainly describe methods suited for creating ontologies on the basis of textual material.

2.5.1. Select relevant passages

By including irrelevant material in a corpus to be processed automatically, there is a greater chance that irrelevant but formally correct output will be generated. Tests using an unsupervised text miner to generate triples (see Spyns & Hogben, 2005, for the details) have been evaluated by domain experts. They have formulated the following method to weed out superfluous material.

– Separate the document into “core text” and “explanatory text”;
– Separate out definitions;
– Separate out paragraphs that are clearly not relevant to any potential application;
– From the “core text” try to pick out any parts which might possibly put a constraint or obligation on an application (without having any particular application in mind);
– Find exceptions to these constraints. The exceptions should be derived from the text (or possibly from the expert’s knowledge).

If different experts are involved, their selections of passages are to be merged and processed together (the overlapping parts and term duplications serve as an indication of the importance of certain terms). The output of this step is a collection of text passages. Getting rid of irrelevant passages is also helpful in case of manual processing (see below).

2.5.2. Define scenarios

The concept of a NS (stands for Narratological Scheme coming from the literary analysis research domain) is derived from stories (sometimes also called storytelling) that have existed for hundreds of

3 Experiments on the EU Privacy Directive using selected and merged passages have been performed. A report on this issue will follow.
years as a means of exchanging (cultural) information. A story has become a favoured technique among
management scientists and consultants. A NS acts a “container” or frame representing many aspects of a
story in a structured way. NSs allow tracing changes. They communicate complicated ideas and concepts
in an easy-to-understand form and convey tacit knowledge that is difficult to articulate.

By drafting a NS, or detailed use cases, the domain experts decompose the domain of interest into
a number of smaller and more manageable micro-domains. In order to limit the cognitive overhead of
the domain experts and to facilitate communication, stories are specified in structured natural language.
NSs should be compiled by referring to the reference knowledge resources. Of course, the tacit and
implicit knowledge of a domain expert will also influence how a NS is completed. This allows ontology
engineers to focus on smaller but well-scoped descriptions.

Since the ontology engineer will be concentrating on NSs alone, domain experts need to ensure that
the micro-domains are described in a fashion that is thorough and complete (anything not mentioned,
will not be included in the ontology). The aim is to explicitly define boundaries for the conceptual scope
of the micro-domain being described. In addition to the cognitive advantage of dividing the domain
of interest into more manageable parts, it is believed that such micro-domains also support distributed
collaboration: different (and distributed) groups can work on compiling different stories.

A scenario embodies the logic of a (complex) ontology processing application including expert sys-
tems and software agents. A scenario concerns the main part of the NS, e.g. the episodes slot, the settings
slot and the characters slot (see Table 3). The NS includes the requirements, purpose and logic of the
ontology in one domain.

A narratological scheme consists of many elements as shown in Table 3. We mention in particular:

– Setting(s): the background of the NS.
– Character(s): actors and stage-props of the NS. Every item that will be involved in the NS should
  be mentioned here.
– Episode(s): the scenario is broken down in a logical sequence. A sequence of episodes acts as the
  ‘plot’ of the NS. If the original material does not have a clear logical sequence or chronological
  order, the NS is considered as a simple document accumulator.

There exists a clear difference between scenarios and episodes. Episodes mostly reveal or express the
internal logic of a scenario, although sometimes the only logically grouping factor is the fact to belong
to the same (application) domain. Scenarios are considered as “raw” episodes or the original material to
distil structured episodes out of. Scenarios can be chapter-alike. Table 5 shows an example of an episode
derived from an e-health sample story source text (shown in Table 4). Table 5 is the episode part (hence
the identifier starting with an E) of the narratological scheme form as shown by Table 3.

2.6. Domain conceptualisation

The domain conceptualisation phase is the most crucial step in the ontology engineering methodology.
During this phase the domain of interest is analysed and finally represented as a DOGMA-style ontol-
ogy. This phase basically corresponds to building the DOGMA ontology base (= a pool of plausible
domain fact types rooted in intuitive natural language meaning descriptions). Several important subac-
tivities result in specific deliverables. There are various (complementary) ways of collecting knowledge
(Section 2.6.1, see Fig. 4). The preference for one of these depends on the available knowledge sources

---

4The NS forms the core of a method reported on earlier as Application Knowledge Engineering Methodology (AKEM) (Zhao
et al., 2004a).
Table 3

Empty narratological scheme

<table>
<thead>
<tr>
<th>Title</th>
<th>&lt;Title of this narratological scheme&gt;</th>
<th>ID</th>
<th>&lt;ID of this document&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theme</td>
<td>&lt;Theme of this narratological scheme&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Author</td>
<td>&lt;The author name of this NS, the contact information of the author can be attached here&gt;</td>
<td>Date</td>
<td>&lt;Creation date of this NS&gt;</td>
</tr>
<tr>
<td>Scope</td>
<td>&lt;The reference to the ontology scoping form (i.e., the resources to build this NS)&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Purpose</td>
<td>&lt;Short description of the purpose&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Settings

S1                     | <Settings of the NS>                       |
S2                     | <Settings of the NS>                       |
...                    | ...                                    |

Characters

C1                     | <Character of the NS>                      |
C2                     | <Character of the NS>                      |
...                    | ...                                    |

Episodes

E1                     | <Episode or scenario of the NS>           |
E2                     | <Episode or scenario of the NS>           |
...                    | ...                                    |

Notes                  | <Extra information of the NS>             |

Table 4

Excerpt of an e-health sample story source text

The patient can select one "primary" GP, which represents the preferred GP of the patient. Further, the patient can select one or more "secondary" GP’s, which can be consulted if the primary GP for example is absent. Also, the patient must assign an "emergency" GP in cases of emergencies. The emergency GP can access the patient record in case of emergencies. When accessing patient records, the GPs authenticate themselves to the HC portal using an electronic ID.

When the GPs perform some clinical tests, such as a blood test, the patient has the freedom to use the laboratory service of LIS that is managed by the same HC portal to process the test samples. He can also choose any of the laboratories external to the system to carry out the prescribed tests. If the patient decides to use the LIS services to execute the laboratory tests, the HC portal generates a token and attaches it to the clinical test prescription along with the required information needed for carrying out the laboratory tests. This information is then sent to the LIS in a pseudonymous way, and the patient can generate a printout of the token or can visually inspect it. Next, the Laboratory doctor affiliated to LIS receives the patient’s prescribed tests to be performed. Note here that the civil identity of the patient remains anonymous to the internal actors and systems of the LIS. After this, the patient provides the required samples needed for testing to the laboratory and the same token represents his identity that maps his clinical test prescription. The Laboratory doctor manages the tests execution (including the samples taken from the patient’s physical presence) using analytical instruments and puts the results report back to the portal in the patient’s personal work space.

After the test results have been saved in the patient’s personal work space, it is the responsibility of the patient to present this information to his GP. If the patient for some reason does not want his primary GP to know about the results, he can choose to present the test result to his secondary GP. The patient has two ways of presenting the test results to the GP, either by taking a printout of the laboratory results for example a kiosk in a (pseudo) public place (e.g., a pharmacy) and present them to the GP himself or by granting permission to the GP to have an access to his personal work space on the portal. After studying the lab results, the GP has the possibility to make changes in the patient’s health profile on the portal.
Table 5

Episode example derived from the e-health sample story of Table 4

<table>
<thead>
<tr>
<th>Episode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1.1</td>
<td>Patient authenticates to HCP</td>
</tr>
<tr>
<td>E1.2</td>
<td>Patient consults GP1, he can choose GP2 if GP1 is not available</td>
</tr>
<tr>
<td>E1.3</td>
<td>Patient performs a lab test</td>
</tr>
<tr>
<td>E1.3.1</td>
<td>HC Portal generates token with pseudonym not linked to civil identity</td>
</tr>
<tr>
<td>E1.3.2</td>
<td>Lab receives non-civil pseudonym from Patient</td>
</tr>
<tr>
<td>E1.3.3</td>
<td>Lab analyses samples</td>
</tr>
<tr>
<td>E1.3.4</td>
<td>Test results uploaded to Portal under token pseudonym</td>
</tr>
<tr>
<td>E1.4</td>
<td>Patient reads and prints test results</td>
</tr>
<tr>
<td>E1.5</td>
<td>Patient consults GP2 (not necessarily = GP1) about test results</td>
</tr>
<tr>
<td>E1.6</td>
<td>Patient gets result printout from a pseudo place and present them to the GP himself or have an access to his personal workspace by granting permission to GP</td>
</tr>
<tr>
<td>E1.7</td>
<td>GP has the possibilities to change patient’s health profile on the portal</td>
</tr>
</tbody>
</table>

Fig. 4. Activities of the domain conceptualisation stage.

and overall context – as described in the ontology scoping form (see Table 2). E.g. if no textual sources are available (for knowledge discovery), consultation of experts (knowledge elicitation) will be an alternative. E.g., if the ontology is issued by an official standard committee, involving many stakeholders in a distributed setting (knowledge negotiation) may be less relevant than working with a few authoritative experts (knowledge breakdown). Depending on whether or not lexons (see below) have been produced, the intermediary step of transforming natural language descriptions into elementary sentences is needed (Section 2.6.2) or not. In turn, these are stepwise converted into a formal domain conceptualisation (Section 2.6.3).

2.6.1. Knowledge collection

Linguistic engineering methods offer a variety of tools that can assist the human modeller in his/her tasks. Under this denomination, we range tools and methods that take a text as input, process it and deliver simple phrase chunks or triples as output. In essence, one can distinguish the following steps in the process of automatically learning ontologies from texts (that are in some way or another common to the majority of methods reported):
1. Collect, select and pre-process an appropriate corpus (done during the previous scoping activity),
2. Discover sets of equivalent words and expressions,
3. Validate the sets (establish concepts) with the help of a domain expert,
4. Discover sets of semantic relations and extend the sets of equivalent words and expressions,
5. Validate the relations and extended concept definitions with the help of a domain expert,
6. Create a formal representation.

Not only terms, roles, concepts and relationships are important, but equally the circumscription (gloss) and formalisation (axioms) of the meaning of a concept or relationship. On the question how to carry out these steps, a multitude of answers can be given. Many methods require a human intervention before the actual process can start (labelling seed terms – supervised learning, compilation/adaptation of a semantic dictionary or grammar rules for the domain, . . .) (Gao & Zhao, 2005; Zhao et al., 2004b). Unsupervised methods do not need this preliminary step (Reinberger et al., 2004). However, the quality of their results is not that high (Spyns & Hogben, 2005) – see Table 6 for output from a privacy domain corpus. A text corpus can preclude the use of some techniques: e.g., machine learning methods require a corpus to be sufficiently large – hence, some use the Internet as additional source. Some methods require a corpus to be pre-processed (e.g., adding POS tags, identifying sentence ends, . . .) or are language dependent (e.g., compound detection). In short, many linguistic engineering tools can be used.

Selecting and grouping terms can be done by means of tools based on distributional analysis, statistics, machine learning techniques, neural networks, and others. An interesting (and recent) alternative are social tagging tools that provide usage frequencies of tags, URLs per tag and tags per URLs. In the same way as natural language processing methods cluster words of a corpus based on their distribution, social bookmarking tags can be “clustered” around the URL tagged. In both cases, it is assumed that the words or tags express somehow a similar meaning. Experience shows that this kind of information provides an interesting bottom-up complement (or maybe even alternative) to the traditional top-down or middle-out (manual) approach of engineering ontologies – see also Mika (2005). All the above mentioned techniques, even if greatly reducing the knowledge acquisition bottle-neck, still require human quality control and post-editing.

Current social tagging systems in general still lack a supporting semantic infrastructure to be used as a direct source of new emergent semantics – see Spyns et al. (2006) for our position in this. These social bookmarking services closely correspond, albeit it in a more global and networked environment, to the

Table 6
Automatically generated lexons for the privacy domain

<table>
<thead>
<tr>
<th>ID</th>
<th>Subject</th>
<th>Predicate</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>working_party</td>
<td>take</td>
<td>decision_by_simple_majority</td>
</tr>
<tr>
<td>2</td>
<td>access_in_particular</td>
<td>where</td>
<td>process_transmission_of_data</td>
</tr>
<tr>
<td>3</td>
<td>accordance</td>
<td>protect</td>
<td>fundamental_right</td>
</tr>
<tr>
<td>4</td>
<td>application_of_directive</td>
<td>take</td>
<td>account_of_specific_characteristic</td>
</tr>
<tr>
<td>5</td>
<td>appropriate_publication</td>
<td>order</td>
<td>block_erasure</td>
</tr>
<tr>
<td>6</td>
<td>association</td>
<td>represent</td>
<td>person_protection_of_right</td>
</tr>
<tr>
<td>7</td>
<td>authority</td>
<td>receive</td>
<td>data_in_framework</td>
</tr>
<tr>
<td>8</td>
<td>authority</td>
<td>seek</td>
<td>view_of_data_subject</td>
</tr>
<tr>
<td>9</td>
<td>authorization</td>
<td>regard</td>
<td>process_such_checking_place_in_course</td>
</tr>
</tbody>
</table>

situation in which a database schema modeller invents table and column names and expects other modellers to understand them as such as natural language words are used. In this case, the shared meaning, however, only resides in the brain of each individual. Well-known issues like ambiguity, cryptic abbreviations, multilinguality etc. inhibit database interoperability and are bound to also manifest themselves in the area of social bookmarking. Instead of language dependent words with implicitly agreed upon meaning, social bookmarking tags should be language neutral concepts that have been explicitly defined to allow its meaning to be shared (Spyns & De Bo, 2004).

To discover semantic relationships between concepts, one can rely on specific linguistic knowledge, already established semantic networks or ontologies, co-occurrence patterns, machine readable dictionaries, association patterns or combinations of all these. In (Karanikas & Theodoulidis, 2002) a concise overview is offered of commercially available tools that are useful for these purposes. To our knowledge no comparative study has been published yet on the efficiency and effectiveness of the various techniques applied to ontology learning, even if overviews on ontology learning methods in general do exist (e.g. Buitelaar et al., 2005; Shamsfard & Barforoush, 2003).

Knowledge elicitation. This includes the well-known way of gathering domain experts or stakeholders who produce on the spot a domain conceptualisation only based on their expertise and the material available at hand. The ontology engineer uses all sorts of direct communication techniques, mainly through interviews and group meetings, to “extract” knowledge from the experts. These activities can be repeated as many times as felt needed by the stakeholders to arrive at a complete and commonly agreed upon set of results. These should be sufficiently adequate to allow verbalisation (see Section 2.6.2). This stage traditionally includes brainstorm meetings, abstraction exercises and sessions to jointly compile a baseline taxonomy (see Fig. 5).

Knowledge negotiation. Recently alternative collaborative techniques have been studied in our lab to create a bakery ontology. As the domain experts have primarily practical skills and no experience whatsoever in modelling, a very down to earth method had to be devised. Moreover, the fact that domain experts belong to their respective organisations resulted in a “community of use” – inspired approach that stresses the importance of the elicitation context and the continuous process of aligning locally evolving interpretations with the common ontology. This elicitation method has been called DOGMA-MESS (DOGMA Meaning Evolution Support System) (de Moor et al., 2006). A specific tool to support DOGMA-MESS has been implemented (Christiaens & de Moor, 2006) using Conceptual Graphs (Sowa, 1999).

Fig. 5. Activities of the knowledge elicitation stage.

6 This research has additionally resulted in a formal framework for context dependency management (De Leenheer et al., 2006).
1984) (CG) as intermediary representation formalism and a Prolog + CG implementation.\(^7\) Typically of DOGMA-MESS is the granular definition of roles assumed by the stakeholders of the modelling exercise. Knowledge engineers perform a high level domain analysis (also taking into account the available NSs of the previous steps) and create an initial set of domain templates (upper part of Fig. 6). These consist of interconnected triples (similar to a binary fact type) that represent a high level view on the domain. The templates use elements from a general upper concept type hierarchy as well as from an upper ontology. A template describes a common knowledge definition most relevant to the common interest of the stakeholders. Domain experts are asked to further specialise these templates (into lower common ontologies) as considered valid for their respective organisations (lower part with up-going arrows in Fig. 6). This is a highly iterative process resulting in many successive versions of templates – represented in Fig. 6 by the dotted line pointing from one version to a next. Cycle after cycle, the conceptualisation is converging towards a commonly agreed upon one by retaining the most relevant triples of the templates. Additions, modifications, and terminations of concepts and relationship and their meaning are the main operations performed by the domain experts (middle part of Fig. 6). To speed up the convergence process, a (still crude) relevancy score is computed. When the process stops, the remaining triples (represented as CGs) can be easily transformed into DOGMA lexons (see below). One could consider this iteratively converging process as a variant of the Delphi method\(^8\) applied to meaning negotiation. The process is mainly asynchronous (hence the need for good supporting tools), while the knowledge elicitation process (Section 2.6.1) is mainly a synchronous one with many of the experts and stakeholders physically sitting together.

Knowledge breakdown. The aim of this activity is to gradually decompose the NSs defined earlier into a hierarchical structure. The structured nature of the NSs should assist in this endeavour. Segmentation and highlighting are the major activities – see Fig. 7.

\(^7\)Maintained by Ulrik Petersen.

Segmentation (Zhao & Meersman, 2005) is used to trim a long sentence into several atomic sentences. The segmentation technique reduces the problem complexity. It can be applied to split a ‘big’ problem into smaller and thus more easily manageable subproblems.

Highlighting (Zhao, 2004) is mainly used as a supporting technique. Although a traditional way of information management, it is still a useful mechanism even if old-fashioned and eating up expert time.

The domain experts should highlight all and only the relevant terms and/or expressions representing key concepts in the text. Three types of phrases are often highlighted: noun phrase, verbal phrase and prepositional phrase. Highlighting should be done consistently and in compliance with the vision statement (cf. Section 2.2) and user requirements and purposes as identified in previous steps.

During this step (see Fig. 7), the knowledge engineer can rephrase certain wordings or formulations to achieve a higher order of generalisation or abstraction. These abstractions will of course be the prime candidates to be highlighted (otherwise it is not really worthwhile to do an abstraction effort). For traceability reasons, it is advised to note down the abstractions made (and for which reasons). According to the expertise level of the knowledge engineer, it is possible that (s)he segments, abstracts and highlights in the same pass (see e.g. Table 7). For the sake of clarity, we present highlighting and segmentation separately. Part of speech taggers combined with word frequency analysis (e.g., the TF/IDF measure) could be potentially of help when the volume of text to highlight is becoming large.

2.6.2. Verbalising elementary sentences

The creation of basic elementary sentences or fact types is linguistic in nature. Based on the outcomes of the previous activities, simple sentences representing elementary plausible fact types must be produced. It is possible that the domain experts and knowledge engineers have already (partially)

---

Segmentation and highlighting of a source sentence from the EU privacy directive

Appropriate technical and organisational measures must be implemented to protect personal data against...

...article 17 of the Directive 95/46 EC

1. Appropriate technical measures must be implemented.
2. Appropriate organisational measures must be implemented.
3. Appropriate technical measures are to protect personal data.
4. Appropriate organisational measures are to protect personal data.

[In earlier publications, most of the following steps have been dubbed DOGMA Ontology Modelling Method (DOM²).]
produced sufficiently fine-grained results (ORM-style binary fact types or RDF-like triples) so that it is no longer necessary to explicitly perform this activity. On the other hand, depending on the natural language processing techniques and tools used, it is possible that the sentences or triples produced automatically should be further split. The main characteristic of an “elementary fact type” is that it cannot be split into smaller constituent parts. These sentences should correspond, as far as possible, to a simple subject-verb-object structure.

2.6.3. Lexon engineering

In the following steps, the informal conceptualisation, under the form of verbalised fact types in natural language, is progressively transformed into language independent formal statements rooted in informal meaning descriptions, according the STAR Lab DOGMA ontology engineering formal framework. Figure 8 shows the steps involved. Even if one wants to implement an ontology in directly in RDF(S) or OWL-DL, these steps are useful as they impose a methodological rigour (resulting in e.g. the inclusion of meaning descriptions in the comment field of a resource identified by a URI – in many cases now lacking).

Create lexons. This activity transforms the verbalised fact types into lexons. The results are formally described as \(<(\lambda, \zeta); \text{term}_1 \ \text{role}_1 \ \text{co-role}_2 \ \text{term}_2 >\), where \(\zeta\) denotes the context, used to group lexons that are intuitively related to each other in the conceptualisation of the domain (Meersman, 2001b). Currently, the context \(\zeta\) refers to the particular section or input document from which the lexon has been extracted. Informally we say that a lexon is a fact type that may hold for some domain, expressing that within the context \(\zeta\) and for the natural language \(\lambda\) the \(\text{term}_1\) may plausibly have \(\text{term}_2\) occur in role with it (and inversely \(\text{term}_2\) maintains a co-role relation with \(\text{term}_1\)) (Spyns, 2005b). Lexons operate on the linguistic level. They are independent of specific applications and should cover relatively broad domains (Spyns et al., 2002). They form a lexon base, which is constituted by lexons grouped by context and language (Spyns, 2005a). Table 8 illustrates how the verbalised plausible fact types from sentences (e.g., S4.1) are turned into lexons.10

Refine lexons. The lexons newly created should undergo a quality check. We say that a lexon is a ‘good’ one when it

- is highly re-usable,
- is as simple as possible,
- represents the correct information,
- is binary.

---

10We omit the language identifier \(\lambda\) as the examples in this paper only concern English.
Table 8

Table 8 shows lexons resulting from the refinement step. Text describing the setting (cf. Table 3) has been segmented. Segment S4.1 has been transformed into a lexon, which has resulted in two lexons after refinement. We are aware that in the current state, it is not so simple to define precise and verifiable metrics. This is an area where (DOGMA) ontology modelling too a large extent still is an art. Practical experience has to lead to quality requirements: e.g. avoid building lexons without a co-role. In particular automated procedures suffer from this. Manual intervention and completion is needed. Current work on ontology evaluation could be a good starting point for defining objective measures. As the lexons can be considered as paths in a graph (or forest), the metrics proposed in (Gangemi et al., 2005) to measure the structural dimension of an ontology are excellent candidates to start with.

Note that the creation of elementary sentences (see Section 2.6.2) should almost automatically lead to good lexons. Nevertheless, it could happen that some elementary sentences are to be represented by more than one lexon and vice versa. It mostly concerns knowledge implied by an elementary sentence.

Ground lexons. Natural language terms are associated, via the language and context combination, to a unique word sense represented by a concept label (e.g. the WordNet identifier person#2). With each word sense, a gloss or explanatory information is associated that describes that particular notion. To cope with synonymy, homonymy and translation equivalents, we take a \( m:n \) relationship between natural language terms and word senses into account. Lexon grounding is a conceptual exercise that links the terms and roles that constitute a lexon to existing entries of dictionaries, lexica or standards, which contain commonly accepted meaning. WordNet (Fellbaum, 1998; Miller, 1995) is commonly used to this end as this resource contains many senses and associated unique definitions. An equivalent for the medical domain is the Unified Medical Language System (UMLS) (Humphreys & Lindberg, 1992).

If no adequate definitions exist,11 new definitions should be drafted by hand following terminological principles by the domain experts and other stakeholders. Table 9 shows some newly made explanations with their term – e.g. the term PersonalData (see Table 8) has now received an explanation. In this way the logical vocabulary of the ontology (in the form of terms and roles) receives semantics. As a result, synonyms are easily detectable (i.e. they point to the same definition). New labels have to be chosen for a set of synonyms or expressions having the same meaning. These labels are preferably (slightly) different from natural language words to indicate that they operate on the conceptual level rather than the language level (Spyns & De Bo, 2004). In the DOGMA ontology engineering framework, the definitions, the

---

11WordNet mostly contains general non technical terms.
labels and the synonyms are entered in the DOGMA concept definition server. This is indeed a laborious exercise, but the only way to achieve common understanding and explicitly defined semantics. Hence the importance of re-using existing lexicographic or terminological resources as they provide descriptions of meaning commonly agreed upon by large communities of users. Tools to exploit these resources and seamlessly enrich them (e.g., Benassi et al. (2004) will become more and more valuable).

Create meta-lexons. Meta-lexons are language-neutral and context-independent (conceptual level). A term or role (or sometimes an expression) used in a specific context and language in principle points to a non ambiguous meaning (or sense). The result of the preceding lexon grounding activity (descriptions for unique senses linked to terms) most naturally provides the basis to create meta-lexons. The English <DataController, collect, beCollected, PersonalData> lexon in the context identified by “setting 4.1 NSID:1” (see Table 8) is transformed using WordNet and the grounded terms of Table 9 into the following meta-lexon: <datacontroller#1, collect#4, personal#3datum#1>.

It can happen that equivalent or synonymous lexons or triples have been produced in the course of the domain conceptualisation and lexon engineering steps. As it is senseless to keep such lexons (also possibly across language borders), meta-lexons are created. Meta-lexons are an abstraction of equivalent lexons and synonymous lexons (see the example below). Two lexons or triples are equivalent when the respective terms and roles point to the same concepts. This also holds when the sequence of terms and roles is inversed. In case of an inversed sequence, words can be antonyms. Let us look at three naive examples (supposedly used in the same context and language):

- <bike, follow, be_followed_by, car> & <bicycle, go_after, be_followed_by, automobile>. This example illustrates that two lexons are synonymous when their composing parts have the same sense. Here, bike is bicycle, follow is to go after, and a car is an automobile.
- <dog, eat, be_eaten, meat> & <meat, be_eaten, eat, dog>. This sample illustrates that two lexons are equal when their terms and roles are the same (possibly in inverse sequence).
- <bike, follow, be_followed, car> & <automobile, precede, is_preceded_by, bicycle>. This example (equivalence) is the combination of two previous examples. The terms are synonyms and inversed while the role names are antonyms.12

12Also generalisation and specialisation relationships can hold between lexons – DOGMA-MESS iteratively refines what could be called template lexons. This resembles ontology patterns (cf. Gangemi, 2005), but has not yet been explored in the DOGMA context.

<table>
<thead>
<tr>
<th>ID</th>
<th>Label</th>
<th>Term</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>datacontroller#1</td>
<td>DataController</td>
<td>Someone who maintains and audits privacy data</td>
</tr>
<tr>
<td>2.</td>
<td>measure#1</td>
<td>OrganisationalMeasure</td>
<td>Any manoeuvre that fits the organisational strategy made as part of progress towards a goal</td>
</tr>
<tr>
<td>3.</td>
<td>personal#3datum#1</td>
<td>PersonalData</td>
<td>That data relating to a living individual which if in the possession of a data controller could by itself or with other data already in the possession of the data controller easily identify the living individual</td>
</tr>
</tbody>
</table>
Fig. 9. Activities of the lexon engineering stage.

2.7. Application specification

This stage of the modelling process corresponds to creating the first-order instantiation of the interpretation of (parts of) the ontology base by an application (Spyns et al., 2002). Stated otherwise, an application maps (or commits) its local vocabulary to a domain conceptualisation and meaning commonly agreed upon as represented by a selection of meta-lexons in the ontology base. The application specification activities are shown in Fig. 9. They include structuring the application domain (Section 2.7.1), adding application specific semantic restrictions on the domain conceptualisation (Section 2.7.3) and preparing for (Section 2.7.2) and performing a validation step (Section 2.7.4). Quite some novel ontology validation and task-based evaluation methods have been proposed in the recent literature (e.g. Porzel & Malaka, 2005) that can be interesting for the DOGMA methodology as well.

2.7.1. Structuring

The knowledge structuring activity is applied on every section of the NS in natural language. Based on several NSs, a knowledge constituent analysis is done. This takes on the form of structured natural language (due to the hierarchical nature of the NS documents) where details are provided for every knowledge constituent. Logical connections inside the episodes, more precisely on the meta-lexons modelled on basis of the segmented episodes, are to be expressed in terms of the more traditional semantic constraints. A pseudo high level language is used to support the structuring mechanism (Zhao, 2004). These have no formal definition but can help to prepare the material at hand to be more easily transformed into specific (business) rule languages (Tang & Meersman, 2007b), expert systems formalisms or plain programming language constructs.

2.7.2. Definition of competency questions

A set of application specific competency questions (Uschold & Gruninger, 1996) is defined. Such a set of competency questions functions as a base line reference of domain knowledge that an ontology is expected to contain from the perspective of an application. Competency questions should be structured
in a hierarchical fashion, proceeding from general to more specific. Complex questions should be built from simpler ones. It is recommended that domain experts, ontology engineers as well as end users be involved in this activity.

2.7.3. Definition of semantic constraints

Additional restrictions on the meta-lexons representing a specific application conceptualisation are situated in the commitment layer (see also Section 1.2). This layer, with its formal constraints, is meant for interoperability issues between information systems, software agents and web services. These kinds of constraints are mathematically founded and concern rather typical DB schema constraints (cardinality, optionality, etc.) as defined in the Object Role Modelling methodology (Halpin, 2001). They also correspond largely to OWL restrictions (Bach et al., 2007). As the ORM methodology has been designed to model database schemas, it cannot be ported as such to the field of ontology engineering (Spyns, 2005a, b). Nevertheless, we have borrowed the following semantic constraints from ORM.

1. Internal uniqueness constraints: this indicates that in a particular instance of a binary fact type, there can be only one instance of the concept involved in the relationship. For instance, if we consider the meta-lexon \(<x, \text{has}, y>\), then an internal uniqueness constraint on the role has, is verbalised as each \(x\) has at most one \(y\).

2. External uniqueness constraints: this specifies that in each instantiation of a combination (or “join”) of two or more meta-lexons, there can only be one instance of the concept with the constrained roles involved. Consider the meta-lexons \(<x, \text{has}, y>\) and \(<x, \text{has}, z>\). If there is an external uniqueness constraint imposed on the role has in both meta-lexons, this may be verbalised as each \(y\) and \(z\) combination has at most one \(x\).

3. Simple mandatory constraints: a simple mandatory constraint indicates that every instance of a concept must necessarily participate in the relationship on which the constraint is defined. Taking the meta-lexon \(<x, \text{has}, y>\) a simple mandatory constraint defined on the relationship has is verbalised as every \(x\) has at least one \(y\).

4. Disjunctive mandatory constraints: a disjunctive mandatory constraint defined on a concept occurring in one or more meta-lexons implies that the concept must necessarily be involved in at least one (but possibly more) of the relationships specified. Consider the meta-lexons \(<x, \text{has}, y>\) and \(<x, \text{has}, z>\) and suppose that a disjunctive mandatory constraint has been defined on \(x\). Verbalising this yields \(x\) has either \(y\) or \(z\) (or both).

5. Subsets: this constraint specifies that the existence of one meta-lexon implies the existence of another. That is, if an instance of the “implying” meta-lexon exists, then an instance of the “implied” meta-lexon also necessarily exists. For instance, if the meta-lexon \(<x, \text{has}, y>\) has a subset constraint on the meta-lexon \(<x, \text{has}, z>\), this may be verbalised as if \(x\) has \(y\), then \(x\) has \(z\).

6. Equality: the equality constraint indicates that two instance populations are equal.

7. Exclusion: this specifies that if a concept occurs in more than one meta-lexon, it cannot be involved in more than one of the roles. Suppose that the meta-lexons \(<x, \text{has}, y>\) and \(<x, \text{has}, z>\) have an exclusion constraint defined on the role has. This is verbalised as no \(x\) has both \(y\) and \(z\).

8. Subtypes: subtyping specifies that one concept type is a subtype of another. In many cases, subtyping is combined with disjunctive mandatoriness (defining partitions).

9. Occurrence frequency: an occurrence frequency constraint is a generalised version of an internal uniqueness constraint, namely when \(n > 1\). That is, an occurrence frequency constraint specifies
Fig. 10. ORM graphical representation of commitments for a DOGMA-style ontology.

that \( n \) instances of a particular concept must be involved in a particular relationship. It is possible to specify the cardinality as exactly \( n \), greater than or equal to \( n \), or greater than \( n \).

10. Final checks imply checking for consistency in the commitments that have been specified. This includes ensuring that different constraints do not contradict one another. Some patterns have been examined in (Jarrar & Heymans, 2006).

An alternative commitment language, called \( \Omega \)-RIDL (Verheyden et al., 2005), is being designed. It is inspired by the RIDL language developed earlier (Meersman, 1982). The current version, supporting a subset of the ORM constraints, is being extended (Trog et al., 2007) to support a better conversion to and from OWL-DL (Bach et al., 2007).

An example of the result of adding semantic constraints is illustrated by Fig. 10. It displays in the ORM graphical notation some of the commitments as captured from the source text of Table 4 and episode represented by Table 5. A HC provider can manage several different HPs (health profile) while one HP can be managed by several different HC providers (the combination of an HP and HC provider must be unique – indicated by a line with arrows above the two boxes). A HCP can access several different HPs while one HP can grant access to several different HCPs (the combination of an HP and HCP must be unique – indicated by a line with arrows above the two boxes). General practitioners (GP) can grant access to HPs (\( m : n \) relationship, no specific graphical indication). HCP (health care portal) accesses at least one HP and at least one Personal Workspace (two simple mandatory constraints indicated by a black dot in Fig. 10). A health status is applied to at most one HP (internal uniqueness constraint on one role indicated by the line with arrows above “applied to”).

2.7.4. Validation by means of competency questions

The application-specific requirements are met if the competency questions defined earlier can be adequately answered. An advantage of ORM is that constraints can be quite easily and naturally verbalised. Consequently, a check should be done to determine whether all competency questions may be answered through some verbalisation based on the commitments defined. If this is not possible, it may be necessary to update and redefine the commitment rules of the previous step.
2.8. Final remarks

The methodology described above offers a general framework for ontology engineering encompassing different ways of tackling the problem. Therefore, several steps and stages have been identified, each with its own preferred technique or method all having pros and cons. It is up to the ontology engineer to select the most appropriate track and related methods depending on his/her particular assignment. Nevertheless, after the domain conceptualisation step (cf. Section 2.6), the various “tracks” come together and the intermediary data and material should result ideally in an equivalent (not identical) ontology, no matter which way has been preferred and which specific ontology language is chosen for an actual implementation.

Lexons generated by machine learning most probably will differ from the ones obtained from brainstorming experts, not only in the terms or labels used but also in model itself. With respect to the former issue, the subsequent lexon engineering phase constitutes an abstraction and selection exercise resulting in meta-lexons and grounded concepts. What counts is the explicit description of the agreed upon meaning, and not so much the labels and terms representing the concepts. Regarding the latter issue, we refer to database schema transformations that solve a similar problem – even if this avenue of research is still largely unexplored for ontologies.

What remains (and is not discussed in this paper) is the concrete implementation of the conceptualisation into an ontology (in the sense of Guarino & Giaretta, 1995). In our case, the lexons and meta-lexons are stored in the OntologyServer while the concepts and relationships are entered in the ConceptDefinitionServer.

3. Related work and discussion

A general survey on ontology modelling methodologies is provided by Gómez-Pérez et al. (2004, pp. 113–153), including short descriptions of the methods. A very recent overview is given by Simperl & Tempich (2006), who do not provide many details of the various methods but rather succinctly situate them. As such it is an excellent good starting point for further bibliographic research.

An older, but still relevant, overview can be found in (Jones et al., 1998). Some novel methodology developments in ontology engineering that concern collaborative and community aspects (Holsapple & Joshi, 2002; Karapiperis & Apostolou, 2006; Kotis & Vouros, 2005) are shortly discussed in (Tempich, 2006, pp. 212–215). This new trend should not be a surprise seen the overall rising interest in combinations of the Web2.0 with the Semantic Web – also for ontology modelling – e.g., see Braun et al. (2007). An overview of overviews can be found on the following OntoWorld wiki page: http://ontoworld.org/wiki/Ontology_Engineering. Some of the surveys mentioned there (e.g., Fernández-López et al., 2002) have too a large extent become obsolete with the publication of Gómez-Pérez et al. (2004).

Almost all the overviews contain to a very large extent descriptions of some “usual suspects” (Methodontology (Fernández et al., 1997), OnToKnowledge (Sure, 2003), TOVE (Grueninger & Fox, 1995), Enterprise Ontology (Uschold & King, 1995), as well the “unified methodology” (Uschold & Grueninger, 1996) of the latter two). In addition, the development of the influential knowledge engineering methodology CommonKADS (Schreiber et al., 1999) has been an inspiration for other ontology engineering methods. The DOGMA modelling methodology partly draws on some best practices from some of these methods:
– TOVE: the inclusion of competency questions as a way to scope the domain of interest and to evaluate its conceptualization.

– Enterprise Ontology: the notions of brainstorming, the middle-out approach, and the grouping of terms. The thorough documentation of the entire development process is also included.

– the “unified methodology”: the specification of target users as a way to scope the domain of interest. Also, the generalization of competency questions that are not rigorously formal is included in the suggested methodology. A number of additional suggestions with regard to brainstorming have been taken into account.

– Methontology: the inclusion of management activities.

– OnToKnowledge: the inclusion of a feasibility study before any ontology development commences.

– CommonKADS: the focus on specific documented deliverables for every activity in the ontology development process.

Nevertheless, we state that general, comprehensive, didactic and scientifically based methodology ‘cookbooks’ (e.g. Benjamin et al., 1994; Mizoguchi, 2003) that cover how to actually create from scratch and deploy a multilingual ontology hardly exist. In (Garcia Castro et al., 2006), a similar consideration led to designing a novel modelling methodology for bio-medical ontologies. There are the ONIONS and ONIONS-II methodologies for ontology creation and merging that have been successfully applied to several domains (bio-medical, legal, fishery) (Gangemi et al., 2002). Even though they contain guidelines on how to engineer an ontology, they seem to be not yet completely engineered. Some didactic papers are tuned (or biased?) towards specific ontology engineering tools (e.g. Noy & McGuinness, 2001). The well documented OnToKnowledge methodology (Sure, 2003) is a centralised ontology development method that risks becoming too much geared toward a single application. This could also be a potential problem for the UPON methodology that claims as its distinctive feature to be primarily use case driven (Missikoff & Navigli, 2005). In our view, an ontology should cover a domain rather than a single application. A recent methodology that puts the focus on collaborative and distributive aspects is Diligent (Tempich, 2006). It applies practices of “meeting techniques” (argumentation theory) to steer the process of collaboratively modelling an ontology supported by internet tools (e.g., a wiki environment) (Sure et al., 2004). A variant of Diligent has been adapted to the legal domain (Casanovas et al., 2007). The DOGMA modelling methodology, and in particular its community-based extension DOGMA-MESS, also explicitly addresses distributed meaning negotiation and consensus building, but without neglecting a centralised approach either. Unlike the Methontology and UPON methods, which are primarily building on software engineering project management methods, the DOGMA project aims at refining and applying an integrated methodology based on solid DB-experience, insights from linguistics and collaborative aspects from social sciences. Just like UPON builds on the Unified Software Development Process, UML and related tools, DOGMA largely builds on the ORM methodology and related technologies. Also others have used UML for ontology construction – e.g. Kogout et al. (2002). However, we consider that using UML as such, without a methodology, is not that particularly helpful for making ontology modelling a science rather than an art. As far as we know, DOGMA is still unique in its methodological double articulation (Spyns et al., 2002).

The main endeavour for DOGMA is to continue to grow the methodology, apply it to many more new fields and modelling situations and maintain its consistency and internal structure. Other topics consist of coping with heterogeneity (ontology merging and alignment), managing changes (ontology evolution) and a double articulation.

\[^{13}\text{An experiment using the UML use case and activity diagrams within the DOGMA framework has been tried out as well (Tang et al., 2006).}\]
and supporting quality assurance (ontology evaluation). These items are needed for a complete ontology
life-cycle engineering methodology. They are being researched in our lab, but the corresponding methods
are not yet integrated in the overall methodology.

A particular challenge is to link up ontologies graphically represented by ORM with popular ontol-
ogy implementation languages, in particular RDFS and OWL-DL. Evidently ORM, being a conceptual
modelling language, does not compete with RDFS and OWL as these latter are W3C standards ontology
implementation languages. In fact, a graphical conceptual model representation language should be part
of any good general ontology modelling methodology. Converters are to be defined that produce RDF(S)
or OWL statements when processing the conceptual model (in ORM). In analogy with the way UML
models are eventually transformed into in Java or C++ software modules. Note that ORM is already long
time in use as a DB schema modelling language.\textsuperscript{14}

An RDFS or OWL-Lite implementation of an ontology (without instances) basically corresponds to
(part of) a DOGMA ontology base. Ontology restrictions encoded with OWL-DL statements correspond
to the constraints of the DOGMA commitment layer. Converting DOGMA-style ontologies into OWL-
DL (and vice versa) has been taken up recently at the lab. Intermediary results show that DOGMA-style
ontologies can be too a large extent transformed into RDF and OWL implementations (Bach et al., 2007).

4. Conclusions

In this paper we have described an ontology engineering methodology that integrates best practices
of many worlds (database modelling, social sciences, linguistics, knowledge representation, ...). This
has been the result of various modelling experiences and exercises in multiple application domains and
different settings. We have presented parts of a practical and didactic methodology in the hope that other
researchers take it up and try it out on their own examples and material. Inspired by the CommonKADS
book (Schreiber et al., 1999), and to be practical, we have introduced for each important modelling step
the expected output (up to the point of designing typical forms). This methodological rigour allowed us
to introduce various paths in the engineering process, characterised by the use of particular techniques
or material, all eventually contributing to the same goal.

Finally, we want to point out that other aspects of ontology engineering research at our lab have not
been mentioned in this paper due to a lack of space, such as ontology evolution (De Leenheer & Mens,
2008), the study of context (De Leenheer et al., 2006), ontology visualisation (Pretorius, 2005), ontol-
ogy mining (Reinberger & Spyns, 2005; Reinberger et al., 2004) and automatic evaluation (Spyns
& Reinberger, 2005; Spyns & Hogben, 2005), mediation (Deray & Verheyden, 2003), emergent se-
manitics (Cudré-Mauroux & Aberer, 2006), scalability (Jarrar & Meersman, 2002; Zhao & Meersman,
2005), ontology-based business intelligence (O’Riaín & Spyns, 2006) and business rules modelling (De-
ney et al., 2002; Tang et al., 2007; Tang & Meersman, 2007a). For many of these research topics, tools
are being implemented (Christiaens & de Moor, 2006; Oberle & Spyns, 2004; Pretorius, 2005; Spyns
et al., 2006; Tang & Meersman, 2007b; Trog et al., 2006). Most of the tools are (or will be) integrated
with DogmaStudio,\textsuperscript{15} the lab’s central ontology workbench. So, next to having presented our ontology
engineering methodology, we have also provided a bird’s eye view on past work at VUB STAR Lab at
the occasion of its 10th anniversary.

\textsuperscript{14}ORM models are being mapped into SQL DDL statements – see also Halpin (2001), resp., Demey et al. (2002), on the
advantages of using ORM for database modelling resp., ontology modelling.

\textsuperscript{15}http://www.starlab.vub.ac.be/website/dogmastudio.
Acknowledgements

We like to thank in particular Dr. Aldo de Moor and Dr. Gang Zhao for their inspiring ideas as well as our other past and present colleagues at VUB STAR Lab who have all contributed their part to the research described in this paper as well as different organisations for funding projects. We gratefully acknowledge the support of the IST (FF POIROT, PRIME, KnowledgeWeb, PROLIX) and Leonardo da Vinci programmes (Co-Drive) of the European Union, the IWT-Vlaanderen (OntoBasis, SCOP, PoCehrMOM), the Vrije Universiteit Brussel (OMTFI, DOGMA), and the FWO-Vlaanderen (BonaTema).

References


P. Spyns et al. / An ontology engineering methodology for DOGMA


