Journal of Data Semantics manuscript No.
(will be inserted by the editor)

Christophe Debruyne, Trung-Kien Tran, and Robert Meersman. Grounding Ontologies with Social Processes and Natural Language. Journal on Data Semantics, 2(2-3):89-118, 2013

Grounding Ontologies with Social Processes and Natural Language

Christophe Debruyne · Trung-Kien Tran · Robert Meersman

Received: date / Accepted: date

Abstract Ontologies for enabling semantic interoperability is one of the branches in which agreement between a heterogeneous group of stakeholders is of vital importance. As agreements are the result of interactions, appropriate methods should take into account the natural language used by the community during those interactions. In this article, we first extend a fact-oriented formalism for the construction of so-called hybrid ontologies. In hybrid ontologies, concepts are described both formally and informally and the agreements are being grounded in community interactions. We furthermore present GOSPL, a collaborative ontology engineering method on top of this extension and describe how agreements on formal and informal descriptions are complementary and interplay. We show how the informal descriptions can drive the ontology construction process and how commitments from the ontology to the application are exploited to steer the agreement processes. All of the ideas presented in this article have been implemented in a tool and used in an experiment involving 40+ users, of which a discussion is presented.

Keywords Collaborative Ontology Engineering · Hybrid Ontologies · Semantics

This study is supported by the INNOViris Open Semantic Cloud for Brussels (OSCB) project, financed by the Brussels Capital Region.

Debruyne, Tran and Meersman
Semantics Technology and Applications Research Lab
Vrije Universiteit Brussel
Pleinlaan 2, B-1050 Brussels, Belgium
Tel.: +32-2-6293308
Fax: +32-2-6293525
E-mail: {chrdebru,truntran,meersman}@vub.ac.be

1 Introduction

The formal semantics of a (computer-based) system quite simply is the correspondence between this system and some real world as perceived by humans. It is usually given by a formal mapping of the symbols in the system's description to objects in that real world, such that relationships and logical statements in the specification language can be assigned a truth-value depending on whether a certain state of affairs among objects exists in the real world. As the real world is not accessible inside a computer, storing and reasoning about semantics requires the world to be replaced by an agreed conceptualization. This conceptualization is often in the shape of a formal (mathematical) construct. A computer-based, shared, agreed formal conceptualization is what is known as an *ontology*. Ontologies constitute the key resources for realizing a Semantic Web. Ontologies also help tackling the difficulty of interoperating *autonomously* developed and maintained information systems in a meaningful way.

As a consequence, ontologies in general will evolve while such agreements are developed and put in place. These ontologies are approximations of a real world; in fact to the Web services involved, ontologies *are* the world. Ontologies represent an *externalization* [12] of the semantics outside of the information system. The basic techniques and architecture for semantic interoperation is based on annotation (of an application system) and reasoning (about the concepts involved, in terms of the ontology).

However, the problem is not what ontologies are, but how they become community-grounded resources of semantics, and at the same time how they are made operationally relevant and sustainable over longer periods of time. In the DOGMA framework [37], fact-oriented approaches such as NIAM/ORM [72,29] have been proven useful for engineering ontologies. A key characteristic here is that the analysis of information is based on natural language fact types. A fact type is the generalization of facts, a collection of objects linked by a predicate. "[Person] knows [Person]" would be an example of a fact type, and "[Christophe] knows [Robert]" would be a fact in this example. This brings the advantage that "layman" domain experts are facilitated in building, interpreting, and understanding attribute-free[1], hence semantically stable ontologies, using their own terminology. The semantics in ontologies are the result from agreements within a community to use particular labels for referring to certain concepts.

Ontology construction must be viewed as a complex, social and distinct methodological activity. It must lead to formalized semantic agreement involving its stakeholder communities and the various social processes within those communities. An important tool in reaching those agreements is the use of *glosses*, natural language descriptions interpretable by humans. The use of glosses while reasoning and discussing concepts among humans aid in the disambiguation of different concepts, discovery of implicit relations between concepts, discovery of gaps in the ontology, etc. We call the process of describing with a natural language description *articulation*. Enabling semantic interoperability should therefore explicitly involve the hybrid aspects of information; i.e. the co-existence of formal reasoning and "informal" human interactions (with natural language).

In previous work [18], we presented a formalism for constructing so called hybrid ontologies [51]. In hybrid ontologies, communities are promoted to first-class citizen, part and parcel of the formalism, such that the interactions within the evolving community result in series of ontology evolution operators. The natural language aspect is vital, as the closer the link between human communication and the resulting system and/or business communication, the more likely such systems will work as intended by their various stakeholders. In [19], we presented a method built on top of that framework – called GOSPL – as well as a tool for this method. GOSPL is a teachable and repeatable collaborative ontology evolution method supporting stakeholders in interpreting and modeling their common ontologies in their own terminology and context, and feeding back these results to the owning community.

This article starts in Section 2 with related work on collaborative ontology engineering and tool support, in which we identify a gap between methods (and their tool support) that take into account a special linguistic resource and social processes leading to agreements. Our approach is described over several sections. Section 3 describes the hybrid ontology-engineering framework – also describing the properties of agreements on formal and informal concept descriptions – and the collaborative method on top of this framework is presented in Section 4. Section 5 describes how the evolution of glosses leads to new fact types and constraints for the formal part of the hybrid ontology. Section 6 then focuses on the exploitation of application commitments, which are descriptions of how one individual application commits to an ontology, are used in community interactions. Application commitments are used to find counterexamples to claims made by community members and thus guide the discussions. Section 7 presents the tool supporting the ideas presented in Sections 3 through 6 next to the results of a usability study presented elsewhere. We conclude the paper with a discussion in Section 8 and conclusions in Section 9.

The main contributions of this paper are the state-of-the-art in Section 2, the translation of ontologies into Description Logic in Section 3 and the exploitation of glosses and application commitments in Sections 5 and 6 respectively.

2 Related Work

Ontology engineering is defined as the set of activities that concern the ontology development process, the ontology life cycle, the principles, methods and methodologies for building ontologies, and the tool suites and languages that support them [25]. One can generally identify two phases in an ontology engineering method: elicitation and application [14]. In elicitation, knowledge is extracted from various resources such as documents of any kind or the experience of domain experts within a specific context. In the subsequent application phase, an ontology is used in an application context.

Quite a few surveys on the state of the art on ontology engineering methods exist [62,61,25]. For this paper, the methods we take into account are: CYC [27, 44], Business Semantics Management (BSM) [11], DILIGENT [59,58], DOGMA [37] and DOGMA-MESS [52], HCOME [41,40], Holsapple and Joshi [33], Karapiperis and Apostolou [38], METHONTOLOGY [22,3], NeOn [24,9], On-To-Knowledge [63,64], OntoEng [1], Ontology 101 [56], the Unified Method [69,68] and UPON [15,16]. We furthermore took both Web-Protégé [67] and Collaborative-Protégé [66] into account. Both are collaborative tools for ontology engineering developed by the same group of Ontology 101, but do not refer to a specific method.

[1] There are only fact types, no distinction between relations and attributes. The constraints on roles in these fact types determine the "attributeness" of a fact type.

Table 1 Comparison of the state-of-the-art on ontology engineering methods.

	CYC	Business Semantics Management	DILIGENT	DOGMA(-MESS)	HCOME	Holsapple et al. 2002	Karapiperis et al. 2006	METHONTOLOGY	NeOn	On-To-Knowledge	OntoEng	Ontology 101 (pre-collab)	Ontology 101 (collab)	Unified Method	UPON
Explicitly intended for distributed & collaborative construction?	N	Y	Y	Y	Y	C	C	N	Y	C	C	N	Y	C	C
Natural language descriptions of concepts?	D	Y	N	Y	A	N	N	Y	Y	D	Y	D	Y	Y	Y
Special linguistic resource as software artifact?	N	Y	N	Y	W	N	N	N	N	N	N*	N	N	N	N*
Tool support?	Y	Y	A	Y	Y	N	O	Y	Y	Y	O	Y	Y	N	O
Tool support for dialogue?	N	Y	Y	N	Y	N	N	N	E	E	N	N	Y	N	N
Social processes for agreements on formal descriptions?	N	I	I	Y	I	Y	Y	N	I	I	Y	N	Y	Y	Y
Tool support for social processes on formal descriptions?	N	I	I	P	I	-	-	N	I	I	-	N	Y	-	-
Social processes for agreements on informal descriptions?	N	I	N	N	I	N	N	N	I	N	Y	N	P	N	N
Tool support for social processes on informal descriptions?	N	I	N	N	I	-	-	N	I	N	-	N	I	-	-
Agreement leads to ontology evolution?	-	N	M	M	M	M	M	-	M	M	M	-	Y	M	M
"Owner" of the ontology?	-	C	H	H	H	K	K	-	K	K	N	-	K	K	K

Table 1 compares the different methods with respect to following aspects:

- **Explicitly intended for distributed and collaborative construction?** The values are: Y) yes; N) no; and C) collaborative aspects are touched upon, but not explicitly mentioned.
- **Natural language descriptions of concepts?** Y) yes; N) no; D) support for documentation in which natural language definitions can be provided; and A) adoption of existing resources to align with concepts of the ontologies.
- **Special linguistic resource as software artifact?** This aspect investigates whether there is a special linguistic resource next to the formal descriptions of the ontology. The values are: Y) yes; N) no; and W) use of an existing linguistic resource. Both OntoEng and UPON do explicitly refer to a glossary in their papers, but it is not stored as a software artifact. Their entries are therefore marked with an asterisk.

Ontologies should be considered as evolving entities. Argumentation and negotiation processes to discuss the evolution of ontologies are thus critical. A negotiation process is defined as a specification conversation about a concept (e.g., a process model) between selected domain experts from the stakeholders (community of organizations) [13]. In order to substantiate their perspectives, domain experts must formulate arguments. The following aspects look to what extent the methods have support for specific social processes and whether their tools provide support for those.

- **Tool support?** Y) there is specific tool support for this method; N) no specific tool support for this method has been proposed; A) adopting/extending existing ontology engineering tools for the method; and O) authors refers to other existing (type of) tools for some tasks.
- **Tool support for dialogue?** Y) Yes; N) no or not proposed; and E) via external (integrated) service or tool. The most accepted argumentation model is Issue-Based Information System (IBIS) [43], which provides a simple and abstract infrastructure for non-trivial problems that cost a lot to solve in terms of time, money, etc. IBIS was the model that DILIGENT, HCOME, and NeOn adopted. DILIGENT processes are still under control of knowledge engineers (the completion of some activities is partially dependent on the decisions of a board of experts), whereas HCOME aims at empowering users to be

completely autonomous in their actions and decisions. The first is also true for NeOn. The NeOn toolkit provides support for discussing issues (based on DILIGENT) via a plugin that is connected with Cicero [21], a platform for keeping track of discussions between the developers and users of an ontology. On-To-Knowledge proposed a plugin that is based on existing commercial software for the brainstorming and elicitation of competency questions. Competency questions state which queries the ontology should support (cfr. [70]). Both NeOn and On-To-Knowledge are therefore regarded as using external (integrated) services and tools.

- **Social processes for agreements on formal descriptions?** This aspect investigates to what extent methods explicitly describe or prescribe special social processes for agreeing on formal descriptions of concepts. Informal here means that concepts are described by means of natural language descriptions rather than a formalism. The values are: Y) yes; N) no; and I) "implied" by the (tool) support for dialogue. Methods with a tool supporting dialogue rely on the dialogue support to support the social processes. DOGMA-MESS proposed a meaning evolution system. The two methods described in [33] and [38] defined processes for achieving consensus and both OntoEng and the Unified Method mentioned the use of - amongst others - brainstorming sessions to elicit knowledge for the formal descriptions. On-To-Knowledge does not provide such support as the dialogue framework is only used for the elicitation of competency questions. The dialogue is thus used to agree what questions should be supported by the ontology, but not how the ontology should look like.
- **Tool support for social processes on formal descriptions?** This aspect compares to what extent the method provides tool support for some of the processes described for the previous point. The values are: Y) yes; N) no or not proposed; P) partial; I) "implied" by the (tool) support for dialogue; and -) not applicable as there is no tool support or the authors referred to other existing (type of) tools.
  DILIGENT, HCOME and NeOn have taken argumentation frameworks into account, which are reflected in the tool support and therefore these processes can be considered implied. Web Protégé offers support for dialogue in a forum-like manner, and recently provided special requests for formal changes that are fairly frequent in the medical domain [2]. Also the Business Semantics Glossary for BSM supports dialogue via their wiki technology. DOGMA-MESS is considered to provide specific tool support for their meaning evolution support system (MESS) module. In DOGMA-MESS, so-called "tickets" are sent around to the stakeholders for rendering their perspectives [12]. The stakeholders receive the assignment to provide their perspective that are then stored on the server. The meaning negotiation processes for evolving the ontology, however, were only described and not implemented in a tool. Only some support for analyzing conflicts between the different perspectives was proposed to lead the MESS process, usually a knowledge engineer or core domain expert [12].
- **Social processes for agreements on informal descriptions?** This aspect investigates to what extent methods explicitly describe or prescribe special social processes for agreeing on informal descriptions of concepts. The values are: Y) yes; N) no or not proposed; P) partial; and I) "implied" by the (tool) support for dialogue. Again here, the methods with a tool supporting dialogue rely on the dialogue system to support the social processes. Only OntoEng proposed social processes for the construction of natural language definitions of concepts, albeit as keywords such as "brainstorming". Unfortunately, OntoEng does not propose tool support for these processes. This aspect for Ontology 101 (collab) is considered partial as the authors proposed special interactions for formal changes, but not for informal descriptions. They did, however, provide a request to introduce terms in which they foresaw a field for a natural language definition.
- **Tool support for social processes on informal descriptions?** This aspect compares to what extent the method provides tool support for some of the processes described for the previous point. The values are: Y) yes; N) no or not proposed; I) "implied" by the (tool) support for dialogue; and -) not applicable as there is no tool support or the authors referred to other existing (type of) tools.
- **Agreement leads to ontology evolution?** This aspect looks to what extent agreements lead to ontology evolution. Ideally, agreements should automatically lead to ontology evolution. The values for this aspect are: A) automatically; M) manually; N) not; and -) not applicable as method is not explicitly intended for distributed collaboration.
  Most methods that take the collaborative aspects of ontology engineering into account manually evolve the ontology after agreement has reached. Only Ontology 101 (collab) proposed some special requests which – after agreement – automatically evolves the ontology [2]. The requests are stored as annotations in the ontology. The only method not really taking this aspect into account is BSM. The wiki sup-

porting this method [17] allows anyone with sufficient rights to add new knowledge, without discussion. The formal parts of the ontology, however, do have a status attribute stating whether some part is a candidate, accepted, etc. The informal parts of the ontology do not have such properties. The wiki paradigm allows someone to make changes and discussions to happen afterwards. This approach has several problems:

- The ontology is not guaranteed to be stable at any time. The authors actually state that a stable version of the ontology has to be then "compiled" for use for the specific interoperability requirements (e.g., into UML, XSD, etc.).
- Community members could already commit to the knowledge they entered, even it has not been accepted yet. This would hamper the possibility of finding compromises.

The reason why most methods require the manual evolution of ontology is that either someone elicits knowledge without tool support and then enters the results or the argumentation frameworks allow users to discuss issues, solutions, etc. rather than discuss changes. If the latter would have been adopted, motivating and discussing the change, then ontology evolution could be automated.

- **"Owner" of the ontology?** This aspect compares who the "owners" of an ontology are for a particular method. Here, the word "owner" refers to the users who can change the ontology. The values for this aspect are: C) the community of stakeholders (possibly including knowledge engineers) are the owner; H) stakeholders are the owner of their ontology, knowledge engineering ensure the evolution of the shared space; K) knowledge engineers have ownership; N) not proposed; and -) not applicable as method is not explicitly intended for distributed collaboration;
BSM is the only method that allows a community to develop and maintain their ontologies. In most methods, the knowledge engineers to be the owners of the ontology. DOGMA-MESS, HCOME and DILIGENT allow individual stakeholders to maintain a "local" view on matters, but the shared perspective is managed by the knowledge engineers. DOGMA-MESS has the notion of organizational ontologies, HCOME refers to it as personal spaces and DILIGENT as local ontologies. A board of stakeholders with sufficient rights will then try to find a consensus or compromise from the different perspectives to evolve the ontology. The stakeholders remain thus owner of their ontology.
The problem with this method is that even though a consensus is sought, people describe their perspective on matters in a formal way and could thus already commit to their own descriptions. Not only that, they could as well already annotate their existing systems with their predicates. Rather than discussing changes in the ontology, changes are performed locally and then discussed upon. And one would indeed benefit from keeping as much as possible their desired changes. This could thus hamper or delay reaching a consensus as it is possible that stakeholders need to revert and commit to the new version of the ontology as decided upon by the board (with the involvement of all stakeholders, of course). OntoEng did not explicitly state who the owners of an ontology are. The owner of the ontology in Ontology 101 (collab) is presumed to be the knowledge engineer as the authors did not explicitly refer to a method in their papers describing Web- and Collaborative-Protégé, but a case study in the medical domain hinted the use of knowledge engineers [57].

One can conclude from above that the methods and tools described in the state-of-the-art do not take into account the social processes for constructing natural language definitions of concepts. Most of the methods that provide support for social interactions rely on this without specifying any specific processes or considering the natural language definitions as an equal import artifact next to the ontology. Definitions are often seen as annotations to the ontology (e.g., comments).

Web- and Collaborative-Protégé are not methods, but a tools. These tools were taken into consideration as Ontology 101 (collab) as they were developed by the same group that proposed Ontology 101 and developed Protégé. It is interesting to note that in this comparison table, only the authors of Collaborative Protégé propose the formalization of specific requests to evolve the ontology. This is an important step towards agreement evolving ontologies as changes are discussed, and not issues.

Also apparent is that most methods rely on knowledge engineers are the owner of the ontology (either immediately, or via a setting in which they own the shared part). Ontologies, however, should belong to the community and the role of knowledge engineers should be reduced to a minimum or even removed. As Heylighen noted in [31]: "If the process were directed by a single individual (say, the group leader), who imposes a consensus view on the others, then that perspective would not be more powerful than the perspective of the leading individual. In other words, the collective would not be in any way more intelligent than its leader."

3 Hybrid Ontology-engineering Framework

Modeling of ontologies within a community of stakeholders and designers is a critical activity for the eventual success of semantic interoperability. Fundamental to our approach is the involvement of structured natural language as a vehicle to elicit useful and relevant concepts from community communication, and the mapping of these social processes to evolutionary processes in the emerging ontology. The formalism and language presented here are therefore upstream from the usual ontology languages such as RDF(S) and OWL and should not be confused with those; in fact it is relatively straightforward to compile the resulting ontologies into, for example, RDF(S) and OWL at any time.

One fundamental principle of all large system designs is the so-called *separation of concerns* resulting in architectures that delegate respective functionalities to the stakeholders responsible for them. For example, modules are provided by the (generic) architecture of information systems driven by a database and largely separate the concern of basic data management from that of application development, the famous paradigm of data independence.

We reapply this principle in our approach by the rigorous separation in conceptualizations of "fact modeling" from all application-specific interpretations. It is this interpretation process (formally, of statements shared in the application system in terms of ontology concepts) that is usually called "reasoning" in the Semantic Web literature. However, there is little or no attention to such separation of concerns in the usual reasoning formalisms of Semantic Web in terms of Description Logics and its syntactical manifestations such as OWL and its dialects. In our approach, this interpretation is exclusively delegated to the mapping between application system and the "lexon base" of the ontology. We shall call these mappings ontological commitments after [26], but we shall reify them in a well-defined manner suited to our formalism[2]. Intuitively, our commitments select the fact types needed, map application symbols to ontology concepts, and contain the rules and constraints – expressed in ontology terms – under which application symbols, relationships and business rules must be interpreted when they are to be shared with other autonomous systems. Those systems will share the concepts, but of course will have their own symbols, business rules, etc.

This separation of concerns allows a natural introduction of formalized social processes in goal-oriented communities such as exist in enterprises, professional networks, standardization groups, etc. In fact, this is true in any "human agent" context for which agreement about fact types is more efficient than reasoning from axioms. Note that nearly all data models for databases and business information systems were arrived at in this manner for the last 50 or so years.

In [50] a formalism and method for ontology development called DOGMA[3] was defined that illustrated and implemented these principles, now lifted to domain level from the mere enterprise system level. As indicated above, such descriptions must be seen as different from their eventual implementations. In the method and lifecycle of semantic systems, the creation of DOGMA ontology descriptions belongs upstream from such implementation - although of course in many cases one will have to "mine" or elicit the required knowledge from existing information systems and their enterprise environments.

Definition 1 (DOGMA Ontology Descriptions) A DOGMA Ontology Description Ω is an ordered triple $\langle \Lambda, ci, K\rangle$ where Λ is a lexon base, i.e. a finite set of lexons. A lexon is an ordered 5-tuple $\langle \gamma, t_1, r_1, r_2, t_2\rangle$ where $\gamma \in \Gamma$ is a context identifier, $t_1, t_2 \in T$ are terms, and $r_1, r_2 \in R$ are role labels. A lexon is a binary fact type that can be read in two directions: t_1 playing the role of r_1 on t_2 and t_2 playing the role of r_2 on t_1. Here, the usual alphabets for constructing the elements of $T \cup R$ are omitted for simplicity. $ci : \Gamma \times T \to C$ is a function mapping pairs of context identifiers and terms to unique elements of C, a finite given set of concepts. K is a finite set of ontological commitments. Each commitment is an ordered triple $\langle \sigma, \alpha, c\rangle$ where $\sigma \subset \Lambda$ is a selection of lexons from the DOGMA ontology description, $\alpha : \Sigma \to T$ is a mapping called an annotation from the set Σ of application (information, system, database) symbols to terms occurring in that selection, and c is a predicate over $T \cup R$ of that same selection expressed in a suitable FOL language.

Context identifiers are pointers to the origin of a lexon, and helps with the disambiguation of term- and role-labels. Within a context $\gamma \in \Gamma$ and $t \in T$, $ci(\gamma, t)$ is the definition itself of the concept agreed by all users. To emphasize this explicit agreement, we shall avoid labeling concepts as such in our formalism, and assuming they are "computed" by the community from the term labels.

Example 1 provides an example of a set of lexons and a commitment.

[2] We do, however, capture what part of the conceptualization and its axiomatization should be present in all commitments to ensure proper semantic interoperation between the different systems (we refer to Section 3.1 for more details).

[3] Developing Ontology Guided Methods and Applications

Example 1 Assuming lexon base Λ containing the following lexons:

- ⟨*Cultural Domain Expert 1, Artist, with, of, Name*⟩
- ⟨*Cultural Domain Expert 1, Artist, with, of, First Name*⟩
- ⟨*VCard, VCard, with, of, Email Address*⟩
- ⟨*Cultural Domain Expert 2, Artist, with, of, Age*⟩
- ⟨*Cultural Domain Expert 1, Artist, born on, of birth of, Date*⟩
- ⟨*Offer #1 of Organization A, Offer, with, of, Title*⟩
- ⟨*Offer #1 of Organization A, Offer, valid, for, Date*⟩
- ⟨*RFP Documentation, RFP, with, matches, Offer*⟩
- ⟨*FOAF, Agent, with, of, Name*⟩
- ⟨*Cultural Domain Expert 3, Artist, contributing to, with contribution of, Sculpture*⟩
- ...

All context-term pairs evoke concepts, which are referred to by the *ci* function. The owner of an information system with a relational database with a table `ARTIST` with a field `FNAM` containing the names of an Artist could commit to this lexon base with a commitment $\kappa \in K$ by selecting the lexon (selection σ): ⟨*Cultural Domain Expert 1, Artist, with, of, First Name*⟩ , constrain this lexon stating that artists have at most one first name (constraints c): *EACH Artist with AT MOST 1 First Name* , and finally annotate the field in this table with this lexon (annotations α): *MAP 'ARTIST'.'FNAM' ON First Name of Artist.*

Note that the separation of concerns mentioned in the previous Section is reflected here through the set of plausible fact types in the lexon base on one side, and the constraints, rules, etc. on a relevant selection of those lexons on the other. In fact there are no constraints or any other reasoning supports included in the lexon base, making for a so-called light ontology.

As the "unique concept" property mentioned above informally and intuitively results from a community agreement, we argued to formalize a community as such a context [18]. We introduced the notion of a hybrid ontology in which the context identifiers refer to communities and introduced a special linguistic resource – called *glossary* – to support the social processes leading to such agreements.

Definition 2 (Hybrid Ontology Description) A Hybrid Ontology Description is an ordered pair $H\Omega = \langle \Omega, G\rangle$ where Ω is a DOGMA ontology description in which the contexts in Γ are labeled communities and G is a glossary. G is an ordered triple $\langle g_1, g_2, EQ_G\rangle$, where: g_1 is a finite set of functions of the form $g_1 : \Gamma \times T \to Gloss$, the Term Glossary; g_2 is a finite set of functions of the form $g_2 : \Lambda \to Gloss$, the Lexon Glossary; $Gloss$ is a set of human-interpretable objects; EQ_G is a finite set of pairs $Gloss \times Gloss$ containing the agreement that two glosses refer to the same concept.

EQ_G will automatically contain $\langle \phi, \phi\rangle$ for each gloss $\phi \in Gloss$. Below we will give an example of a Hybrid Ontology Description.

Example 2 Given the DOGMA Ontology Description Ω of Example 1, the contexts of the lexons in the lexon base are restricted to communities. The lexon base Λ will thus look as follows:

- ⟨*Cultural Domain Community, Artist, with, of, Name*⟩
- ⟨*Cultural Domain Community, Artist, with, of, First Name*⟩
- ⟨*Address Community, VCard, with, of, Email Address*⟩
- ⟨*Cultural Domain Community, Artist, with, of, Age*⟩
- ⟨*Cultural Domain Community, Artist, born on, of birth of, Date*⟩
- ⟨*Vendor Community, Offer, with, of, Title*⟩
- ⟨*Vendor Community, Offer, valid, for, Date*⟩
- ⟨*RFP Community, RFP, with, matches, Offer*⟩
- ⟨*Address Community, Agent, with, of, Name*⟩
- ⟨*Cultural Domain Community, Artist, contributing to, with contribution of, Sculpture*⟩
- ...

With this lexon base, a glossary G to construct the Hybrid Ontology Description could look as follows:

- $g_1 = \{$⟨⟨*Cultural Domain Community, Date*⟩, *"The day of the month or year as specified by a number."*⟩, ⟨⟨*Vendor Community, Date*⟩, *"Time stated in terms of the day, month, and year."*⟩,...$\}$
- $g_2 = \{$⟨⟨*Cultural Domain Community, Artist, contributing to, with contribution of, Sculpture*⟩, *"The part played by an artist in bringing about a result."*⟩, ...$\}$
- $EQ_G = \{$⟨"The day of the month or year as specified by a number.", "Time stated in terms of the day, month, and year."⟩, ...$\}$

3.1 The Commitment Layer

In the hybrid ontology-engineering framework, we distinguish two types of ontological commitments. The first is called a *community commitment* and is introduced to add structure to the agreement processes to construct ontologies for reaching semantic interoperability. It is a selection of lexons of the lexon base together with a set of constraints on this selection. The selection of lexons and constraints should capture the intention of the types of applications that will commit to the community commitment. Agreements on constraints are focused on identifying a set of attributes that uniquely and totally identifies instances of a concept (as we will describe later on); a necessity for proper interoperation. Indeed, the number of concepts that should have such a "reference-structure" depends on the semantic interoperability requirements of the community.

The second type, called an *application commitment*, will i) commit to one or more community commitments and ii) provide mappings of its application symbols to terms and relations in the selection. In addition, iii) an application commitment can commit to other lexons and constraints not necessarily appearing in a community commitment. They may provide more information on how this application uses the concepts (e.g., in terms of extra constraints), or even application-specific knowledge to ensure that the information inside that application is properly connected. An example would be the annotation of join-tables and identifiers that are specific to the application.

Fig. 1 depicts the different "layers" of the hybrid ontology engineering framework graphically.

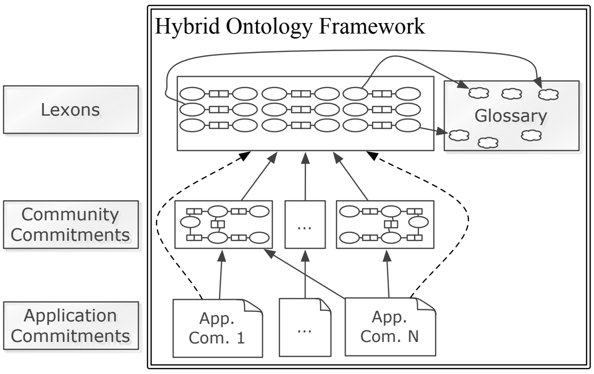


Fig. 1 Different "layers" of the hybrid ontology-engineering framework.

3.1.1 Constraints in a Commitment

The constraints in a commitment are largely based on Object Role Modeling (ORM) [29] constraints. ORM is a fact-oriented method for performing information analysis at the conceptual level. We will describe the constraints we have adopted, as well as constraints introduced for the purpose of GOSPL.

From ORM, the constraints necessary for creating referable terms are taken into account. A term is referable when that term is either lexical (thus its instances can be printed on a screen) or if that term has a unique reference, a set of attributes that uniquely and totally identify instances of concepts referred to by this term. The terms referred to in those attributes (i.e., played by the co-role of the term) have to be referable as well. "Uniquely" means a role played at most once by every instance; "Totally" means that the role is mandatory, this role has to be played by every instance (a mandatory constraint); "Identifying" means that every combination of instances of concepts referred to by each term playing the co-role refers to only one instance.

Assume that a floor is uniquely and totally identified by its floor number and the floor number is lexical in some community C1. Then the constraints will look as follows:

```
<C1, Floor, with, of, Floor Number>
EACH Floor with AT MOST 1 Floor Number.
EACH Floor with AT LEAST 1 Floor Number.
EACH Floor IS IDENTIFIED BY (Floor Number of Floor).
EACH Floor Number IS LEXICAL.
```

Floor thus has a unique simple reference. A simple reference is one unique, total and identifying attribute. A unique composite reference has more than one attribute. Given the description of floor from above, assume that each hotel room is uniquely and totally identified by its room number (which is lexical) and the floor in the same community. This would look as follows:

```
<C1, Hotel Room, with, of, Room Number>
<C1, Hotel Room, with, of, Floor>
EACH Hotel Room with AT MOST 1 Room Number.
EACH Hotel Room with AT LEAST 1 Room Number.
EACH Hotel Room with AT MOST 1 Floor.
EACH Hotel Room with AT LEAST 1 Floor.
EACH Hotel Room IS IDENTIFIED BY
     (Floor of Hotel Room) AND
     (Room Number of Hotel Room).
EACH Room Number IS LEXICAL.
```

Sometimes the instances of a lexical term are limited to a certain set, finite or not. These value constraints can be part of the domain and even shared. A value constraint is described in terms of a value range, which can be an explicit enumeration of elements, ranges or even regular expressions. For example, if we want to limit the occurrences of category type to "Single" and "Double", we have:

```
<C1, Hotel Room, with, of, Category Type>
EACH Category Type IS LEXICAL.
EACH Category Type IN ('Single', 'Double').
```

The interpretation of some role-label combinations can be constrained by the community. For instance, the 'Cultural Domain' community can declare that the combination "is a / subsumes" refers to the taxonomic relation with:

```
INTERPRET 'Cultural Domain Community' is a / subsumes
AS TAXONOMY.
```

By doing so, all occurrences of lexons in the commitment of that community with that role-label combination are interpreted as such. The same can be done with meronomic relations. By default, the "is a / subsumes" is considered the taxonomic relation. Note that these interpretations are grounded with the community.

3.2 Synonyms within and across Communities

Two communities $\gamma_1, \gamma_2 \in \Gamma$ can agree that their respective terms $t_1, t_2 \in T$ refer to the same concept. This agreement is capture with the relation $\equiv_C$, $ci(\gamma_1, t_1) \equiv_C ci(\gamma_2, t_2)$ is the agreement of both communities that their respective terms refer to the same concept. Note that if $\gamma_1 = \gamma_2$, it refers to this type of agreement within one community.

Example 3 Given the Hybrid Ontology Description in Example 2, both vendor- and cultural domain community can agree that their term “Date”, which happens to be used by both communities, refers to the same concept. By agreeing, they assert that *ci(’Vendor Community’, “Date”)* $\equiv_C$ *ci(’Cultural Domain Community’, “Date”)* is part of both community commitment’s constraints.

We note that assertions of gloss-equivalences and synonymy are only symmetric, reflexive and transitive – i.e. an equivalence relation – *within one agreement process.* This constraint was put in place to avoid synonymy and gloss-equivalences to be propagated without each of the communities validating the new relations inferred from these assertions. These new relations can, however, be analyzed to make additional $\equiv_C$ assertions if those concepts indeed are synonymous across two ore more agreement processes.

The $\equiv_C$ agreements are stored in the community commitments. When one wishes to extend an application commitment with application-specific knowledge about one of the shared concepts in the community commitment, however, he also needs to make explicit that the term he’s using from this concept is synonymous to that of the community. For instance, assume that a particular application is committing to the term “Artist” in the cultural domain community and wishes to annotate his application-specific identifier belonging to instances of artist in his database, he would add following lexon and synonym statement to his application commitment (where the origin of this lexon is his organization):

```
<’MyOrganization’, Artist, with, of, AID>
LINK(’Cultural Domain Community’,Artist,
     ’MyOrganization’,Artist).
```

3.3 Implementing Commitments in OWL and DL

We now present a lossless schema transformation for community commitments in GOSPL in the Description Logic (DL) dialect DL-Lite$_{A,id}$ [7]. A lossless schema transformation is a transformation of a schema that allows one to preserve each permitted population. In other words, the populations can be reconstructed unambiguously. As a consequence, a bijective mapping between both sets of permitted populations must exist. The “losslessness” of a transformation needs to be shown.

There are several attempts to bridge the gap between fact-oriented modeling approaches and logical formalisms. Notable works include [32] and [23], presenting a translation into OWL 2[4] of ORM graphical representations. Most of them identify a subset of ORM that can be semantically translated to DLs. Authors in [32] directly transfer ORM representations to DL formulas case by case. In [23], the corresponding linear syntax was introduced. The syntax is then given set-theoretic semantics and translation to DLs. But instead of a direct translation, this approach uses some sort of schema transformation in which the fact type is reified (i.e. becomes a concept) and the object types playing role inside that fact type are now playing roles with the newly introduced concept. Both approaches try to capture the intended meaning of ORM representations and express them in DL formulas. However, we will show later in this Section that both approaches are not lossless.

To demonstrate that our proposed translation into DL-Lite$_{A,id}$ is lossless, we translate both the model in the community commitment and the translation in DL-Lite$_{A,id}$ in a set of first-order logic (FOL) formulas. The first is done by adopting the formalization provided by [28] and the latter by [4]. Then we show that the extensions of both sets of FOL formulas are equal. In other words, we will show that one is a logical consequence of the other and vice versa.

Description logics are a decidable fragment of FOL. Concept names are unary predicates and role names are binary predicates. Concept descriptions correspond to FOL formulas with one free variable, which will be bound when used in a concept inclusion statement [4]. The translation of assertions in a DL into FOL formulas are provided by [4]. The translation of concept description C into a FOL formula with one free variable $\tau_x(C)$ is defined as follows:

1. $\tau_x(A) := A(x)$ for all concept names A
2. $\tau_x(C \sqcap D) := \tau_x(C) \wedge \tau_x(D)$
3. $\tau_x(C \sqcup D) := \tau_x(C) \vee \tau_x(D)$
4. $\tau_x(\neg C) := \neg \tau_x(C)$
5. $\tau_x(\forall r.C) := \forall y(r(x, y) \to \tau_y(C))$, where $y \neq x$
6. $\tau_x(\exists r.C) := \exists y(r(x, y) \wedge \tau_y(C))$, where $y \neq x$

Given a TBox $\mathcal{T}$ with concept-inclusions, the translation $\tau(\mathcal{T})$ of $\mathcal{T}$ is given by:

$$\tau(\mathcal{T}) := \bigwedge_{C \sqsubseteq D \in \mathcal{T}} \forall x(\tau_x(C) \to \tau_x(D))$$

[4] http://www.w3.org/TR/owl2-overview/

3.3.1 Lexon

The translation of a lexon into DL-Lite$_{A,id}$ is shown below. To prove that in this translation is lossless, we will demonstrate their equivalence. To do so, we will first translate both the fragments in DOGMA and their proposed translation in DL in FOL and use a semantic tableau[5] to demonstrate that $\models \Sigma \leftrightarrow \Phi$ holds.

Before we translate, we need to state that all roles with the same label are translated in such a way the labels become unique. For instance, in $\langle \gamma, A, r, s, B \rangle$ and $\langle \gamma, C, r, s, D \rangle$ the labels of the first roles are the same, but the roles are different: the first has domain A and range B, the second domain C and range D. During translation, they are thus transformed into r_1 and r_2 and the same happens for both roles with label s.

A lexon $\langle \gamma, A, R, S, B \rangle$ is translated into FOL results in the following formulas Σ:

$$\forall x(\forall y(R(x,y) \rightarrow (A(x) \wedge B(y)))) \tag{1}$$

The corresponding DL-Lite$_{A,id}$ statements are:

$$\exists R.\top \sqsubseteq A \tag{2}$$

$$\exists R^{-}.\top \sqsubseteq B \tag{3}$$

Translated into FOL, the statements in Φ look as follows:

$$\forall x(\exists y R(x,y) \rightarrow A(x)) \tag{4}$$

$$\forall x(\exists y R(y,x) \rightarrow B(x)) \tag{5}$$

The semantic tableaus in Fig. 2 and 3 both close, meaning there are no counterexamples. Since one is a consequence of the other and vice versa, both sets of FOL formulas are equivalent. As they are equivalent, so are their possible extensions. Because of this equivalence, it follow naturally that both sets of formulas are population equivalent.

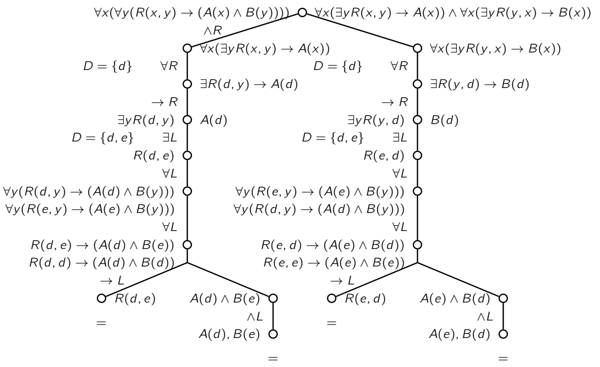

$\forall x(\forall y(R(x,y) \rightarrow (A(x) \wedge B(y))))$
$\forall x(\exists y R(x,y) \rightarrow A(x)) \wedge \forall x(\exists y R(y,x) \rightarrow B(x))$
$\wedge R$
$\forall x(\exists y R(x,y) \rightarrow A(x))$
$\forall x(\exists y R(y,x) \rightarrow B(x))$
$D = \{d\}$ $\forall R$
$\exists R(d,y) \rightarrow A(d)$
$D = \{d\}$ $\forall R$
$\exists R(y,d) \rightarrow B(d)$
$\rightarrow R$
$\exists y R(d,y)$
$A(d)$
$\rightarrow R$
$\exists y R(y,d)$
$B(d)$
$D = \{d,e\}$ $\exists L$
$R(d,e)$
$D = \{d,e\}$ $\exists L$
$R(e,d)$
$\forall L$
$\forall y(R(d,y) \rightarrow (A(d) \wedge B(y)))$
$\forall y(R(e,y) \rightarrow (A(e) \wedge B(y)))$
$\forall L$
$\forall y(R(e,y) \rightarrow (A(e) \wedge B(y)))$
$\forall y(R(d,y) \rightarrow (A(d) \wedge B(y)))$
$\forall L$
$R(d,e) \rightarrow (A(d) \wedge B(e))$
$R(d,d) \rightarrow (A(d) \wedge B(d))$
$\forall L$
$R(e,d) \rightarrow (A(e) \wedge B(d))$
$R(e,e) \rightarrow (A(e) \wedge B(e))$
$\rightarrow L$
$R(d,e)$
$=$
$A(d) \wedge B(e)$
$\wedge L$
$A(d), B(e)$
$=$
$\rightarrow L$
$R(e,d)$
$=$
$A(e) \wedge B(d)$
$\wedge L$
$A(e), B(d)$
$=$

Fig. 2 Semantic tableau to show that $\Sigma \models \Phi$

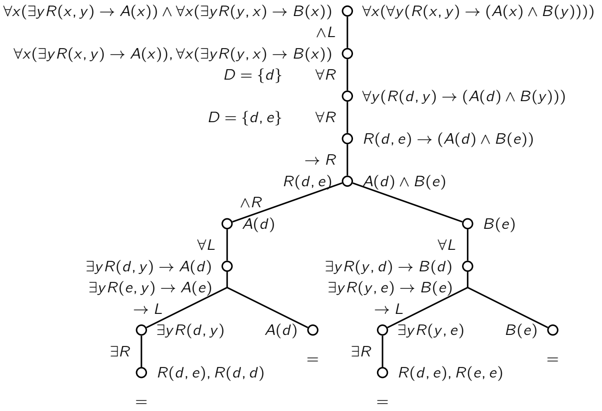

$\forall x(\exists y R(x,y) \rightarrow A(x)) \wedge \forall x(\exists y R(y,x) \rightarrow B(x))$
$\forall x(\forall y(R(x,y) \rightarrow (A(x) \wedge B(y))))$
$\wedge L$
$\forall x(\exists y R(x,y) \rightarrow A(x)), \forall x(\exists y R(y,x) \rightarrow B(x))$
$D = \{d\}$ $\forall R$
$\forall y(R(d,y) \rightarrow (A(d) \wedge B(y)))$
$D = \{d,e\}$ $\forall R$
$R(d,e) \rightarrow (A(d) \wedge B(e))$
$\rightarrow R$
$R(d,e)$
$A(d) \wedge B(e)$
$\wedge R$
$A(d)$
$B(e)$
$\forall L$
$\exists y R(d,y) \rightarrow A(d)$
$\exists y R(e,y) \rightarrow A(e)$
$\forall L$
$\exists y R(y,d) \rightarrow B(d)$
$\exists y R(y,e) \rightarrow B(e)$
$\rightarrow L$
$\exists y R(d,y)$
$A(d)$
$=$
$\exists R$
$R(d,e), R(d,d)$
$=$
$\rightarrow L$
$\exists y R(y,e)$
$B(e)$
$=$
$\exists R$
$R(d,e), R(e,e)$
$=$

Fig. 3 Semantic tableau to show that $\Phi \models \Sigma$

3.4 Mandatory Constraints

The translation of a mandatory constraint into DL-Lite$_{A,id}$ is shown below. As the reader can see below, both translations in FOL yield the exact same formula and are thus equivalent. The constraint that states that *EACH A R's AT LEAST 1 B* is translated into FOL as follows: $\forall x(A(x) \rightarrow \exists y(R(x,y)))$. The corresponding DL-Lite$_{A,id}$ statement is $A \sqsubseteq \exists R.\top$. Translated into FOL, this statement looks as follows: $\forall x(A(x) \rightarrow \exists y(R(x,y)))$

3.5 Internal Uniqueness Constraints

In DL-Lite$_{A,id}$, a role functionality assertion expresses the functionality of a role. An attribute functionality assertion expresses the functionality of an atomic attribute.

Since we only have lexons, we have to examine only five cases: 1) no external uniqueness constraint, 2) one spanning only the first role, 3) one spanning only the second role, 4) one spanning the first role and one spanning the second role (a so-called one-to-one relation)

[5] Semantic tableaux are an efficient and convenient means to test whether a formula ϕ is a logical consequence of a set of formulas Σ in FOL. This is done by trying to make ϕ false with respect to Σ by looking for counterexamples for the sequent $\Sigma \circ \phi$. A sequent are two sets of formulas $\phi_1, ..., \phi_n$ and $\psi_1, ..., \psi_m$ separated by the symbols $\circ$. A evaluation V is called a counterexample of a sequent $\phi_1, ..., \phi_n \circ \psi_1, ..., \psi_m$ if $V(\phi_1) = ... = V(\phi_n) = 1$ and $V(\psi_1) = ... = V(\psi_m) = 1$. When a formula ϕ occurs on both sides of the sequent, the evaluation of ϕ returns both 1 and 0. In that case, the sequent contains a contradiction and thus has no counterexample.
Formulas on the LHS of a sequent have to be made true and formulas on the RHS of a sequent false. For instance, in $\Sigma, \alpha \wedge \beta \circ \Pi$, $\alpha \wedge \beta$ is true if and only if both α and β are true, which then yields the subproblem $\Sigma, \alpha, \beta \circ \Pi$ (by using the $\wedge L$ rule, where the 'L' stands for left). A branch is considered closed if it contains the same formula both on the LHS and RHS in one of the sequents of a branch. Otherwise the branch is open and a counterexample is found. A counterexample is a model that makes the LHS of the top sequent true, but the RHS false.

and finally 5) one spanning both roles. Cases 1 and 5 are the same. In this Section, we thus only consider uniqueness constraints over the first role. The third case is the same as the second merely needs to be reversed and the fourth is a combination of cases 2 and 3.

The translation of *EACH A R'S AT MOST 1 B* into FOL is $\forall x(\forall y(\forall z((R(x,y) \wedge R(x,z)) \rightarrow y = z)))$. In DL-Lite$_{A,id}$, this constraints is asserted with $(funct\ R)$. In turn, its translation into FOL yields the same formula: $\forall x(\forall y(\forall z((R(x,y) \wedge R(x,z)) \rightarrow y = z)))$. Again, the translation of DOGMA and DL-Lite$_{A,id}$ into FOL are equivalent.

3.6 External Uniqueness Constraints

To implement external uniqueness constraints in DL, we use identification assertions. Identification assertions were first introduced in [7]. Identification assertions are of the form $(id\ B\ \pi_1, ..., \pi_n)$, where B is a basic concept and every π_i is a path. A path is either: an atomic role or the inverse of an atomic role; an atomic attribute or the inverse of an atomic attribute; a composition of two paths π_a, π_b denoted as $\pi_a \circ \pi_b$, where $\circ$ denotes the composition operator on two paths; a test relation 'D?' representing the identity relation on instances of D (either a basic concept or a value-domain). Test relations are used to impose involving instances of a certain concept or value-domain in the paths. At least one of the paths in an identification assertions has to have a length of one, i.e., be an atomic role or attribute (or the inverse thereof).

The interpretation of an identification constraint states that for any two instances of $a_1, a_2 \in I(A)$, if the intersection of the interpretation of each path π in the identification constraint for these two instances are not empty, then these two instances are actually the same instance.

The translation of *EACH A IS IDENTIFIED BY (B1 S1 A) AND ... AND (Bn Sn A)* into FOL looks as follows:

$$\forall x_1(\forall x_2(\forall y_1(...\forall y_n((R_1(x_1,y_1) \wedge ... \wedge R_n(x_1,y_n) \\ \wedge R_1(x_2,y_1) \wedge ... \wedge R_n(x_2,y_n)) \rightarrow x_1 = x_2)...))) \quad (6)$$

The statement in DL-Lite$_{A,id}$ that corresponds with this statement is $(id\ A\ R_1\ ...\ R_n)$, which again yields in the same translation into FOL.

3.6.1 Subtyping

Translating concept hierarchies in DOGMA and DL-Lite$_{A,id}$ into FOL is straightforward. The translation of a subtype declaration – shown below – into FOL provided by [28] is the same as the translation of the corresponding concept-inclusion in DL-Lite$_{A,id}$, and thus equivalent. The lexon $\langle\gamma$,*A*,*is a*,*subsumes*,*B*$\rangle$ is translated into FOL as follows: $\forall x(B(x) \rightarrow A(x))$. Which is also the FOL translation of the corresponding DL-Lite$_{A,id}$ statement: $B \sqsubseteq A$.

The problem with subtyping, however, is that the instances of all object types that are not the child in a taxonomic relation are considered to be disjoint. Halpin calls these object types *primitive* [28]. For any conceptual schema, there will be a finite number of such primitive object types. The disjiontness of the instances of these object types are given with the following rule: given $A_1, ..., A_n$ primitive object types

$$\forall x(\neg(A_1(x) \wedge A_2(x)) \wedge \neg(A_1(x) \wedge A_3(x)) \wedge \\ ... \wedge \neg(A_{n-1}(x) \wedge A_n(x))) \quad (7)$$

In other words, it is prohibited for an instance to be a member of two object types from the set of primitive object types.

In order to be population equivalent, this same restriction needs to be modeled in the DL language we have adopted. The problem, however, is that the DL-Lite dialect we have adopted has no means for describing disjointness in an explicit way. In order to solve this, the disjoint concepts need to be modeled via binary Horn inclusions. In other words: for every two concepts A, B that are disjoint, the following concept-inclusion needs to be asserted $A \sqsubseteq \neg B$, which states that an instance of A is not in an instance of B. It is not necessary to assert $B \sqsubseteq \neg A$ as well, as the translation of both concept inclusion into FOL show that both formulas are equivalent as one is a quantification of the contraposition of the other formula: $\forall x(A(x) \rightarrow \neg B(x)) \leftrightarrow \forall x(B(x) \rightarrow \neg A(x))$.

For every two object types in the primitive object types of the DOGMA model, such a concept-inclusion is added in the translation into DL-Lite$_{A,id}$. Now we need to show that the translation of these concept-inclusions into FOL is equivalent with the FOL formulas in equation (7). We show this by means of the two semantic tableaus in Fig. 4.

3.7 Relation with Related Work

In the last few years, several authors addressed the problem of providing an encoding for ORM diagrams in DL knowledge bases [39,36,35,32,23]. Only the work of Keet [39], and Franconi and Mosca [23] can be con-

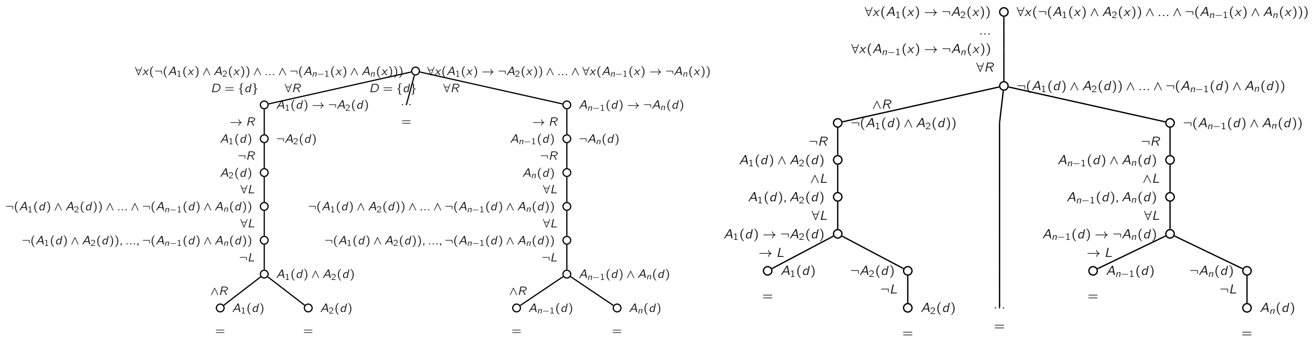

$\forall x(\neg(A_1(x) \wedge A_2(x)) \wedge ... \wedge \neg(A_{n-1}(x) \wedge A_n(x)))$
$\forall x(A_1(x) \to \neg A_2(x)) \wedge ... \wedge \forall x(A_{n-1}(x) \to \neg A_n(x))$
$D = \{d\}$
$\forall R$
$D = \{d\}$
$\forall R$
$A_1(d) \to \neg A_2(d)$
$...$
$=$
$A_{n-1}(d) \to \neg A_n(d)$
$\to R$
$A_1(d)$
$\neg A_2(d)$
$\neg R$
$A_2(d)$
$\forall L$
$\neg(A_1(d) \wedge A_2(d)) \wedge ... \wedge \neg(A_{n-1}(d) \wedge A_n(d))$
$\forall L$
$\neg(A_1(d) \wedge A_2(d)), ..., \neg(A_{n-1}(d) \wedge A_n(d))$
$\neg L$
$A_1(d) \wedge A_2(d)$
$\wedge R$
$A_1(d)$
$A_2(d)$
$=$
$=$
$\to R$
$A_{n-1}(d)$
$\neg A_n(d)$
$\neg R$
$A_n(d)$
$\forall L$
$\neg(A_1(d) \wedge A_2(d)) \wedge ... \wedge \neg(A_{n-1}(d) \wedge A_n(d))$
$\forall L$
$\neg(A_1(d) \wedge A_2(d)), ..., \neg(A_{n-1}(d) \wedge A_n(d))$
$\neg L$
$A_{n-1}(d) \wedge A_n(d)$
$\wedge R$
$A_{n-1}(d)$
$A_n(d)$
$=$
$=$
$\forall x(A_1(x) \to \neg A_2(x))$
$\forall x(\neg(A_1(x) \wedge A_2(x)) \wedge ... \wedge \neg(A_{n-1}(x) \wedge A_n(x)))$
$...$
$\forall x(A_{n-1}(x) \to \neg A_n(x))$
$\forall R$
$\neg(A_1(d) \wedge A_2(d)) \wedge ... \wedge \neg(A_{n-1}(d) \wedge A_n(d))$
$\wedge R$
$\neg(A_1(d) \wedge A_2(d))$
$\neg(A_{n-1}(d) \wedge A_n(d))$
$\neg R$
$A_1(d) \wedge A_2(d)$
$\wedge L$
$A_1(d), A_2(d)$
$\forall L$
$A_1(d) \to \neg A_2(d)$
$\to L$
$A_1(d)$
$\neg A_2(d)$
$=$
$\neg L$
$A_2(d)$
$=$
$...$
$=$
$\neg R$
$A_{n-1}(d) \wedge A_n(d)$
$\wedge L$
$A_{n-1}(d), A_n(d)$
$\forall L$
$A_{n-1}(d) \to \neg A_n(d)$
$\to L$
$A_{n-1}(d)$
$\neg A_n(d)$
$=$
$\neg L$
$A_n(d)$
$=$

Fig. 4 Semantic Tableau

sidered to have tackled the problem from a formal perspective[6].

The inadequacy of the mapping prosed in [35,32] can easily be shown by means of semantic tableaus. Note that the translation in [32] contains quite a few syntactical errors, but builds further upon the work presented in [35]. Given the lexon: $\langle \gamma, A, R, S, B \rangle$. The translation proposed in [35,32] is as follows:

$$A \sqsubseteq \forall R.B \tag{8}$$

$$B \sqsubseteq \forall S.A \tag{9}$$

$$R \sqsubseteq S^- \tag{10}$$

Not only is the last statement incorrect and should be replaced with $R \equiv S^-$, after this correction one can find a counter example for the translation of the binary fact type into FOL according to Halpin and the translation of these DL statements into FOL. The formulas below provide the latter translation.

$$\forall x(A(x) \to \forall y(R(x,y) \to B(y))) \tag{11}$$

$$\forall x(B(x) \to \forall y(S(x,y) \to A(y))) \tag{12}$$

$$\forall x(\forall y(R(x,y) \leftrightarrow S(y,x))) \tag{13}$$

Indeed, an interpretation $\mathcal{I}$ with $\mathcal{I}(R) = \{\langle d,e \rangle\}$, $\mathcal{I}(S) = \{\langle e,d \rangle\}$, $\mathcal{I}(A) = \{\}$ and $\mathcal{I}(B) = \{\}$ is a model for above FOL formulas, but not for the translation provided by Halpin: $\forall x(\forall y(R(x,y) \to (A(x) \wedge B(y))))$. In other words, there are counter examples and therefore there is not a bijective mapping between the two.

In [36,39], both Jarrar and Keet provided a translation of ORM into a DL that support n-ary relations where $n \geq 2$, namely the dialect $\mathcal{DLR}_{ifd}$ [8]. The problems with translation proposed by Jarrar were examined by Keet in the second version of this paper[7]. Keet criticized the inaccuracy of Jarrar's work with respect to the syntax and semantics. Franconi, in turn, provides critique on Keet's work on several inaccuracies [23]. Both proposals are thus inadequate for our translation, even though the idea was appealing. However, as DOGMA limits itself to the use of binary lexons and DL-Lite$_{A,id}$ provides constructs for a lossless translation, there is no need for constructs to support arbitrary n-ary relations.

Franconi and Mosca provided a translation of ORM into $\mathcal{ALCQI}$, thus using the DL $\mathcal{ALC}$ extended with qualified cardinality restrictions and inverse roles. In essence, they "reify" fact types with uniqueness constraints spanning two ore more roles by first introducing a new concept and then transform each of the involved roles into a DL role where the domain is the newly introduced concept and the range the object type to which the ORM role connected to. Those new roles are then declared to be functional. Indeed, each instance of that relation only plays each role once. As they claim, their translation is indeed sound and complete. Every model of the ORM translation into FOL is also a model for their DL translation into FOL. But, as will be seen, the inverse is not true.

Their translation is actually a lossy schema transformation (as shown in Fig. 5). The figure presents a binary fact type and its lossless schema transformation using only attributive fact types. Lets call this translation (A). The figure also contains the "corresponding" ORM diagram for the DL translation proposed by Franconi and Mosca. Lets call this translation (B). Assuming that the FOL translation of (A) is contained in Σ and that of (B) in Φ. It is easy to see that due to the additional constraints in (A), every model of Σ is also a model for Φ, but the inverse is not true. Φ is thus a logical consequence of Σ, but not the other way round. Since the sets of formulas are not equivalent, there can

[6] Franconi and Mosca published a technical paper containing more information entitled "The formalisation of ORM2 and its encoding in OWL2". Available from `https://www.inf.unibz.it/krdb/pub/TR/KRDB12-2.pdf`

[7] The first version of her paper was published in 2007 in the Computer Research Repository. Later on, she provided a second version of her paper with corrections, more extensive related work, etc. in 2009. Both versions of the paper can be found here: `http://arxiv.org/abs/cs.LO/0702089`

not exist a bijective mapping between the two. Hence, the translation proposed by Franco and Mosca is not lossless.

```
# The binary fact type
<'OrgA',Person,working_for,employing,Company>
# The lossless schema transformation of this fact
# type using only attributive fact types
<'OrgA',Person,with,of,Contract>
<'OrgA',Company,providing,of,Contract>
EACH Contract of AT MOST ONE Person.
EACH Contract of AT MOST ONE Company.
EACH Contract of AT LEAST ONE Person.
EACH Contract of AT LEAST ONE Company.
EACH Contract IS IDENTIFIED BY (Person of Contract)
    AND (Company of Contract).
# ''Corresponding'' Translation proposed by Franconi
# and Mosca.
<'OrgA',Person,with,of,Contract>
<'OrgA',Company,providing,of,Contract>
EACH Contract of AT MOST ONE Person.
EACH Contract of AT MOST ONE Company.
```

Fig. 5 A binary fact type, its lossless schema transformation using only attributive fact types and the "corresponding" lossy translation by Franconi and Mosca [23].

3.8 The Glossary

Gloss is a set of natural language descriptions – called glosses – each providing an "explanation" for a term in T or a lexon in Λ adequate within a given community.

Guidelines on the construction of glosses were given in [34]. A gloss should i) start with the term of principal or super type of the concept being defined; ii) be written in the form of propositions; iii) focus on the distinguishing characteristics of the concept being defined; iv) be supportive (examples are encouraged); v) be consistent with the formal axioms in the ontology and vi) be sufficient, clear and easy to understand.

When two different terms are articulated with the exact same gloss, one would assume that the glosses and therefore also the described terms refer to the same concept. If this property holds, we call the hybrid ontology consistent. In other words, if two terms in two communities point to exactly the same gloss, then they must refer to the same concept as well. For most purposes, however, this condition is too limiting since often glosses will express "the same thing" without being textually identical. It suffices that the communities agree on their equivalence; this leads to the following definition.

Definition 3 (Gloss-equivalence) Given communities $\gamma_1, \gamma_2 \in \Gamma$ and terms $t_1, t_2 \in T$, the two term-glosses $g_1(\gamma_1, t_1)$ and $g_1(\gamma_2, t_2)$ are said to be *gloss-equivalent* EQ_G if the two communities agree that the described terms refer to the same concept.

Note that there are two special cases of gloss equivalence: one in which the communities are different and the terms are the same (term-equivalence) and one in which the terms are different but the within the same community (community-equivalence). We can now define the definition for the glossary-consistency principle as follows:

Definition 4 (Glossary-consistency principle) A hybrid ontology satisfies the *glossary-consistency principle* if for every two pairs $\langle\gamma_1, t_1\rangle, \langle\gamma_2, t_2\rangle \in \Gamma \times T$, if $EQ_G(g_1(\gamma_1, t_1), g_1(\gamma_2, t_2))$ then $ci(\gamma_1, t_1) \equiv_C ci(\gamma_2, t_2)$. The converse does not necessarily hold.

We do not impose that $EQ_G(g_1(\gamma_1, t_1), g_1(\gamma_2, t_2))$ implies $ci(\gamma_1, t_1) \equiv_C ci(\gamma_2, t_2)$ in order to maintain glossary-consistency, as both communities might carry other agreements with other communities for their respective terms. An agreement between those two communities is sufficient.

Gloss-equivalences are on the level of the glossary whereas $\equiv_C$ agreements are on the level of the formal descriptions of the concepts (i.e. the lexons). We can impose that for $\equiv_C$, the term must appear in a lexon as a term will only be in the community commitment if and only if that term plays at least one role (otherwise the term has no purpose for this community). If the term would end up in a taxonomy, then it plays the role of being the sub- or supertype of another term (e.g., with the role-labels "is a/subsumes"), hence satisfying the condition.

Communities can start gradually building their glossary before formally describing their concepts. However, nothing should prevent community members for having agreements on the "sameness" of descriptions across or within their own community. If the definition would impose $\equiv_C$ on the formal descriptions, the community first needs to agree on at least one lexon concerning that term.

Another reason is validation of the equivalences. The glossary-consistency principle will pinpoint the descriptions used for terms that are EQ_G, but whose terms in those communities are not $\equiv_C$. The glossary-consistency principle does not become a property that needs to hold or else the ontology project fail, instead it becomes a tool to drive the community in establishing $\equiv_C$, double checking whether the gloss-equivalence was not misleading and both terms really do refer to the same concept.

This is particularly handy as the validity of the natural language descriptions and the equivalence of two such descriptions are relative to the communities participating in these discussions. If glosses were not adequate and yet agreed upon, the second agreement while the terms are formally described are more than welcome and the community will be able to rectify the mistakes.

Gloss-equivalence is a symmetrical property as it captures the communities agreeing that their glosses to describe their terms refer to the same concept. Term-adoption, however, is asymmetrical. The definition is given below.

Definition 5 (Term-adoption) Given a hybrid ontology description $H\Omega = \langle \Omega, G \rangle$ and two communities $\gamma_1, \gamma_2 \in \Gamma$ and term $t_1 \in T$, γ_2 is said to adopt $\langle \gamma_1, t_1 \rangle$ when $gloss_1 = g_1(\gamma_1, t_1)$ and $gloss_2 = g_1(\gamma_2, t_1)$ are defined, and we have (i) $EQ_T(gloss_1, gloss_2)$, i.e. first "match" the two glosses; and (ii) $ci(\gamma_2, t_1) \Leftarrow_{cea} ci(\gamma_1, t_1)$, i.e. agree that both concepts are equal with γ_2 also incorporating the meaning agreements inside $ci(\gamma_1, t_1)$.

In this definition $\Leftarrow_{cea}$ allows the adopting community to incorporate the meaning agreements of the other community by asserting $c_i \equiv_C ci(\gamma_2, t_1)$ for every c_i in $cea(ci(\gamma_1, t_1))$. In other words, by adopting the gloss of another community-term pair, the adopting community agrees with all existing $\equiv_C$ agreements the adoptee has with other communities. Term-adoption allows γ_1 and γ_2 to agree their respective glosses refers to the same concept (a symmetric condition) and γ_2 agreeing to use t_1 as a term to refer to γ_1's concept behind it (an asymmetric condition).

4 Hybrid Ontology-engineering Method

In the previous Section, we introduced a framework for hybrid ontology engineering on top of DOGMA, a fact-oriented ontology engineering approach. In this Section, we present the method for hybrid ontology engineering. A method prescribes certain guidelines and steps to be taken to achieve a certain goal; the construction of a hybrid ontology in this paper. The method uses the hybrid ontology-engineering framework defined in the previous Section. We will introduce the social processes as we go along each of the steps of the method. The social processes were defined in [18] and allow a community to alter the hybrid ontology towards a closer approximation of the community's domain.

Fig. 6 summarizes the different phases in GOSPL. Starting communities and their requirements that co-evolve, the informal descriptions of key terms have to be gathered before formally describing those concepts. These formal descriptions can be constrained and then committed to by application by using a commitment language, e.g., Ω-RIDL [71]. During the processes from creating the glossary to committing to the hybrid ontology description, the communities can make agreements on gloss-equivalences and synonyms. The hybrid ontology, and the data described with those commitments can then be re-internalized by the community for another iteration, gradually approximating the domain that needs to be captured by the ontology.

4.1 Defining Semantic Interoperability Requirements

We restrict ourselves to communities of users representing autonomously developed and maintained information systems with a need to exchange information for a purpose. This need is translated into a semantic interoperability requirement (SIR). The objectives of a SIR are to ensure the application or components interoperate with other specified information systems and their components in a meaningful manner. The data needs to be exchanged between those components and be useable upon reception and the different components "know" how to consult the data from other information systems or components. A community is partly identified by its SIRs. As we will see later on while describing the co-evolution between communities and their SIRs, we will identify communities by those requirements and its set of members.

A SIR for a community $\gamma \in \Gamma$ $SIR(\gamma)$ consists of an ordered pair $\langle KT, GO \rangle$: a non-empty set of key terms $KT \subset \Gamma \times T$ and a non-empty set of goals GO for which descriptions of those key concepts are needed. The community interacts and agrees upon the elements in those two sets. The social processes for this phase are:

- Request to add key term
- Request to remove key term
- Request to add goal
- Request to remove goal

4.2 Building the Glossary

Interoperability is achieved by annotating the symbols of an information system with terms and relations in the hybrid ontology. As the hybrid ontology and the glossary are initially empty, we must ask ourselves how these ontologies come to be. We have already described how – in a hybrid ontology – terms are on one hand described "informally" by means of natural language descriptions called glosses for humans and described formally for annotating information systems and their computerized systems on the other. To ensure all members of a community are referring to the same referent

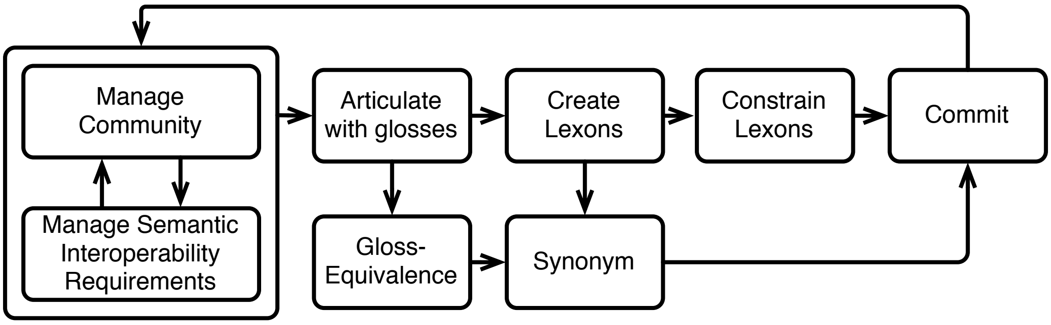


Fig. 6 The GOSPL method

for a particular label, the community needs to align their ideas of the concept symbolized by the term. We call this process *alignment*. Alignment is achieved by (1) describing the concepts referred to by these labels and (2) having the community members agree on one such description per label.

To facilitate alignment, GOSPL imposes terms to be articulated before formal descriptions are added, starting with the list of key terms in the SIR.

In a first iteration, there are no lexons. A community needs to wait for articulating lexons for when start emerging. Lexons can be articulated with a gloss only if both its terms are articulated. In GOSPL, a community is able to articulate all the lexons. However, GOSPL strongly encourages articulating at least those lexons whose internal uniqueness does not span only one role. In other words, GOSPL encourages the articulation of "many-to-many" relations in ER terminology. In the absence of an internal uniqueness constraint, the uniqueness constraint is assumed to be spanning the two roles. Such relations must correspond with a concept in the domain that needs to be approximated by the ontology. This is in contrast with so called "attributive" relations, which can be too "trivial" to fully articulate. Take for instance the lexon ⟨*C1,Person,working for,employing,Organization*⟩ where a person can work for many organizations and an organization can employ many employees. This many-to-many relation could denote the concept of position. In the example of ⟨*C2, Person,born on,of birth of,Date*⟩ with a person born on at most one Date, date (of birth of) becomes an attribute of Person. We therefore don't need to describe the relation as being the occurrence of persons having a birth date. Our claim is that non-attributive relations denote concepts, and therefore need to be described by the community. The relation between concepts and non-attributive relations will become apparent after we will treat the constraint one can put on lexons and reference structures of concepts.

The social processes in this phase are:

- Request to add term-gloss
- Request to remove term-gloss
- Request to change term-gloss
- Request to add lexon-gloss
- Request to remove lexon-gloss
- Request to change lexon-gloss

4.3 The Creation of Lexons

Lexons can only be entered in the lexon base when one of the terms in this lexon has already been articulated. Indeed, it would be undesirable to describe a relation between two terms if both terms playing the roles in that relation are not described themselves, meaning that their intended meaning has not yet been made explicit. If at least one of the terms described, one can assume that the lexon proposed around that term is in function of the informal definition and/or the SIRs.

The social processes in for constructing the domain-conceptualization are:

- Request to add lexon
- Request to remove lexon
- Change supertype of term

The last social process in the list above corresponds with the management of the taxonomy. Changing the supertype of a term when it has already a place in the taxonomy of the hybrid ontology will result in the removal of the previous relation.

4.4 Constraining Lexons

An application commitment contains (1) a selection of community commitments, loading all the lexons and constraints agreed upon by those communities; (2) a selection of lexons added by the application-owner; (3) constraints on lexons (imported of added) that indicate how that particular application uses those concepts;

and (4) mappings from application symbols to terms and roles in that selection.

Some of these constraints have to be shared and agreed upon by the community in order to meet the semantic interoperability requirements. Those constraints should not stem from the individual applications, but be part of the domain that has to be modeled. A classic example of such a constraint is a book being uniquely and totally identified by its ISBN number[8]. Those constraints are needed to ensure proper interoperation between the different systems.

The community thus might need to agree on constraints in order to meet the goals captured by their SIRs. We make a distinction between two types of constraints: on terms and on roles of lexons. In either case, the GOSPL method imposes the terms to be articulated with a gloss. Indeed, it would be undesirable to constrain the use of a term, a role, or a lexon with insufficient articulation, as this means that their intended meaning has not yet been made explicit.

For the social process "Request to change superlexon of lexon (role hierarchy)", however, we require that the four terms of both lexons involved be articulated. Indeed, how can one imply that an instance playing a particular role "r1" implies that same instance playing another role "r2" if the terms or the relation itself are not specified. Remember that lexons can be articulated as well only if both its terms are defined.

The social processes for this phase are:

- Request to create a constraint
- Request to remove a constraint
- Request to change superlexon of lexon (corresponds with role-hierarchies and ORM subset-constraints).

4.5 Committing to the Hybrid Ontology

Once there is a close approximation of a (part of) the hybrid ontology for meeting the SIRs, the stakeholders can start annotating their information systems, with the hybrid ontology by means of a commitment. The commitments enable the exchange of information residing in those systems. With every (closer) approximation of the domain with the hybrid ontology, the commitments will provide access to instances of concepts that can be used for defining and/or refining the definitions, fact types and constraints in the hybrid ontology description.

Ω-RIDL [71] is the application commitment language we have adopted. It was extended to include references to community commitments. Take for example the ER-diagram for a fictitious database storing information about artists and works of art in Fig. 7. The corresponding application commitment is shown below the diagram. Notice the reference to the cultural domain community, which will include all lexons and constraints currently agreed upon by that community. This particular commitment furthermore includes some application specific knowledge to annotate the artificial IDs. The commitment describes how these IDs uniquely and totally identify instances of artists and works of art. Furthermore the terms "Artist" and "Work Of Art" inside the application's lexons are declared to be synonymous with that of the community. The lexons and constraints of the cultural domain community in this example were assumed to include (where 'C' stands for "Cultural Domain Community"):

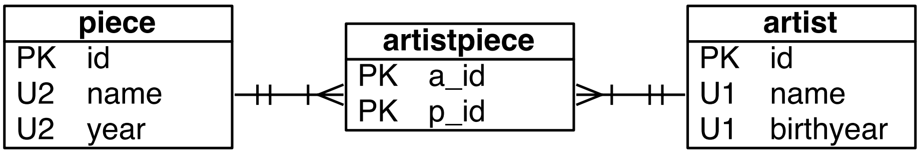
```
<C,Art Movement,with,of,Name>
<C,Artist,with,of,Art Movement>
<C,Artist,born in,of birth of,Year>
<C,Work Of Art,with,of,Title>
<C,Work Of Art,made in,of,Year>
<C,Artist,with,of,Gender>
<C,Artist,contributed to,with contributor,
   Work Of Art>
<C,Gender,with,of,Code>
<C,Artist,having,of,Name>
EACH Name IS LEXICAL.
EACH Code IS LEXICAL.
EACH Year IS LEXICAL.
EACH Title IS LEXICAL.
```

The lexical constraints limit instances of concepts denoted by a term to "things" that can be printed on a screen.

4.6 Community and SIR Co-evolution

We explained how a community starts the development of a hybrid ontology by first defining their SIRs, articulate the key terms in those requirements and gradually construct agreements on fact types, glosses, constraints, gloss-equivalences and synonyms. Communities and their SIRs are, however, not static. They are evolving and even co-evolving. With the additional of a new stakeholder in the community, the community changed not only with the presence of a new member, but also with the addition of new ideas, a possible different perspective on matters and possible new requirements for the community. Also requirements can change from external forces, e.g. due to legislation changes. The community constitution does not necessarily need to change for the SIRs to evolve; a community can come to the conclusion that the current approximation of the domain by the hybrid ontology description does not

[8] Which is only true for only certain types of applications.



```
BEGIN SELECTION
 # Selection of the community.
[’Cultural Domain Community’]
 # Application specific lexons
 <’MyOrganization’, Artist, with, of, AID>
 <’MyOrganization’, Work Of Art, with, of, WID>
END SELECTION
BEGIN CONSTRAINTS
 # Declaration of synonyms
 LINK(’Cultural Domain Community’, Artist, ’MyOrganization’, Artist).
 LINK(’Cultural Domain Community’, Work Of Art, ’MyOrganization’, Work Of Art).
 # List application specific constraints
 EACH Artist with AT MOST 1 AID.        EACH Artist with AT LEAST 1 AID.
 EACH AID of AT MOST 1 Artist.          EACH Work Of Art with AT MOST 1 WID.
 EACH Work Of Art with AT LEAST 1 WID.  EACH WID of AT MOST 1 Work Of Art.
END CONSTRAINTS
BEGIN MAPPINGS
 # Mapping of application symbols, in this case from Table X Field -> Term role Term (role Term)+
 MAP ’Artist’.’name’ ON Name of Artist.
 MAP ’Artist’.’birthyear’ ON Year of birth of Artist.
 MAP ’Artist’.’id’ ON AID of Artist.
 MAP ’piece’.’name’ ON Title of Work Of Art.
 MAP ’piece’.’year’ ON Year of Work Of Art.
 MAP ’piece’.’id’ ON WID of Work Of Art.
 MAP ’artistpiece’.’a_id’ ON  AID of Artist contributed to Work Of Art.
 MAP ’artistpiece’.’p_id’ ON WID of Work Of Art with contributor Artist.
END MAPPINGS
```

Fig. 7 Example ER diagram and corresponding Ω-RIDL application commitment.

meet their needs even though it complied with the requirements. In that case, the community will negotiate changes to the requirements. This can happen when the community starts to better understand the domain.

5 Glossary Evolution

Now that we have presented the method, we will describe how the evolution of glosses impacts the hybrid ontology. We follow Jarrar that the purpose of a gloss is not to provide or catalogue general information and concepts about a concept, as conventional dictionaries and encyclopedias do, but is supposed to render factual knowledge that is critical to understanding a concept in ontology engineering [34].

A gloss is composed of one ore more sentences constructed with the community's usual alphabet. Those sentences have to be themselves human-interpretable in order for the gloss to become understandable. We denote $\mathcal{S}$ as the set of all possible sentences that can be constructed with those alphabets. We ignore whether this set contains sentences (or parts thereof) that are valid syntax- and grammar wise. As the community will choose and discuss the elements of this set used for constructing a gloss, they will make sure that what is chosen makes sense (at least for this community). We will thus avoid talking about truth, since we are not in a formal logical context. We exclusively use this word between an ontology and semantics. Occasionally and carefully, we will use this word for addressing the agreement on validity assumed to exist in the community. In other words, truth is relative to the community; if there is an agreement, it is assumed to be valid.

Every part of a gloss should contribute to a better understanding of the concept described. As a consequence, some of these parts should correspond with (parts of) the formal description of that concept. In other words, as the glosses evolve, so should the lexons and constraints. We will describe how glosses can evolve, and their impact on commitments.

Glosses evolve for a reason, which is captured by the motivation of the change and the communities' discussion. How the gloss changes, can be formalized. There are two types of gloss updates. A first is a complete change of a gloss. We deem this kind of update to happen only accidentally; as such a change would imply that the community – as a whole – misinterpreted the term being described in the context of that community (and their goals).

The second type of change is a (more gradual) refinement of the gloss. Glosses are composed of one or more sentences. Sentences or parts of sentences can be added or removed. The sentences that have been added or removed serve (or served in the case of the latter) a particular *purpose* for that gloss. That purpose captures and describes how the gloss and those sentences were related. On can define many such purposes. In this paper, the set of purposes is referred to by Θ.

Definition 6 (Gloss evolution) Within a community $\gamma \in \Gamma$, gloss evolution is defined as mappings of the form $\Theta_i^\gamma : 2^{\mathcal{S}} \rightarrow 2^{\mathcal{S}}$, where a glossary $Gloss \rightarrow \Theta_i^\gamma(Gloss) = Gloss'$ by means of the application of a purpose element in Θ. This creates a discrete gloss evolution. Gloss evolution is a mapping $\epsilon : \Gamma \times \mathbb{N} \rightarrow 2^{\mathcal{S}}$, in which $(\gamma, t) \rightarrow Gloss(t)$ where t is a point in time, such that $\epsilon(\gamma, 0) = \emptyset$ and $\exists g \in Gloss, \exists s \in \mathcal{S} : Gloss(t) = \{Gloss(t-1) \setminus \{g\}\} \cup \{\Theta_i^\gamma(g, s)\}$.

The *linguistic amalgamation operators* to add or remove (a part of a) sentence from a gloss is defined as: $\oplus : Gloss \times \Theta \times \mathcal{S} \rightarrow Gloss$ for adding an element of $\mathcal{S}$ to a gloss and $\ominus : Gloss \times \Theta \times \mathcal{S} \rightarrow Gloss$ for removing an element of $\mathcal{S}$ from a gloss.

For simplicity's sake, the exact places where sentences would be added or which occurrence of a part is removed are "ignored". Those are assumed to be additional parameters of above-mentioned operators. The changes in text necessary for it to be proper to the communities' language, such as readjusting capitalization, are also "ignored" for the same reason.

5.1 The Elements of Θ

This Section will provide a proposal for the elements in Θ. To this end, inspiration is drawn from discourse theory, and more concretely Rhetorical Structure Theory [49]. Elements from RST are furthermore refined and complemented with elements introduced specific for GOSPL. One such element specific to GOSPL is gloss-adoption, where one community explicitly states to adopt the gloss of another community.

The goal is not to provide an exhaustive list, as it can vary depending on the type of community or even the type of language they employ. The framework is therefore defined in such a way that new elements can be introduced to Θ.

A distinction is made between elements that affect the community commitment and elements that merely serve to provide additional text to (better) understand the concept described.

The application of some gloss-amalgamation operators might imply the introduction of lexons, constraints and even instances in the community commitment. Lexons and constraints are useful for populating the lexon base and community commitments. The instances to validate the constraints explicitly agreed upon by the community.

5.1.1 Drawing Inspiration from RST

For the elements of Θ, inspiration is – as already stated above – first drawn from RST. RST was originally developed as part of research on computer-based text generation. RST was intended to describe texts by means of two types of "relations", each at a different level. The first is the "nucleus-satellite relation" and is the most frequent structural pattern. It captures that two spans of text (usually adjacent) are related such that one of them has a specific role relative to the other. The other type is "multinuclear relations", grouping a set of nuclei. RST thus offers means to annotate the role (the purpose) that a part in the text plays on another part.

Before moving on, the reader needs to be aware that in RST, the purposes that spans serve to other spans are somewhat confusingly labeled "relations". A term obviously not suitable for a computer science text. The authors of RST are linguists and the "relations" they proposed actually refer to "functionalities", a purpose. From here onward, the use of the word "relation" in the context of RST will be avoided.

RST thus allows one to describe how two segments of discourse are connected to one another. With elements of the first type, the nucleus (N) is part the of the text onto which the satellite (S) will play a particular role.

Example 4 For instance, in the sentence: *"Employees are urged to complete new beneficiary designation forms for retirement or life insurance benefits whenever there is a change in marital or family status"* The part *"whenever there is a change in marital or family status"* is the satellite and expresses a *condition* for *"Employees are urged to complete new beneficiary designation forms for retirement or life insurance benefits"*, the nucleus.

Glosses need to briefly describe terms or lexons employed by the community. They also need to be agreed upon by that same community. Opinions or statements in favor of a particular gloss are part of the discussion leading to an agreement, and not part of a gloss. As RST provides "relations" with a subjective nature (e.g., the *antithesis* that describes ideas favored by the author), only a subset of these "relations" that is deemed relevant for gloss evolution will be presented, together with examples. This is in line with the guidelines on the construction of glosses given in [34] mentioned in Section 3. But first, the purposes provided by RST that were not taken into consideration are:

1. Purposes for expressing opinions: antithesis, concession, and justify.
2. Purposes aimed at enabling the reader in undertaking actions: enablement.
3. Purposes at interpreting and evaluating text: interpretation and evaluation.
4. Purposes concerned with relating information with causes and effects: non-volitional cause/result, volitional cause/result.

The purposes involving a nucleus and a satellite we adopted are:

- With **background**, S is used to facilitate the understanding of N.
- **Circumstance** is used to denote that S sets the framework for interpreting N.
- A **condition** is used to state that the truth-value accorded to the proposition in N depends on the truth-value accorded to the proposition in S. Related functions are: **Unconditional**, **Otherwise**, **Unless**.
- **Elaboration** denotes the addition of information to already available information. There are several specializations of elaboration. Examples are: Specialization, Part-whole, etc.
- **Evidence** is a piece of information that supports a claim, in this case, the gloss. Examples are typically used as evidence; they support the definition contained in the gloss. Examples, however, are already present as a special case of the elaboration function (instantiation). We therefore limit the type of evidence to information supporting a claim.
- **Means** is used to denote S presenting an instrument used for achieving the concept described in N.
- In **preparation**, S is used to prepare the reader to expect and interpret N.
- **Purpose** is used to describe that an activity in N needs to be initiated in order to achieve what is described in S. In other words, S thus describes the purpose for doing the activity described in N.
- In RST, "solutionhood" describes the function that N presents a solution to the problem described in S. As this function is one of the few where the description of the function is more intuitive from N to S and all other functions presented here described the function of S on N, we choose to rename this function into **problem-for**. This way, the meaning of each function is presenting a role S is playing on N.

The elements of Θ currently described involved a satellite playing a role on a nucleus. However, RST also described other "relations" which involves multiple nuclei. All but two of this type are taken into account. The multinuclear restatement was not considered for the same reason as the restatement purpose. Also the joint purpose was not considered, as it is used to "glue" two pieces of text that are not related. As a gloss needs to present a brief description of the described term, all parts in that gloss need to be relevant.

The purposes involving multiple nuclei used in this paper are:

- **Conjunction**. The items are conjoined to form a unit in which each item plays a comparable role. Items can be combined with words such as "and" and "nor".
- **Disjunction**. An item presents an alternative for the other(s). The disjunction is not necessarily an exclusive disjunction.
- **Contrast** is used for at most two nuclei. The two are understood to be similar (or the same) in many respects and to differ in a few respects, and both are compared with respect to those differences.
- **List** for linking items are comparable to each other and **sequence** for linking items with a logical succession, e.g., steps to perform a task.

5.1.2 Remaining Gloss Evolution purposes

RST provides a foundation for choosing gloss-evolution purposes. The hybrid ontology-engineering framework we adopted, however, also provides processes that result in gloss evolution not related with these purposes.

Given two communities $\gamma_1, \gamma_2 \in \Gamma$ and their respective terms $t_1, t_2 \in T$ and γ_2 has articulated t_2 with a gloss g. Community γ_1 is able to adopt g for describing t_1. It is obvious that this operation evolves the gloss for $\langle \gamma_1, t_1 \rangle$. The implications of the adopted gloss on the hybrid ontology remain within the original community, but are known to the adopting community as this operator links both terms.

Also important is the identification of the set of attributes that uniquely and totally identify instances of a concept. To this end, a special kind of attribute-purpose is introduced: the **identifies**. With this purpose, one will later on be able to identify these attributes and distill the necessary constraints for it.

5.2 Glossary, Lexon Base & Commitment Co-evolution

The elements in Θ and the amalgamation operators enable the support of the co-evolution of communities, ontologies and glossaries. Changes in the requirements of the community are reflected in the formal part of the ontology and possibly require the refinement of the glosses based on the newly defined gloss evolution operators. In turn, those changes might start a series of social processes for the formal part of the hybrid ontology to reflect those changes accordingly. At any time, changes in both the lexon base and the glossary will influence the communities' next decisions. Some elements of Θ provide only additional information to the community for understanding the gloss. Other elements, however, can and *should* have an impact on the hybrid ontology. These elements influence the hybrid ontology at three levels:

1. The introduction of one or more pre-lexons in the lexon base.
2. The introduction of pre-constraints.
3. The introduction elements in the population of a term or a lexon.

Pre-lexons are "raw" lexons that have not yet been refined by the community (e.g., proper stemming of verbs in roles, the introduction of the co-role, etc.). Some gloss-evolution purposes result in lexons of which roles, concepts or generalization of concepts are known. For instance, when describing a specialization of a concept, the roles *is a / subsumes* – interpreted as the taxonomic relation – will be proposed. Pre-constraints are constraints in terms of the pre-lexons.

Example 5 Given some community γ wishing to articulate the term "Car" with a gloss. The application of the generalization (a type of elaboration): *"A car"*$\oplus$*Generalization("is a road vehicle")* results in the following pre-lexon.

- $\langle\gamma,\mathit{Car},\underline{\mathit{is\ a},\mathit{subsumes}},\mathit{road\ vehicle}\rangle$

The roles are underlined as they are pre-filled and have a special interpretation. Nothing prevents community members to refine this pre-lexon and change its role-labels. But as the gloss has evolved with a generalization, one would expect that this would reflect with the addition of a taxonomic relation in the hybrid ontology.

Once refined, the execution of this gloss evolution triggers social processes for adding this lexon.

Example 6 Taking the following gloss g for "Car" in some community $\gamma \in \Gamma$: *"A car is a road vehicle"*. One can elaborate on this term by adding the following sentence s *"powered by an internal combustion engine and able to carry a small number of people."*. The sentence s is actually the result of a conjunction $s' \oplus \mathit{Conjunction}(s'')$ where s' = *"powered by an internal combustion engine"* and s'' = *"and able to carry a small number of people."* Elaborating g with the conjunction contained in s with $g \oplus \mathit{Elaboration}(s)$ results in the following pre-lexons:

- $\langle\gamma,\mathit{Car},\mathit{powered\ by},.,\mathit{an\ internal\ combustion\ engine}\rangle$
- $\langle\gamma,\mathit{Car},\mathit{able\ to\ carry},.,\mathit{a\ small\ number\ of\ people}\rangle$

These pre-lexons have to be refined by the community and social processes are started to include these lexons in the hybrid ontology.

- $\langle\gamma,\mathit{Car},\mathit{powered\ by},\mathit{powers},\mathit{Internal\ Combustion\ Engine}\rangle$
- $\langle\gamma,\mathit{Car},\mathit{carrying},\mathit{carried\ by},\mathit{Group}\rangle$

Instances can be distilled from glosses and used as a test population in hybrid ontology engineering.

Example 7 Given a gloss g = *"A planet, in astronomy, is one of a class of celestial bodies that orbit stars."* for the term "planet". One can elaborate this gloss by giving examples. s = *"Examples are Mercury, Mars and Earth."* $g \oplus \mathit{Instantiation}(s)$ = *"A planet, in astronomy, is one of a class of celestial bodies that orbit stars. Examples are Mercury, Mars and Earth."* The instances of planets are proposed to be taken into account, and will – once accepted – serve as a test population for ontology engineering. $\mathit{Population}(\gamma, \mathit{Planet}) = \{\mathit{Mercury}, \mathit{Mars}, \mathit{Earth}\}$

There are several ways to discover constraints in glosses. For example, the conditional can express subset constraints between roles of lexons. Others, such as the elaboration, contain hints on frequency or totality constraints within a pre-lexon.

Example 8 Given a gloss g = *"A proposal results in a project"* for the term "Proposal" in a certain community $\gamma \in \Gamma$ and assuming that the lexon $\langle\gamma$, *Proposal*, *results in*, *result of*, *Project*$\rangle$ is already present. By adding a condition by applying $g \oplus$ *Condition("when the proposal is accepted by the review board.")*, the following pre-lexon and subset constraint are distilled:

- $\langle\gamma,\mathit{Proposal},\mathit{is\ accepted\ by},.,\mathit{review\ board}\rangle$
- Subset constraint from *"is accepted by"* to *"results in"*.

The community can refine the pre-lexon as well as the subset constraint and trigger social processes to accept these in the formal part of the hybrid ontology.

The extraction of possible lexons, constraints and instances for the community commitment can be to some extent automated by applying natural language processing (NLP) techniques.

While extracting the pre-lexons for a purpose f_1, the detection of terms in the nucleus will often follow the same steps. In essence, if the nucleus is a sentence, then noun-extraction will be applied on this sentence. However, if the satellite is a purpose f_2, then the nucleus of that purpose should be examined. If the purpose f_1 would be applicable to the nucleus of this purpose f_2, then the structure of this gloss would look different. This becomes clear if the application of elements of Θ for glosses are visualized as a tree (see Fig. 8). Above, the wheels are part of the car, which is the nucleus of the purpose. Below, however, the meronymic relation is pointing to the road vehicle. It is up to the community to ensure that the structure of the gloss corresponds makes sense. When the gloss is defined by means of prepositions, those prepositions can easily be structured according to these rules. In case of a multi-nucleic relationship, the terms will be detected in each element of that relationship. The multi-nucleic relationships are mainly exploited to distill a series of pre-lexons that play a particular purpose. For instance, in the case of an identification purpose, the disjunction will denote distinct reference structures, whereas the conjunction will indicate which attributes belong together.

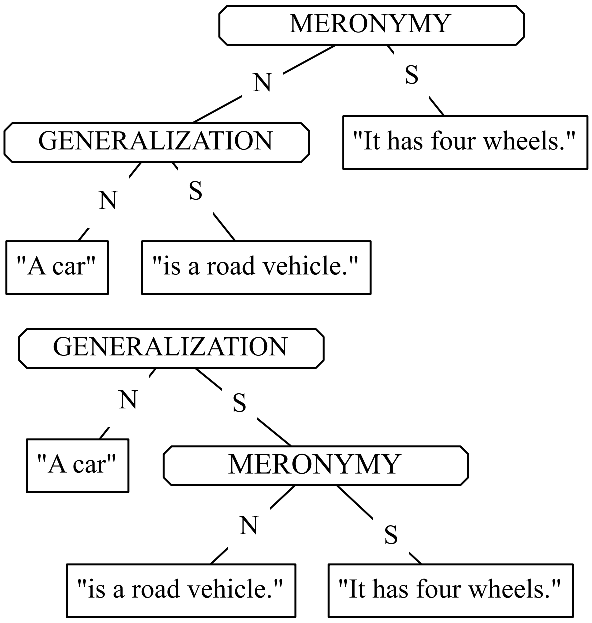


Fig. 8 Tree representation of differently structured glosses that appear the same.

6 Application commitments in the Feedback Loop

Application commitments provide valuable information about which terms and lexons the different members of the community representing their organization commit to. This selection is exploited by informing those members when changes are requested (and occur) in the ontology as to stimulate discussion.

The mapping α in those commitments is furthermore used to delve into the annotated data in search for support or counterexamples for certain statements made by the community, e.g. to notify the community whether proposed constraint is true for all annotated information systems currently known in the community. This process will guide the community in its dialogue to achieve agreement. This is done by generating the necessary queries using the commitments of each of the applications, populating the lexons in the conceptual schema and then reason over the data in terms of lexon populations. This tool is called Ω-DIPPER Fig. 9 extends and depicts the place of Ω-DIPPER in the feedback loop.

6.1 Semantics of Constrained Lexons

We presented in Section 3 the encoding of lexons and constraints in Description Logics (DLs) so that we can utilize DL-reasoners for reasoning tasks over the resulting ontologies. The Open World Assumption (OWA) in DLs allows the existence of unknown information. However, in many cases, we want information to be as complete as possible to support business - as defined by the semantic interoperability requirements of a community of stakeholders. In other words, the instances in the annotated information systems must follow certain business rules to ensure proper business. For example, in some cultural domain, we want that every “Event must have at least one explicit associated Location”. The OWA, however, cannot capture our intuition of constraints that need to be imposed, thus we assign different semantics for constraint lexons to treat them as OWL integrity constraints (ICs) [54,65].

Our purpose is twofold: (1) using ontologies to provide shared conceptualization and to enrich data, and (2) using constraints as integrity constraints for data inside each application. To achieve this, we combine the Open World Assumption and Closed World Assumption (CWA). OWA is used in reasoning to derive new knowledge, and CWA is adopted when validating the integrity of the application data. We present a survey of existing approaches for OWL integrity constraints. Note that some authors refer to an ontology as a knowledge base. This knowledge base is denoted as a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where TBox $\mathcal{T}$ consists of terminological axioms and ABox $\mathcal{A}$ consists of assertions or data sets. For the hybrid ontology engineering framework, we agree that [30] that in an ontology, the description of concepts and relations should be separated from its instances. For simplicity's sake, however, we consider both the axioms and assertions when discussing the integrity constraints in this Section. And we use a set of assertions as an abbreviation of a model.

Definition 7 (ICs by consistency [42]) An ontology $\mathcal{O}$ satisfies an integrity constraint IC if and only if $\mathcal{O} \cup IC$ is satisfiable.

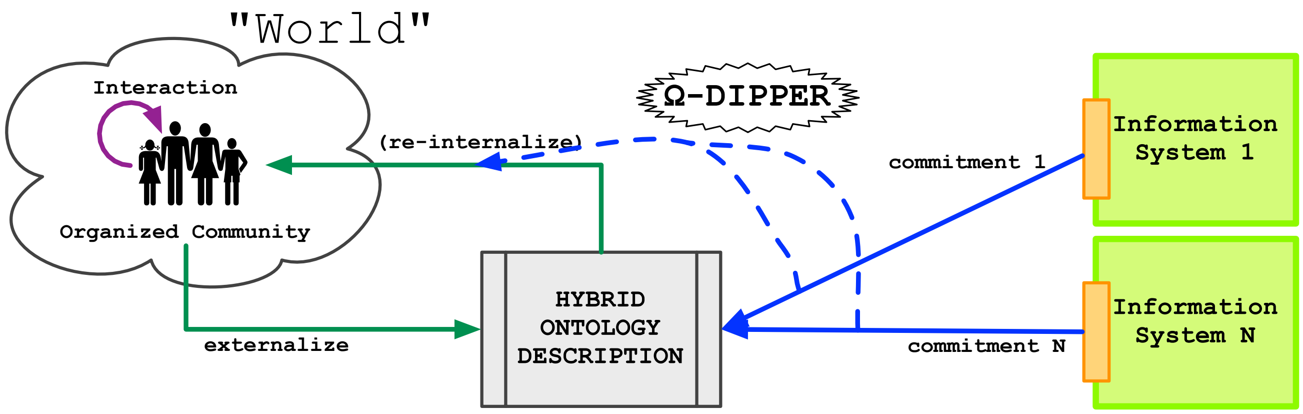


Fig. 9 Feedback loop from the ontologies to the community by not merely taking into account the lexons committed to by the application, but the data in the annotated organization information systems as well.

Example 9 Suppose that $\mathcal{O}_1$ consists of the following axioms

$$\exists has^{-}.\top \sqsubseteq Location \tag{14}$$

$$MusicEvent \sqsubseteq Event \tag{15}$$

$$MusicEvent(boomtown) \tag{16}$$

and IC_1 contains only

$$Event \sqsubseteq \exists has.\top \tag{17}$$

It is easy to see that, under OWA, $\mathcal{O}_1 \cup IC_1$ is satisfiable. So $\mathcal{O}_1$ satisfies IC_1 by Definition 7. However, it does not fit our intention of using the constraint to ensure that every event has explicit locations; *boomtown* is an event but its location is not explicitly presented.

Definition 8 (ICs by entailment [60]) An ontology $\mathcal{O}$ satisfies an integrity constraint IC if and only if $\mathcal{O} \models IC$.

Example 10 We consider another example in which IC_2 only contains (17). Ontology $\mathcal{O}_2$ consists of all axioms of $\mathcal{O}_1$ together with $\{has(boomtown, ghent), Location(ghent)\}$. Intuitively, one might think that ontology $\mathcal{O}_2$ satisfies IC_2. However, there is also a model $\mathcal{I}_1 = \{$*MusicEvent(boomtown), Event(boomtown), Event(polepole), has(boomtown, ghent), Location(ghent)*$\}$ of $\mathcal{O}_2$ for which $\mathcal{I}_1 \not\models IC_2$. By Definition 8, $\mathcal{O}_2$ does not satisfy IC_2. That contradicts the intuition.

Definition 8 states that all models must entail the integrity constraints. However, the example above suggests that entailment in this Definition should be restricted to minimal models of $\mathcal{O}$. $\mathcal{I}$ is a *minimal model* of $\mathcal{O}$ if and only if $\mathcal{I}$ is a model of $\mathcal{O}$ and there is no model $\mathcal{J}$ of $\mathcal{O}$ such that $\mathcal{J} \subset \mathcal{I}$. Therefore, Definition 8 should be formalized as follows: $\mathcal{O}$ satisfies IC if and only if all minimal models of $\mathcal{O}$ entails IC. This idea has been nicely captured in [54, 53], where ontology axioms are expressed as FOL formulas [5] and *skolemization* [55] is applied to deal with existential quantifiers.

Definition 9 (ICs by minimal models & skolemization [54, 53]) Let $\pi(\mathcal{O})$ and $\pi(IC)$ be FOL formulas that express axioms in $\mathcal{O}$ and IC respectively, $\mathsf{sk}(\mathcal{O})$ be the set of formulas obtained by skolemization of $\pi(\mathcal{O})$. $\mathcal{O}$ satisfies integrity constraint IC if and only if $\mathcal{I} \models \pi(IC)$ for every minimal Herbrand model $\mathcal{I}$ of $\mathsf{sk}(\mathcal{O})$. We shall sometimes write $\mathcal{I} \models IC$ instead of $\mathcal{I} \models \pi(IC)$.

Definition 10 (Herbrand-based model) Given an ontology $\mathcal{O} = \{\mathcal{T}, \mathcal{A}\}$.$\mathcal{I}$ is a *Herbrand-based model* of $\mathcal{O}$ if $\mathcal{I}$ is a model of $\mathcal{O}$, $\Delta^{\mathcal{I}}$ consists of ABox's individuals, and every individual is mapped to itself. A Herbrand-based model $\mathcal{I}$ of an ontology $\mathcal{O}$ is *minimal* if there is no Herbrand-based model $\mathcal{J}$ of $\mathcal{O}$ such that: $\Delta^{\mathcal{I}} \subset \Delta^{\mathcal{J}}$ and interpretation function $\cdot^{\mathcal{J}}$ contains every mapping in the interpretation function $\cdot^{\mathcal{I}}$.

Now we reconsider examples 9 and 10 with respect to Definition 9. In Example 9, the only minimal Herbrand model of $\mathcal{O}_1$ is $\mathcal{I}'_1 = \{MusicEvent(boomtown), Event(boomtown)\}$. By Definition 9, $\mathcal{O}_1$ does not satisfy IC_1 because $\mathcal{I}'_1 \not\models IC_1$. It follows the intuitive interpretation of avoiding an unknown *Location* for *Event*. In example 10, the only minimal Herbrand model of $\mathcal{O}_2$ is $\mathcal{I}_2 = \mathcal{I}_1 \setminus \{Event(polepole)\}$ and $\mathcal{I}_2 \models IC_2$, then $\mathcal{O}_2$ satisfies IC_2. This also fits the intuitive interpretation, however in some cases the skolemization could lead to unexpected consequences.

Example 11 Let $\mathcal{O}_3$ consists of following axioms:

$$\exists has^{-}.\top \sqsubseteq Location \tag{18}$$

$$MusicEvent \sqsubseteq Event \tag{19}$$

$$Event \sqsubseteq \exists has.\top \tag{20}$$

$$MusicEvent(boomtown) \tag{21}$$

IC_3 contains only the axiom $MusicEvent \sqsubseteq \exists has.\top$

The minimal Herbrand model of ontology $\mathcal{O}_3$ is of the form $\mathcal{I}_3 = \{$*MusicEvent(boomtown), Event(boomtown), Location(u), hasLocation(boomtown, u)*$\}$ where u is generated by skolemization of axiom (18) and (20). We see that $\mathcal{I}_3 \models$

IC_3, so $\mathcal{O}_3$ satisfies the IC_3 although the exact location of *boomtown* is unknown.

Definition 9 almost captures the intuition of ICs, but it has some drawbacks discussed in [65]. To avoid the problem of unknown individual and avoid unintuitive meaning of integrity constraints, we use an alternative semantics for OWL integrity constraints. Before that, however, the *Herbrand-based model* of an DL ontology is first defined, which is similar to the Herbrand model in FOL.

Now the *integrity constraint interpretation* is defined to interpret the translated GOSPL constraints. For each ontology, there is at most one such interpretation. In case of unsatisfiable ontologies, there is no such interpretation.

Definition 11 (Constraint interpretation) Given an ontology $\mathcal{O} = \{\mathcal{T}, \mathcal{A}\}$ and $\mathcal{M} = \{\mathcal{I}_1, \ldots, \mathcal{I}_n\}$ is a set of its minimal Herbrand-based models. Concept name A, role R, and individual d in integrity constraints are interpreted by the following *integrity constraint interpretation* $\mathcal{I}_{IC} = \{\Delta^{\mathcal{I}_{IC}}, \cdot^{\mathcal{I}_{IC}}\}$, where $\Delta^{\mathcal{I}_{IC}}$ is a set of all individuals in $\mathcal{A}$, as follows:

$$\begin{aligned}
A^{\mathcal{I}_{IC}} &= \{d^{\mathcal{I}_{IC}} \mid d^{\mathcal{J}} \in A^{\mathcal{J}}, \text{for all } \mathcal{J} \in \mathcal{M}\} \\
R^{\mathcal{I}_{IC}} &= \{(c^{\mathcal{I}_{IC}}, d^{\mathcal{I}_{IC}}) \mid (c^{\mathcal{J}}, d^{\mathcal{J}}) \in R^{\mathcal{J}}, \text{for all } \mathcal{J} \in \mathcal{M}\} \\
d^{\mathcal{I}_{IC}} &= d
\end{aligned}$$

The extension of $\mathcal{I}_{IC}$ to inverse roles and complex concepts is done as normal.

We will use *integrity constraint interpretation* to define how an ontology satisfies an integrity constraint. Note that we only consider satisfiable ontologies whose *integrity constraint interpretation* exists.

Definition 12 (Integrity constraint satisfaction) Given an ontology $\mathcal{O}$ and its *integrity constraint interpretation* $\mathcal{I}_{IC}$, $\mathcal{O}$ satisfies an integrity constraint IC if and only if $\mathcal{I}_{IC} \models IC$.

We reconsider the previous examples and check whether Definition 12 captures our intuition of integrity constraints.

Example 12 Reconsidering Example 9 with respect to Definition 12. The only minimal Herbrand-based model of $\mathcal{O}_1$ is of the form $\mathcal{I} = \{$*MusicEvent(boomtown), Event(boomtown)*$\}$. We have $Event^{\mathcal{I}_{IC}} = \{boomtown\}$ but $(\exists has.\top)^{\mathcal{I}_{IC}} = \emptyset$. Thus, $\mathcal{O}_1$ does not satisfy IC_1. This matches the intuition of the constraint.

It is easy to check that Definition 12 also matches the intuition in Example 10. Now, we reconsider Example 11 in the light of Definition 12.

Example 13 Every minimal Herbrand-based model of $\mathcal{O}_3$ is of the form $\mathcal{I}_i = \{$*MusicEvent(boomtown), has(boomtown,* u_i*), Event(boomtown), Location(*u_i*)*$\}$ Where u_i is different in each $\mathcal{I}_i$. We have $MusicEvent^{\mathcal{I}_{IC}} = \{boomtown\}$, but $(\exists has.\top)^{\mathcal{I}_{IC}} = \emptyset$. Thus $\mathcal{O}_3$ does not satisfy IC_3 as expected.

In the next part, we briefly present the process of checking integrity constraints.

6.2 Checking Constraints over Annotated Data

Our proposal resembles the *IC-interpretation* in [65]. We differ from [65] in using minimal Herbrand-based models instead of classical models. To validate integrity constraints, we follow the approach in [65]. However, we adopt the Unique Name Assumption instead of the Weak Unique Name Assumption [65].

We have implemented a prototype supporting the common ORM constraints [29]: *mandatory constraint*, *internal uniqueness constraint*, and *external uniqueness constraint*. Some SPARQL queries to check the validity of those integrity constraints and get the counter examples are presented below. To get complete answers, we use HermiT[9] reasoner to classify the ontology and then perform *materialization* before running those SPARQL queries. Note that our method works correctly with ontology languages in which classification task can be done without taking assertions into account.

Mandatory constraint $C \sqsubseteq \exists R.\top$

```
PREFIX ont: <http://path.to.my.ontology/#>
SELECT ?x WHERE {
    ?x a ont:C.
    OPTIONAL {?x ont:R ?y.}
    FILTER (!BOUND(?y))
}
```

Internal uniqueness constraint $(funct\ R)$

```
PREFIX ont: <http://path.to.my.ontology/#>
SELECT ?x WHERE {
    ?x ont:R ?y1.
    ?x ont:R ?y2.
    FILTER (?y1 != ?y2)
}
```

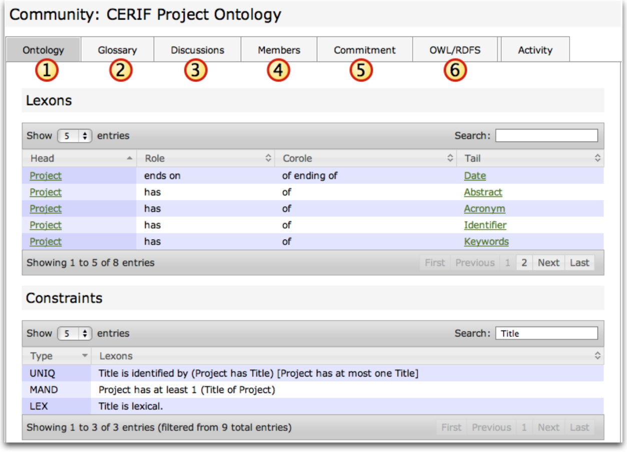
External uniqueness constraint $(id\ C\ R_1\ ...\ R_n)$

```
PREFIX ont: <http://path.to.my.ontology/#>
SELECT ?x1 ?x2 WHERE {
    ?x1 a ont:C.
    ?x1 ont:R1 ?y1.
    ...
    ?x1 ont:Rn ?yn.
    ?x2 a ont:C.
    ?x2 ont:R1 ?y1.
    ...
    ?x2 ont:Rn ?yn.
    FILTER (?x1 != ?x2).
}
```

[9] http://www.hermit-reasoner.com

7 GOSPL: The Tool

The tool is developed in Java and runs inside an application container such as JBoss. It contains two layers: the base layer contains all the domain classes and communication with the server and a web application providing the interface layer. The base layer can also be consulted by other software agents, making the development of standalone clients possible. Fig. 10 shows a screenshot of the GOSPL tool. It shows a screen with the lexons and constraints of one communities' hybrid ontology description (1) and glossary (2), links to the discussions (3), community management (4), the commitments of applications to the ontology (5) and the OWL implementation of the hybrid ontology (6).



Lexons

Head	Role	Corole	Tail
Project	ends on	of ending of	Date
Project	has	of	Abstract
Project	has	of	Acronym
Project	has	of	Identifier
Project	has	of	Keywords

Showing 1 to 5 of 8 entries

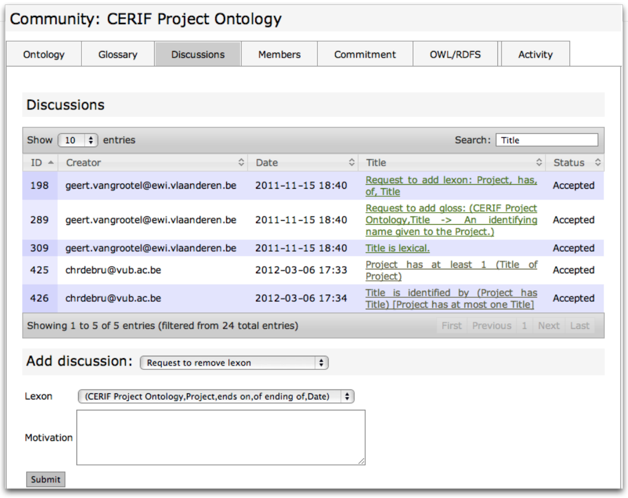
Constraints

Type	Lexons
UNIQ	Title is identified by (Project has Title) [Project has at most one Title]
MAND	Project has at least 1 (Title of Project)
LEX	Title is lexical.

Showing 1 to 3 of 3 entries (filtered from 9 total entries)

Fig. 10 Screenshot of the lexons and constraints of one communities' hybrid ontology description.

GOSPL is discussion-oriented and both the ontology and glossary evolve only if the community reaches an agreement[10]. This results in traceability not only at change level, but also on decision level. In Fig. 11, several discussions are shown. Different discussions can be started, one for each of the social processes defined in Section 4. Depending whether a person is a member of the community, some discussions might not be available. However, all users can leave comments and all users can start "informal" discussions (even when they are not part of the community). In other words, we not only record who changes what, but also the reasons certain changes have been made by linking changes to discussion on the platform. This was possible by formalizing the social processes and its corresponding operators.



Discussions

ID	Creator	Date	Title	Status
198	geert.vangrootel@ewi.vlaanderen.be	2011-11-15 18:40	Request to add lexon: Project, has, of, Title	Accepted
289	geert.vangrootel@ewi.vlaanderen.be	2011-11-15 18:40	Request to add gloss: (CERIF Project Ontology,Title -> An identifying name given to the Project.)	Accepted
309	geert.vangrootel@ewi.vlaanderen.be	2011-11-15 18:40	Title is lexical.	Accepted
425	chrdebru@vub.ac.be	2012-03-06 17:33	Project has at least 1 (Title of Project)	Accepted
426	chrdebru@vub.ac.be	2012-03-06 17:34	Title is identified by (Project has Title) [Project has at most one Title]	Accepted

Showing 1 to 5 of 5 entries (filtered from 24 total entries)

Add discussion: Request to remove lexon

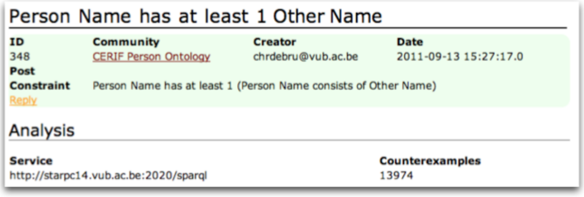
Lexon (CERIF Project Ontology,Project,ends on,of ending of,Date)

Motivation

Fig. 11 Discussions (social processes) in GOSPL.

A voting system is used to gather the opinion of people without the need of participating in the discussion.

The application commitments belonging to community members describe how the application symbols of their system commit to the ontology, allowing the information in those database systems to be retrieved through the ontology. Of course, the discovery of counterexamples does not necessarily mean that the statement is false, however, this information might direct the discussion into another direction. Fig. 12 shows a dataset has over 13000 counterexamples for the mandatory constraint on "has" between "Person" and "Other Name".



Person Name has at least 1 Other Name

ID	Community	Creator	Date
348	CERIF Person Ontology	chrdebru@vub.ac.be	2011-09-13 15:27:17.0

Post

Constraint Person Name has at least 1 (Person Name consists of Other Name)

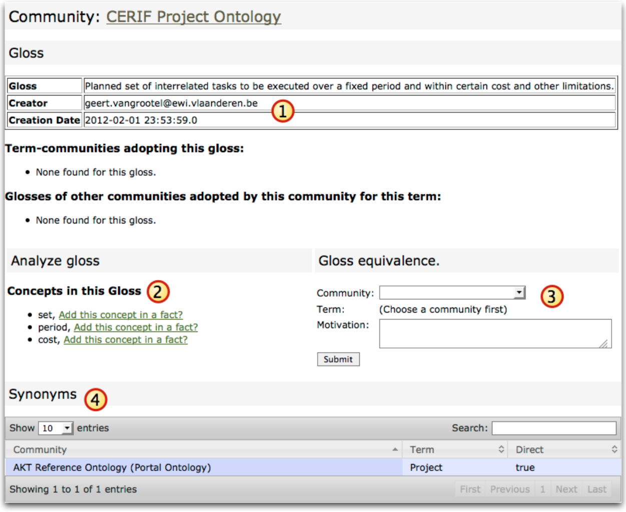
Analysis

Service	Counterexamples
http://starpc14.vub.ac.be:2020/sparql	13974

Fig. 12 Finding counterexamples for statements.

Fig. 13 depicts the description of community-term pair ⟨*CERIF Project Ontology, Project*⟩. GOSPL also shows the communities adopting this gloss or the glosses that the CERIF Project Ontology has adopted for this term. Glosses are a very important means to achieve consensus within and across communities. Others can easily start a discussion to state that this gloss is equivalent with another gloss (3). The application furthermore suggests the community members to introduce concepts, fact types, etc. distilled from this gloss (2). Glosses thus provide "food for thought" to refine or

[10] Or a part of the community that felt the need to participate in a particular discussion.



Community: CERIF Project Ontology

Gloss

Gloss	Planned set of interrelated tasks to be executed over a fixed period and within certain cost and other limitations.
Creator	geert.vangrootel@ewi.vlaanderen.be
Creation Date	2012-02-01 23:53:59.0

Term-communities adopting this gloss:

- None found for this gloss.

Glosses of other communities adopted by this community for this term:

- None found for this gloss.

Analyze gloss

Concepts in this Gloss

- set, Add this concept in a fact?
- period, Add this concept in a fact?
- cost, Add this concept in a fact?

Gloss equivalence.

Community:
Term: (Choose a community first)
Motivation:

Synonyms

Community	Term	Direct
AKT Reference Ontology (Portal Ontology)	Project	true

Showing 1 to 1 of 1 entries

Fig. 13 Displaying the gloss of a community-term pair.

complete the formal part of the hybrid ontology, a process that can be facilitated by the tool. This information can be then exploited to guide the discussion processes, by transforming certain statements into queries that will look for counterexamples.

Fig. 14 depicts a simple "scenario" with the tool. After logging in, users are presented a list of communities (A), users can take a look in each community – for instance the Venue community in (B) and the discussions of that community (C). The image in (B) corresponds with the screenshot in Fig. 10. Depending whether the user is a member of a community, the user has access to a number of social processes he can start within that community. In (D), we show how a discussion to add a gloss is started. The discussion presented in (E) stems from the experiment we will describe later on. Once a term is articulated, lexons can be built around this term (F) and constraints on the created lexons (G). After a while, the community has obtained a closer approximation of their domain and can start creating/updating their application commitments (H). These commitments can be (users are not obliged) registered to the platform, which can then be used to test statements made in a discussion, e.g., by looking for counter-examples (H). When users are not part of a community, the interactions they can start only involves general requests (e.g., request an edit, or request to become a member), they have no access to requests on the glossary or lexon base. If that user is part of another community, he can trigger processes to discuss the "sameness" of glosses or terms.

Information on synonymy and gloss-equivalences are shown on a separate page (a community-term page), accessible by - for instance - clicking on one of the terms of the accepted lexons. The GOSPL tool supports a community in applying the method for ontology engineering, but its purpose is indeed not to replace other means of interaction that can be more effective when possible (e.g., face-to-face meetings when community members are near, or even teleconferences). The outcome of these interactions outside of the tool, however, needs to be properly written down when concluding a discussion.

In [10], we reported on the user satisfaction with the GOSPL ontology-engineering platform. Based on this study, we identified the main (usability) problems and drew valuable conclusions and recommendations for improvement.

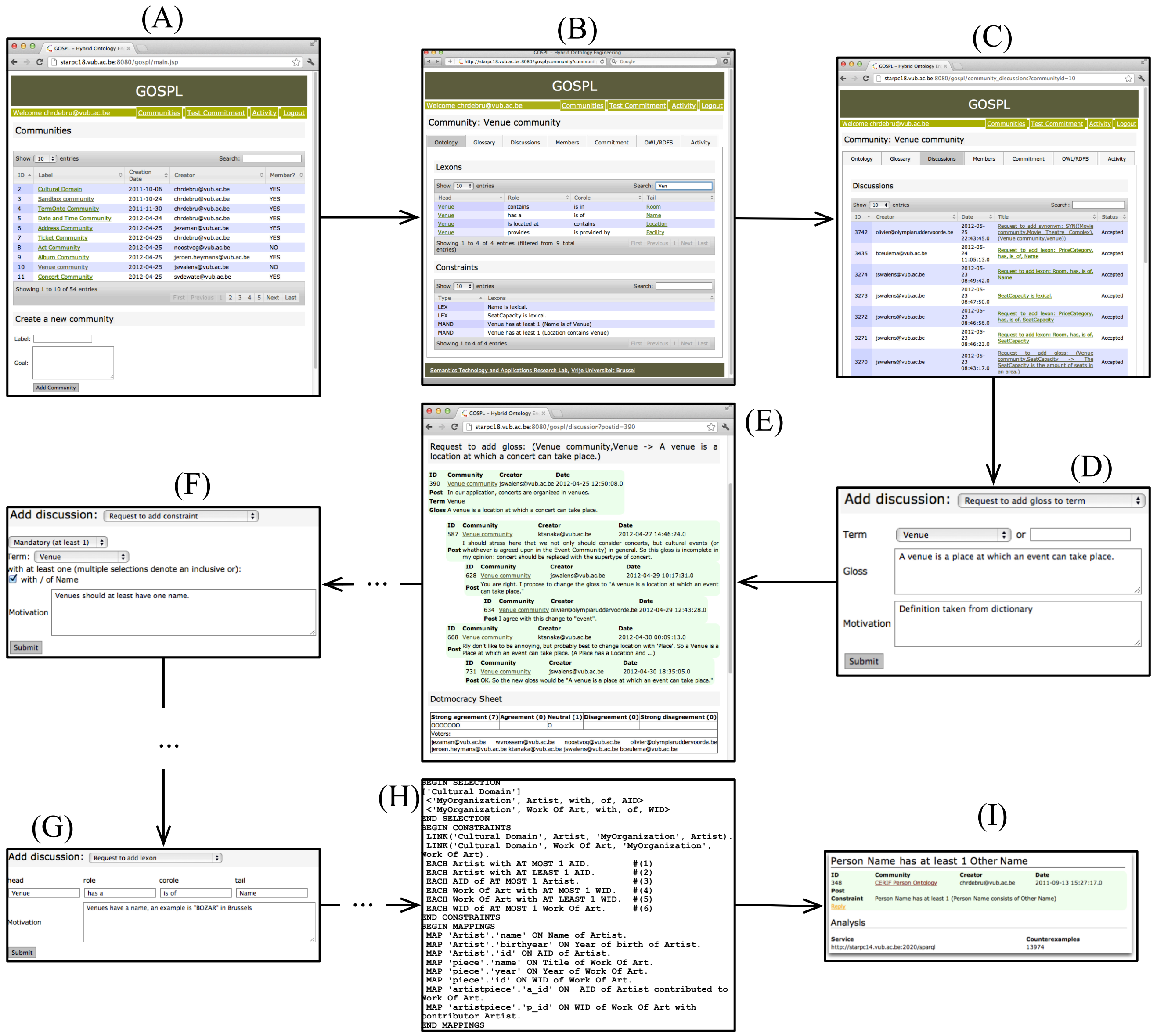
Satisfaction was measured using the standardized Post-Study System Usability Questionnaire (PSSUQ) [45,46] developed by IBM. PSSUQ originally consisted of 19 questions, each question being a statement about the usability of the system. Participants need to answer each statement using a Likert scale of 7 points, where 1 indicates that the user "strongly agrees" with the statement whilst 7 indicates that the user "strongly disagrees" with it. PSSUQ is based on a comprehensive psychometric analysis, providing scales for three sub-factors, namely: (1) system usefulness; (2) information quality; and (3) interface quality. The short (and most recent) version of PSSUQ, illustrated in Table 2, was used, in order to save time.

In Table 2, the questions correspond with the sub-factors as follows:

– System usefulness: the avg. of items 1 through 6;
– Information quality: the avg. of items 7 through 12;
– Interface quality: the avg. of items 13 through 16;
– Overall: the avg. of items 1 through 16.

In summary, the participants were successful in delivering ontologies following the method and tool. Taking the satisfaction results obtained from PSSUQ and the user comments, we derived the following conclusions [10]: out of the three sub-factors identified by PSSUQ, the system usefulness measure performed best and the information quality of the system is the sub-factor that needs to be improved the most. This also corresponds to the (Information Quality-related) problems the most cited by the users: usability problems 1, 2 and 3 in Table 3. An improvement regarding problem 1 is to introduce more information support for the user in the form of intelligible error messages or supportive documentation. From [10], we report only the major problems in this article.

Regarding the "delete" and "edit" options, most of the participants would be content with a feature that allows a post to be edited within a number of seconds. Regarding the problems related to communities, the



GOSPL – Hybrid Ontology En
starpc18.vub.ac.be:8080/gospl/main.jsp

GOSPL

Welcome chrdebru@vub.ac.be | Communities | Test Commitment | Activity | Logout

Communities

Show 10 entries Search:

ID	Label	Creation Date	Creator	Member?
2	Cultural Domain	2011-10-06	chrdebru@vub.ac.be	YES
3	Sandbox community	2011-10-24	chrdebru@vub.ac.be	YES
4	TermOnto Community	2011-11-30	chrdebru@vub.ac.be	YES
5	Date and Time Community	2012-04-24	chrdebru@vub.ac.be	YES
6	Address Community	2012-04-25	jezaman@vub.ac.be	YES
7	Ticket Community	2012-04-25	chrdebru@vub.ac.be	YES
8	Act Community	2012-04-25	noostvog@vub.ac.be	NO
9	Album Community	2012-04-25	jeroen.heymans@vub.ac.be	YES
10	Venue community	2012-04-25	jswalens@vub.ac.be	NO
11	Concert Community	2012-04-25	svdewate@vub.ac.be	YES

Showing 1 to 10 of 54 entries First Previous 1 2 3 4 5 Next Last

Create a new community

Label:

Goal:

Add Community

GOSPL – Hybrid Ontology Engineering
http://starpc18.vub.ac.be:8080/gospl/community?communit

GOSPL

Welcome chrdebru@vub.ac.be | Communities | Test Commitment | Activity | Logout

Community: Venue community

Ontology | Glossary | Discussions | Members | Commitment | OWL/RDFS | Activity

Lexons

Show 10 entries Search: Ven

Head	Role	Corole	Tail
Venue	contains	is in	Room
Venue	has a	is of	Name
Venue	is located at	contains	Location
Venue	provides	is provided by	Facility

Showing 1 to 4 of 4 entries (filtered from 9 total entries) First Previous 1 Next Last

Constraints

Show 10 entries Search:

Type	Lexons
LEX	Name is lexical.
LEX	SeatCapacity is lexical.
MAND	Venue has at least 1 (Name is of Venue)
MAND	Venue has at least 1 (Location contains Venue)

Showing 1 to 4 of 4 entries First Previous 1 Next Last

Semantics Technology and Applications Research Lab, Vrije Universiteit Brussel

GOSPL – Hybrid Ontology En
starpc18.vub.ac.be:8080/gospl/community_discussions?communityid=10

GOSPL

Welcome chrdebru@vub.ac.be | Communities | Test Commitment | Activity | Logout

Community: Venue community

Ontology | Glossary | Discussions | Members | Commitment | OWL/RDFS | Activity

Discussions

Show 10 entries Search:

ID	Creator	Date	Title	Status
3742	olivier@olympiaruddervoorde.be	2012-05-25 22:43:45.0	Request to add synonym: SYN((Movie community,Movie Theatre Complex),(Venue community,Venue))	Accepted
3435	bceulema@vub.ac.be	2012-05-24 11:05:13.0	Request to add lexon: PriceCategory, has, is of, Name	Accepted
3274	jswalens@vub.ac.be	2012-05-23 08:49:42.0	Request to add lexon: Room, has, is of, Name	Accepted
3273	jswalens@vub.ac.be	2012-05-23 08:47:50.0	SeatCapacity is lexical.	Accepted
3272	jswalens@vub.ac.be	2012-05-23 08:46:56.0	Request to add lexon: PriceCategory, has, is of, SeatCapacity	Accepted
3271	jswalens@vub.ac.be	2012-05-23 08:46:23.0	Request to add lexon: Room, has, is of, SeatCapacity	Accepted
3270	jswalens@vub.ac.be	2012-05-23 08:43:17.0	Request to add gloss: (Venue community,SeatCapacity -> The SeatCapacity is the amount of seats in an area.)	Accepted

Add discussion: Request to add gloss to term

Term: Venue or

Gloss: A venue is a place at which an event can take place.

Motivation: Definition taken from dictionary

Submit

GOSPL – Hybrid Ontology En
starpc18.vub.ac.be:8080/gospl/discussion?postid=390

Request to add gloss: (Venue community,Venue -> A venue is a location at which a concert can take place.)

ID 390 **Community** Venue community **Creator** jswalens@vub.ac.be **Date** 2012-04-25 12:50:08.0
Post In our application, concerts are organized in venues.
Term Venue
Gloss A venue is a location at which a concert can take place.

ID 587 **Community** Venue community **Creator** ktanaka@vub.ac.be **Date** 2012-04-27 14:46:24.0
Post I should stress here that we not only should consider concerts, but cultural events (or whathever is agreed upon in the Event Community) in general. So this gloss is incomplete in my opinion: concert should be replaced with the supertype of concert.

ID 628 **Community** Venue community **Creator** jswalens@vub.ac.be **Date** 2012-04-29 10:17:31.0
Post You are right. I propose to change the gloss to "A venue is a location at which an event can take place."

ID 634 **Community** Venue community **Creator** olivier@olympiaruddervoorde.be **Date** 2012-04-29 12:43:28.0
Post I agree with this change to "event".

ID 668 **Community** Venue community **Creator** ktanaka@vub.ac.be **Date** 2012-04-30 00:09:13.0
Post Rly don't like to be annoying, but probably best to change location with 'Place'. So a Venue is a Place at which an event can take place. (A Place has a Location and ...)

ID 731 **Community** Venue community **Creator** jswalens@vub.ac.be **Date** 2012-04-30 18:35:05.0
Post OK. So the new gloss would be "A venue is a place at which an event can take place."

Dotmocracy Sheet

Strong agreement (7)	Agreement (0)	Neutral (1)	Disagreement (0)	Strong disagreement (0)
OOOOOOO		O		

Voters:
jezaman@vub.ac.be wvrossem@vub.ac.be noostvog@vub.ac.be olivier@olympiaruddervoorde.be jeroen.heymans@vub.ac.be ktanaka@vub.ac.be jswalens@vub.ac.be bceulema@vub.ac.be

Add discussion: Request to add constraint

Mandatory (at least 1)

Term: Venue

with at least one (multiple selections denote an inclusive or):

☑ with / of Name

Motivation: Venues should at least have one name.

Submit

Add discussion: Request to add lexon

head	role	corole	tail
Venue	has a	is of	Name

Motivation: Venues have a name, an example is "BOZAR" in Brussels

Submit

```
BEGIN SELECTION
['Cultural Domain']
 <'MyOrganization', Artist, with, of, AID>
 <'MyOrganization', Work Of Art, with, of, WID>
END SELECTION
BEGIN CONSTRAINTS
 LINK('Cultural Domain', Artist, 'MyOrganization', Artist).
 LINK('Cultural Domain', Work Of Art, 'MyOrganization',
Work Of Art).
 EACH Artist with AT MOST 1 AID.          #(1)
 EACH Artist with AT LEAST 1 AID.         #(2)
 EACH AID of AT MOST 1 Artist.            #(3)
 EACH Work Of Art with AT MOST 1 WID.     #(4)
 EACH Work Of Art with AT LEAST 1 WID.    #(5)
 EACH WID of AT MOST 1 Work Of Art.       #(6)
END CONSTRAINTS
BEGIN MAPPINGS
 MAP 'Artist'.'name' ON Name of Artist.
 MAP 'Artist'.'birthyear' ON Year of birth of Artist.
 MAP 'Artist'.'id' ON AID of Artist.
 MAP 'piece'.'name' ON Title of Work Of Art.
 MAP 'piece'.'year' ON Year of Work Of Art.
 MAP 'piece'.'id' ON WID of Work Of Art.
 MAP 'artistpiece'.'a_id' ON  AID of Artist contributed to
Work Of Art.
 MAP 'artistpiece'.'p_id' ON WID of Work Of Art with
contributor Artist.
END MAPPINGS
```

Person Name has at least 1 Other Name

ID 348 **Community** CERIF Person Ontology **Creator** chrdebru@vub.ac.be **Date** 2011-09-13 15:27:17.0
Post
Constraint Person Name has at least 1 (Person Name consists of Other Name)
Reply

Analysis

Service	Counterexamples
http://starpc14.vub.ac.be:2020/sparql	13974

Fig. 14 Screenshots corresponding with different social processes supported by the tool.

participants wished the ability to delete communities, in particular the communities that became obsolete as the different communities evolved. Even though they understood that even those communities might once again become active, they would be happy to be able to “filter” the dead communities from the list and toggle that filter. Also the organization of the information could be improved (e.g. show more entries by default – the default number of items shown in a table is 10 and users wish to augment this number for caret browsing). Other improvements would be a list of changes after the last visit and displaying which discussions have been not yet looked at by the user.

Concerning the method we observed that terms that were articulated before lexons around this term were entered into a community commitment were less likely to have changes in their formal description than those that weren’t [20]. We analyzed the interactions involving terms in a community with the following criteria: (1) the term had to be non-lexical, meaning that instances of this concept cannot be printed on a screen, only its lexical attributes can, (2) the term was the subject of at least 4 interactions (not including gloss-equivalences and synonyms, thus focusing on the formal and informal descriptions around this term), and (3) the term took part in at least one lexon. We took into account terms with a fair amount of activity. This is due to

Table 2 PSSUQ - short version [47]

Item	Item Text
Q1	Overall, I am satisfied with how easy it is to use this system.
Q2	It was simple to use this system.
Q3	I was able to complete the tasks and scenarios quickly using this system.
Q4	I felt comfortable using this system.
Q5	It was easy to learn to use this system.
Q6	I believe I could become productive quickly using this system.
Q7	The system gave error messages that clearly told me how to fix problems.
Q8	Whenever I made a mistake using the system, I could recover easily and quickly.
Q9	The information provided with this system was clear.
Q10	It was easy to find the information I needed.
Q11	The information was effective in helping me complete the tasks and scenarios.
Q12	The organization of information on the system screens was clear.
Q13	The interface of this system was pleasant.
Q14	I liked using the interface of this system.
Q15	This system has all the functions and capabilities I expect it to have.
Q16	Overall, I am satisfied with this system.

Table 3 Summative user satisfaction. INF = Information Quality, INT = Interface Quality and SYS = System Usefulness.

	Usability problem	Nature	# of reports
1	The (error) messages displayed by the system were often not clear to the user. There was in general no online help or documentation available.	INF	6
2	There is no "undo" or "edit" option available	INF, INT	5
3	No (top menu) link to the current community in the discussion page	INT, INF	5
4	It took a while to understand how the system works	SYS	1
5	Sometimes, listing items in the dynamic tables did not go well when after returning to a page it displayed the first item again.	INT	3
6	There was no "delete" option for the communities that "died" during the process.	INF, INT	2
7	The user name is not clear (just email addresses appear)	INT	1
8	Sometimes, more clicking necessary that one would expect (e.g. when browsing through several discussions).	SYS	1

the fact that the communities employed terms only relevant to their application, and therefore only inspired discussions within that group. These discussions are not interesting as the community tended to agree on what has been decided for their application.

We then analyzed how much of these terms changed in terms of their formal description if when the gloss is provided. With these criteria, we identified 49 terms. Of these 49 terms, 38 started with the natural language description as described by the GOSPL method. Of these 38 terms, 11 of them had changes in their formal description (29%). And of the remaining 11 terms that did not start with the informal description, 5 of them changes in their formal description (45%).

The reason we left out lexicals is that they often play in an attributive role. Lexons are supposed to be entered when at least one of the terms is articulated. At the start, the key terms are often described first. And when the second term concerns a lexical in an attributive role, the community tends to agree on the meaning of this attribute based on the label of that term. If we were to take lexicals into account, we again observe that terms that did not start with an informal description are more likely to change its formal description: 18 terms out of 46 that started with a gloss and 6 terms out of 12 that did not start with a gloss.

8 Discussion

Every method needs to be teachable, repeatable and traceable. The GOSPL method for hybrid ontology engineering complies with all three criteria. The first two criteria have already been proven in industry; we went beyond the current state of affairs with the third criterion by formalizing the social processes involved. This allows us to store the whole dialogue within the community, supporting decision-making that could result in ontology evolution.

Teachable. The DOGMA framework for ontology engineering, on which GOSPL is based upon, drew inspiration from database design methods and techniques such as NIAM and ORM. NIAM/ORM and therefore also DOGMA are fact-oriented approaches in which stakeholders communicate fact types expressed in natural language.

Fact-oriented approaches differ from frame-oriented approaches (e.g., UML) by eliminating the distinction between attributes and relations; every thing is a fact between concepts. This reduces the learning curve. Unlike UML, fact-orientation was not intended to capture the dynamic aspects of a system (e.g., methods). The use of natural language to express these fact types also facilitates the knowledge elicitation processes.

Repeatable. Ontology engineering processes and possible interactions have been described and therefore repeatable by a community who have been trained or have access to the documentation. Because the method is repeatable, the third aspect – traceability – is a logical consequence.

Traceable. In order to support ontology evolution, one needs to record the changes over time. As in software engineering, it is a good practice to also document why certain changes have been made. The different evolution operators on the formal parts are therefore traceable (who, why, when, etc.), what is not often captured is the whole process of reaching a decision, with GOSPL, the social processes leading to a change in the ontology will have been formalized and stored for future reasoning.

The GOSPL tool supports a community in applying the method for ontology engineering, but its purpose is indeed not to replace other means of interaction that can be more effective when possible (e.g., face-to-face meetings when community members are near, or even teleconferences). The outcome of these interactions outside of the tool, however, needs to be properly written down when concluding a discussion. For a closer integration of other means of interaction such as teleconferences, we could draw inspiration from [48] where they presented a customizable collaborative environment focused to support ontology-based enterprise interoperability.

9 Conclusion

In computer science, the problem is not what ontologies are, but how they become to be shared and explicit agreements useable for semantic interoperability within a community. In this article we have presented a method and tool for hybrid ontology engineering called GOSPL, which stands for Grounding Ontologies with natural Language and Social Processes. Hybrid ontologies are ontologies in which concepts are both described formally and informally, where the latter uses a special linguistic resource called Glossary. For the formal descriptions, we adopted a fact-oriented ontology framework where the knowledge building blocks are binary fact types, also grounded in natural language. All agreements within and across communities of stakeholders are the result of social interactions, which are captured. In more detail, this article presented:

- A framework for hybrid ontology engineering, which constitutes the adoption of an existing fact-oriented ontology engineering approach in which the context of fact types is limited to communities. We also introduced a glossary for the informal descriptions and introduced the notion of community- and application commitment. The first capturing a communities' engagement to comply with a selection of fact types and constraints to ensure proper business and the latter a description of how one individual application commits to the ontology.
- We explained the nature of the agreements of "sameness" at both informal and formal level, and how the two interplay. We also provided a motivation why an agreement why the agreement of two glosses – used to describe two terms in two different communities – being considered referring to the same concept should not automatically imply that the terms are synonymous.
- The framework merely provides a setting in which hybrid ontologies are built, but no method to guide the community in this process. The GOSPL method was presented where – starting from key terms of the semantic interoperability requirements – one ideally starts from an informal description to describe the concepts formally. To drive the social interactions in the method, we described how we exploit glosses and application commitments. The glossary is used to extract potential new knowledge to feed the discussions. The application commitments are used to analyze some claims made by the community that is presented as additional information during the communities' interactions.

All these ideas were implemented in a tool, used in the context of a Linked Data project in Brussels. The usability study conducted in [10] showed that the method and tool helped the participants in constructing hybrid ontologies, but there were some suability problems that needed to be addressed. Furthermore, the usability study showed that the there was a need for additional documentation and information from the tool in guiding the users; a tutorial and reading material on the method and tool were deemed not sufficient for the users. An updated version of the method, documentation and tool will therefore be used in an new experiment with a similar number of participants, but in a different domain. Part of future work will be to examine to what extent GOSPL can be applied for all semantic interoperability projects.

The community model was intentionally kept simple to avoid groupthink [31]. Part of ongoing research, however, is to explore to what extent community leaders can be identified based on the interactions they have with other community members, other communities and the platform. The aim is to provide community leaders with additional privileges only to steer the discussions.

References

1. Al-Debei, M.M., Fitzgerald, G.: OntoEng: A design method for ontology engineering in information systems. Proceedings of the ACM OOPSLA'09, ODiSE **2009**, 1–25 (2009)
2. Alexander, P.R., Nyulas, C., Tudorache, T., Whetzel, P.L., Noy, N.F., Musen, M.A.: Semantic infrastructure to enable collaboration in ontology development. In: W.W. Smari, G. Fox (eds.) CTS, pp. 423–430. IEEE (2011)
3. Arpírez Vega, J.C., Corcho, O., Fernández-López, M., Gómez-Pérez, A.: Webode in a nutshell. AI Magazine **24**(3), 37–48 (2003)
4. Baader, F., Horrocks, I., Sattler, U.: Description logics. Foundations of Artificial Intelligence **3**, 135–179 (2008)
5. Borgida, A.: On the relative expressiveness of description logics and predicate logics. Artificial Intelligence **82**(1-2), 353–367 (1996)
6. Bussler, C., Tannen, V., Fundulaki, I. (eds.): Semantic Web and Databases, Second International Workshop, SWDB 2004, Toronto, Canada, August 29-30, 2004, Revised Selected Papers, vol. 3372 (2005)
7. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Path-based identification constraints in description logics. In: G. Brewka, J. Lang (eds.) KR, pp. 231–241. AAAI Press (2008)
8. Calvanese, D., De Giacomo, G., Lenzerini, M.: On the decidability of query containment under constraints. In: A.O. Mendelzon, J. Paredaens (eds.) PODS, pp. 149–158. ACM Press (1998)
9. del Carmen Suárez-Figueroa, M., Gómez-Pérez, A.: Towards a glossary of activities in the ontology engineering field. In: LREC. European Language Resources Association (2008)
10. Ciuciu, I., Debruyne, C.: Assessing the user satisfaction with an ontology engineering tool based on social processes. In: P. Herrero, H. P., R. Meersman, T.S. Dillon (eds.) On the Move to Meaningful Internet Systems: OTM 2012 Workshops, *LNCS*, vol. 7567, pp. 242–251. Springer (2012)
11. De Leenheer, P., Christiaens, S., Meersman, R.: Business semantics management: A case study for competency-centric HRM. Computers in Industry **61**(8), 760–775 (2010)
12. De Leenheer, P., Debruyne, C.: DOGMA-MESS: A tool for fact-oriented collaborative ontology evolution. In: On the Move to Meaningful Internet Systems 2008: ORM (ORM 2008), LNCS. Springer, Monterrey, Mexico (2008)
13. De Leenheer, P., Meersman, R.: Towards community-based evolution of knowledge-intensive systems. In: R. Meersman, Z. Tari (eds.) OTM Conferences (1), *LNCS*, vol. 4803, pp. 989–1006. Springer (2007)
14. De Leenheer, P., de Moor, A., Meersman, R.: Context dependency management in ontology engineering: a formal approach. Journal of Data Semantics **8**, 26–56 (2007). LNCS 4380, Springer
15. De Nicola, A., Missikoff, M., Navigli, R.: A proposal for a unified process for ontology building: Upon. In: K.V. Andersen, J.K. Debenham, R. Wagner (eds.) DEXA, *LNCS*, vol. 3588, pp. 655–664. Springer (2005)
16. De Nicola, A., Missikoff, M., Navigli, R.: A software engineering approach to ontology building. Information Systems **34**, 258–275 (2009)
17. Debruyne, C., De Leenheer, P., Meersman, R.: Empowering enterprise data governance with bsg. In: M.A. Musen, O. Corcho (eds.) K-CAP, pp. 197–198. ACM (2011)
18. Debruyne, C., Meersman, R.: Semantic interoperation of information systems by evolving ontologies through formalized social processes. In: J. Eder, M. Bieliková, A.M. Tjoa (eds.) ADBIS, *LNCS*, vol. 6909, pp. 444–459. Springer (2011)
19. Debruyne, C., Meersman, R.: GOSPL: A Method and Tool for Fact-Oriented Hybrid Ontology Engineering. In: T. Morzy, T. Härder, R. Wrembel (eds.) ADBIS, *LNCS*, vol. 7503, pp. 153–166. Springer (2012)
20. Debruyne, C., Vasquez, C.: Exploiting natural language definitions and (legacy) data for facilitating agreement processes. In: D. Winkler, S. Biffl, J. Bergsmann (eds.) SWQD, *LNBIP*, vol. 133, pp. 244–258. Springer (2013)
21. Dellschaft, K., Engelbrecht, H., Monte Barreto, J., Rutenbeck, S., Staab, S.: Cicero: Tracking design rationale in collaborative ontology engineering. In: S. Bechhofer, M. Hauswirth, J. Hoffmann, M. Koubarakis (eds.) ESWC, *LNCS*, vol. 5021, pp. 782–786. Springer (2008)
22. Fernández-López, M., Gomez-Perez, A., Juristo, N.: METHONTOLOGY: from ontological art towards ontological engineering. In: Proceedings of the AAAI97 Spring Symposium, pp. 33–40 (1997)
23. Franconi, E., Mosca, A., Solomakhin, D.: Orm2 encoding into description logics. 2012 International Description Logics workshop (DL-2012), Rome, Italy (2012)
24. Gómez-Pérez, A., del Carmen Suárez-Figueroa, M.: Scenarios for building ontology networks within the neon methodology. In: Y. Gil, N.F. Noy (eds.) K-CAP, pp. 183–184. ACM (2009)
25. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2003). ISBN: 1852335513
26. Guarino, N.: Formal ontology and information systems. In: International Conference On Formal Ontology In Information Systems FOIS'98, pp. 3–15. Amsterdam, IOS Press, Trento, Italy (1998)
27. Guha, R., Lenat, D.: Cyc: A midterm report. AI Magazine **11**(3), 32–59 (1990)
28. Halpin, T.A.: A logical analysis of information systems: static aspects of the data-oriented perspective. Ph.D. thesis, University of Queensland (1989)
29. Halpin, T.A., Morgan, T.: Information Modeling and Relational Databases. Morgan Kaufmann, San Francisco, CA, USA (2008)
30. Hepp, M.: Ontologies: State of the art, business potential, and grand challenges. In: M. Hepp, P. De Leenheer, A. de Moor, Y. Sure (eds.) Ontology Management, *Semantic Web And Beyond Computing for Human Experience*, vol. 7, pp. 3–22. Springer (2008)

31. Heylighen, F.: Self-organization in communicating groups: The emergence of coordination, shared references and collective intelligence. In: À. Massip-Bonet, A. Bastardas-Boada (eds.) Complexity Perspectives on Language, Communication and Society, Understanding Complex Systems, pp. 117–149. Springer Berlin (2013)
32. Hodrob, R., Jarrar, M.: Mapping orm into owl 2. In: A. Alnsour, S. Aljawarneh (eds.) ISWSA, p. 9. ACM (2010)
33. Holsapple, C.W., Joshi, K.D.: A collaborative approach to ontology design. Commun. ACM **45**, 42–47 (2002)
34. Jarrar, M.: Position paper: towards the notion of gloss, and the adoption of linguistic resources in formal ontology engineering. In: L. Carr, D. De Roure, A. Iyengar, C.A. Goble, M. Dahlin (eds.) WWW, pp. 497–503. ACM (2006)
35. Jarrar, M.: Mapping orm into the shoin/owl description logic. In: R. Meersman, Z. Tari, P. Herrero (eds.) OTM Workshops (1), *LNCS*, vol. 4805, pp. 729–741. Springer (2007)
36. Jarrar, M.: Towards automated reasoning on ORM schemes. In: C. Parent, K.D. Schewe, V.C. Storey, B. Thalheim (eds.) ER, *LNCS*, vol. 4801, pp. 181–197. Springer (2007)
37. Jarrar, M., Meersman, R.: Ontology engineering – the dogma approach. In: T.S. Dillon, E. Chang, R. Meersman, K. Sycara (eds.) Advances in Web Semantics I, *LNCS*, vol. 4891, pp. 7–34. Springer Berlin Heidelberg (2009)
38. Karapiperis, S., Apostolou, D.: Consensus building in collaborative ontology engineering processes. Knowledge Creation Diffusion Utilization **1**(3), 199–216 (2006)
39. Keet, M.C.: Mapping the object-role modeling language ORM2 into description logic language DLRifd. CoRR **abs/cs/0702089** (2007)
40. Kotis, K., Vouros, G.A.: Human-centered ontology engineering: The HCOME methodology. Knowledge Information Systems **10**(1), 109–131 (2006)
41. Kotis, K., Vouros, G.A., Alonso, J.P.: HCOME: A Tool-Supported Methodology for Engineering Living Ontologies. In: Bussler et al. [6], pp. 155–166
42. Kowalski, R.A.: Logic for data description. In: H. Gallaire, J. Minker (eds.) Logic and Data Bases, pp. 77–103. Plenum Press, New York, N. Y. (1978)
43. Kunz, W., Rittel, H.: Issues as elements of information systems. Working Paper 131, Institute of Urban and Regional Development, University of California, Berkeley, California (1970)
44. Lenat, D.: Cyc: a large-scale investment in knowledge infrastructure. Commun. ACM **38**, 33–38 (1995)
45. Lewis, J.R.: IBM computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use (technical report 54.786). Tech. rep., IBM (1993)
46. Lewis, J.R.: Psychometric evaluation of the pssuq using data from five years of usability studies. International Journal of Human-Computer Interaction **14**(3-4), 463–488 (2002)
47. Lewis, J.R.: Usability testing. In: Handbook of Human Factors and Ergonomics, 4 edn., pp. 1267–1312. John Wiley (2012)
48. Liapis, A., Christiaens, S., De Leenheer, P.: Collaboration across the enterprise - an ontology-based approach for enterprise interoperability. In: J. Cordeiro, J. Filipe (eds.) ICEIS (4), pp. 255–262 (2008)
49. Mann, W.C., Thompson, S.A.: Rhetorical structure theory: A theory of text organization. Text **8**, 243–281 (1988)
50. Meersman, R.: Semantic ontology tools in IS design. In: Z.W. Ras, A. Skowron (eds.) ISMIS, *LNCS*, vol. 1609, pp. 30–45. Springer (1999)
51. Meersman, R., Debruyne, C.: Hybrid ontologies and social semantics. In: Proceedings of 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST 2010). IEEE Press (2010)
52. de Moor, A., De Leenheer, P., Meersman, R.: DOGMA-MESS: A meaning evolution support system for interorganizational ontology engineering. In: Proceedings of the 14th International Conference on Conceptual Structures (ICCS 2006), *LNCS*, vol. 4068, pp. 189–203. Springer (2006)
53. Motik, B., Horrocks, I., Sattler, U.: Adding integrity constraints to owl. In: C. Golbreich, A. Kalyanpur, B. Parsia (eds.) OWLED, *CEUR Workshop Proceedings*, vol. 258. CEUR-WS.org (2007)
54. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between owl and relational databases. Journal of Web Semantics **7**(2), 74–89 (2009)
55. Nonnengart, A., Weidenbach, C.: Computing small clause normal forms. In: J.A. Robinson, A. Voronkov (eds.) Handbook of Automated Reasoning, pp. 335–367. Elsevier and MIT Press (2001)
56. Noy, N.F., McGuinness, D.L.: Ontology development 101: A guide to creating your first ontology, vol. 8. Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880 (2001)
57. Noy, N.F., Tudorache, T., De Coronado, S., Musen, M.A.: Developing biomedical ontologies collaboratively. AMIA Annual Symposium Proceedings **2008**, 520 (2008)
58. Pinto, H.S., Staab, S., Sure, Y., Tempich, C.: Ontoedit empowering swap: a case study in supporting distributed, loosely-controlled and evolving engineering of ontologies (diligent). In: C. Bussler, J. Davies, D. Fensel, R. Studer (eds.) ESWS, *LNCS*, vol. 3053, pp. 16–30. Springer (2004)
59. Pinto, H.S., Tempich, C., Staab, S.: Ontology engineering and evolution in a distributed world using DILIGENT. In: S. Staab, R. Studer (eds.) Handbook on Ontologies, pp. 153–176. Springer (2009)
60. Reiter, R.: On integrity constraints. In: M.Y. Vardi (ed.) TARK, pp. 97–111. Morgan Kaufmann (1988)
61. Simperl, E., Tempich, C.: Ontology engineering: A reality check. In: R. Meersman, Z. Tari (eds.) OTM Conferences (1), *LNCS*, vol. 4275, pp. 836–854. Springer (2006)
62. Siorpaes, K., Simperl, E.: Human intelligence in the process of semantic content creation. World Wide Web **13**(1-2), 33–59 (2010)
63. Sure, Y., Angele, J., Staab, S.: OntoEdit: Multifaceted inferencing for ontology engineering. Journal of Data Semantics **1**, 128–152 (2003)
64. Sure, Y., Staab, S., Studer, R.: Ontology engineering methodology. In: P. Bernus, J. Blazewicz, J. Schmidt Günter, M.J. Shaw, S. Staab, R. Studer (eds.) Handbook on Ontologies, International Handbooks on Information Systems, pp. 135–152. Springer Berlin Heidelberg (2009)
65. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Integrity constraints in owl. In: M. Fox, D. Poole (eds.) AAAI. AAAI Press (2010)
66. Tudorache, T., Noy, N.F., Tu, S., Musen, M.: Supporting collaborative ontology development in protégé. In: A. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. Finin, K. Thirunarayan (eds.) International Semantic Web Conference, *LNCS*, vol. 5318, pp. 17–32. Springer (2008)

67. Tudorache, T., Vendetti, J., Noy, N.F.: Web-protege: A lightweight owl ontology editor for the web. In: C. Dolbear, A. Ruttenberg, U. Sattler (eds.) OWLED, *CEUR Workshop Proceedings*, vol. 432. CEUR-WS.org (2008)
68. Uschold, M.: Building ontologies: towards a unified methodology. In: 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems, pp. 16–18 (1996)
69. Uschold, M., Grüninger, M.: Ontologies: principles, methods, and applications. Knowledge Engineering Review **11**(2), 93–155 (1996)
70. Uschold, M., King, M.: Towards a methodology for building ontologies. In: Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95 (1995)
71. Verheyden, P., De Bo, J., Meersman, R.: Semantically unlocking database content through ontology-based mediation. In: Bussler et al. [6], pp. 109–126
72. Wintraecken, J.: The NIAM Information Analysis Method: Theory and Practice. Kluwer Academic Publishers (1990)