Representing and Integrating Light-weight Semantic Web Models in the Large

Matteo Palmonari, Carlo Batini

Department of Computer Science, Systems and Communication (DISCo)
University of Milan - Bicocca
via Bicocca degli Arcimboldi, 8
20126 - Milan (Italy)
tel +39 02 64487857 - fax +39 02 64487839
{palmonari,batini}@disco.unimib.it

Abstract. Semantic Web model representation and integration can be exploited to provide organizations that deal with a large amount of data sources with an integrated view on the overall information managed. In order to support semantic Web model representation and integration in the large users must be provided with light-weight languages to represent and integrate the models, in particular avoiding the design of complex Tbox axioms. Assuming to adopt at the front-end level graph-based Concept-to-Concept Relationship (CCR) representations, in this paper we question about two semantic issues. First, we inquire whether light-weight semantic Web languages such as RDFS and DL-Lite can be used to provide the semantics of individual CCR models. Second, we inquire whether these languages can be used to provide appropriate semantics for the mappings needed for model integration. Discussing a case study in the eGovernment domain we claim that both the answer are negative. Therefore, we propose a new semantic interpretation for CCR models and we define three main classes of integration and abstraction relations defining their semantics.

1 Introduction

Semantic Web technologies and languages such as RDF, RDFS and OWL provide knowledge sharing and logical modeling capabilities based on ontologies [1], and techniques to achieve data and schema integration [2]. Web ontologies (ontologies represented in a semantic Web language) can support meta-data management, but also different applications targeted to data integration, document management, or service provision [1] by representing Web-compliant conceptual model referring to logical and conceptual schemata. Moreover, Web ontologies map to different data models of information sources, ranging from XML, to RDBMS [2]. The level of expressiveness ranges from light-weight ontologies, with taxonomies as the least expressive one, to heavy-weight ontologies with very-expressive constraints as the most expressive representative [3]. Scalability in Semantic Web ontologies is related to at least two problems. A first problem is related to the expressiveness/complexity trade-off w.r.t. reasoning: roughly speaking, the more the language of an ontology is expressive the more complex is reasoning on the ontology [1]. A second scalability problem is related to an expressiveness/cost trade-off in ontology management and engineering according w.r.t. a cost/benefit model: roughly speaking, the more a language is expressive the more it costs to manage and engineer it (design, development, maintenance and so on) [4–6,3].

This paper focuses on the latter scalability problem in semantic Web ontologies, which is gaining more and more attention in the last few years. A number of studies showed that rich Web ontologies represented in languages like OWL-DL are too costly in the large and are difficult to use for people with little formal background [5]; in particular, mastering the complex Tbox-level OWL axioms' semantics can be difficult and impacts on a number of ontology engineering costs [4]. RDFS is simpler and easier than OWL, as proved by the number of RDFS ontologies actually published on the Web [5]; in this paper, we refer to RDFS as the main light-weight semantic Web language. DL-Lite [7] provides the logical foundation for the OWL 2 QL profile of OWL 2, which is another semantic Web language that we consider to a certain extent light-weight (we will refer to this language as DL-Lite throughout the paper); in fact, it is more expressive than RDFS but by far less expressive than OWL-DL. Ontology frameworks such as semantic wikis [8,9], which are explicitly targeted to support end-users in collaborative ontology design, provide tools to simplify the design process; in these tools a simplified syntax for the specification of simple ontology axioms (domain and range restrictions on properties) prevent users from defining complex axioms using quantifiers and complex concept constructors.

Based on the above considerations, it seems that the languages/tools that are more used in fact by non skilled ontology designers, e.g. RDFS and semantic wikis, tend to present ontologies, at the front-end level, as graphs where nodes represent concepts and arcs represent relationships among these concepts in this paper we will call these models Concept-To-Concept Relationship (CCR) models. Different languages or subsets of them isomorphic to CCR models (e.g. RDFS, and DL-Lite, the Semantic Media Wiki syntax), or visual interfaces based on graphs or quantifiers-free forms (e.g. [9]) can be considered as front-end concrete languages for light-weight ontology design.

Of course the expressiveness/cost trade-off need to be considered w.r.t. a cost/benefit model, where the benefit depends on specific application contexts (e.g. DL-Lite is particularly useful for vertical data integration applications with few information sources because of its good computational properties). In this paper, we focus on contexts where conceptual models of many different sources, semantically heterogeneous and referring to different domains need to be represented and integrated; as an example consider to represent and integrate about 500 models representing the databases of the Italian public administrations. This scenario is typical when large organizations need to be provided with an overall view of the information they manage, and of their semantics, to improve the government of their data. We refer to this context as to *conceptual schema representation and integration in the large*. In this context, we often call conceptual schemata, or *schemata* for short, the semantic Web models to represent and integrate.

Assuming to represent light-weight ontologies as isomorphic to CCR schemata at the front-end level, this paper questions whether the current available semantic Web language are suitable in this context. In particular; a first research question considered in the paper is the following "in the context of conceptual schema representation and integration in the large, are the available light-weight semantic Web languages and their traditional semantics appropriate for the representation of each CCR schema?" (Q1). A second research question considered in the paper is "in the context of conceptual schema representation and integration in the large, are the current light-weight semantic Web languages, and their traditional semantics, suitable for designing the integration of such schemata?" (Q2).

By discussing a case study in conceptual schema representation and integration in the large in the eGovernment domain, the paper argues that the answer to both the questions is negative. W.r.t. Q1, the interpretation of CCR models based on standard proposal (RDFS, DL-Light) are too restrictive. An alternative semantics for CCR models is proposed to overcome the discussed limitations. W.r.t. Q2, the available languages fails to cover important loose abstraction relations among concepts of different conceptual schemata needed in the integration process. Based on the literature the paper proposes three kinds of integration-abstraction relationships, and discusses their semantics.

The paper is organized as follows: the problem context and the case study are introduced in Section 2; the problems and the proposed solution for individual schema representation and their integration are discussed in Section 3; related works are discussed in Section 4; conclusions end the paper in Section 5.

2 Schema Representation and Integration in the Large

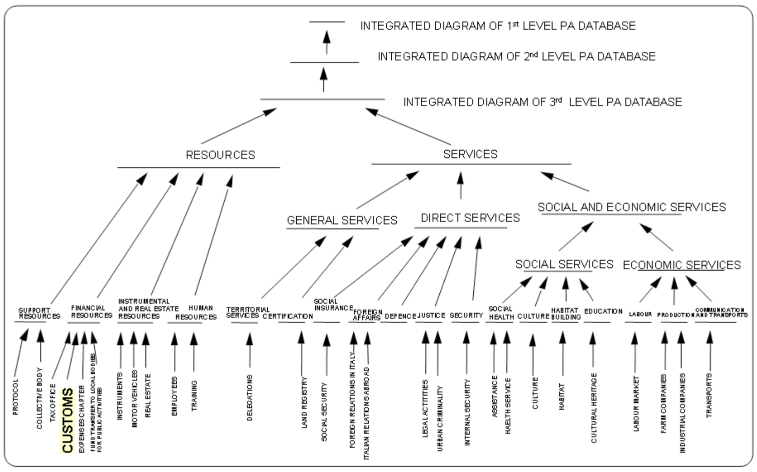


Fig. 1: The multi-layered conceptual schemata' repository of the Italian central public administrations

Schema representation and integration in the large support conceptual meta-data management, in order to provide large organizations, or networked enterprises with an integrated view of the information managed. To make an example we consider a case study referring to past experiences in the design of repository of conceptual schemata in the eGovernment domain.

Experiences in the design and exploitation of structured repositories of conceptual schemata (Entity Realtionship schemata) related to the most relevant databases of the Italian central public administrations are described in [10]. A Central Public Administration repository of schemata (CPA repository) has been developed to provides public institutions with a conceptual meta-data management framework; the structure of the repository is shown in Figure 1, where each node in the hierarchy represents an ER schema. The bottom level of the CPA repository consists of approximately 500 conceptual schemata representing at the conceptual level logical schemata of the information sources (*basic schemata* - not represented in the figure); these basic schemata are recursively clustered and integrated by exploiting integration and abstraction primitives defined in [11]; the bottom level of the figure shows the conceptual schemata obtained from the basic schemata at the first integration and abstraction step.

The benefits of exploiting structured repositories of schemata at the back-end and at the front-end level, e.g. to improve government-to-citizen and government-to-business relationships, have been discussed in [10]. As for the exploitation of repositories at the application level, the CPA repository has been exploited to support the semi-automatic construction of a Local Public Administration (LPA) repository based on reverse engineering techniques [10]. Building such a repository with current semantic Web languages could bring even more benefits, such as the possibility to exploit the concepts and relationships for semantic annotation and search in SOA, document management or data integration initiatives.

2.1 Conceptual Schema Representation

Focusing on the representation of individual schemata in the repository, the context of conceptual schema integration in the large is characterized by the necessity of representing and integrating many schemata (about 500 in the example), referring to many heterogeneous domains (e.g. financial resources, certifications, justice, security, education, and so on). However, a number of concepts are shared among the different schemata (e.g. the concept of Subject appear in most of the different eGovernment domains); these concepts are the key concepts to integrate the different schemata [10]. Such a scenario requires a lot of effort for ontology design and engineering. It is very difficult to commit such an effort to one skilled and experienced ontology designer. Experts in the domain need to be provided with tools to design their models. Because of the amount of the schemata to design and integrate, and the designers' profiles (domain experts with little formal background), light weight languages are needed to cope with the expressiveness/cost trade-off. Moreover, capturing deep ontological commitments (e.g. with cardinality restrictions) it is often difficult (e.g. at the more abstract levels) and not useful (e.g. to support softer tasks such as navigation, semantic annotation and search rather than more specific reasoning-based tasks such as data integration).

In order to support scalable design of the schamas in the large, it is important to consider the knowledge continuum perspective [3]. According to this perspective, knowledge is represented in a continuum of knowledge artifacts represented in languages with different expressivity. In the context addressed in this paper, focusing on schemata of semi-structured and structured information sources, this means that light-weight semantic models could be used as sketches to develop more expressive Web ontologies when needed. As an example, suppose that a new eGovernment data integration project targeted to the Human Resources' data sources (see Figure 1) is started: richer ontologies with more expressive axioms might be needed in this case. However, the reuse of the light-weight models defined at the conceptual meta-data management level should be guaranteed: the semantics of such light-weight models need not to be inconsistent with the new axioms. We will refer to this point as to the *knowledge continuum* issue.

Given the amount of schemata to represent, their levels of detail and the need for a collaborative design process carried out by domain experts, the CCR expressivity level can be considered a good compromise between the need of exploiting at the front-end level a language easy to be used, and the need of going beyond taxonomies representing at least the relationships between concepts. Considering the case study discussed, although the CPA repository is based on the ER model, the schemata represented are not far from CCR models (no cardinality restrictions are used and relationships with arity greater than 2 represents less than 5% of the total number of relationships used).

2.2 Conceptual Schema Integration

In the context addressed in the paper (when many schemata are considered) a one-step integration is nearly impossible, and schemata need to be integrated in a recursive way: sets of schemata clustered according to similarity criteria and balanced according to their levels of detail (LOD) are merged together by means of a *schema integration* primitive (see [12] for schema integration mechanisms); the process is iterated on the resulting set of schemata. However, the integration of schemata would easily lead to schema very large in size with difficulties in their management and comprehension. *Schema abstraction* primitives can be applied to obtain a schema at coarser LOD from a given schema. When considering the application of an integration or an abstraction primitive, we will call *source schemata* the schemata to which the primitive is applied, and *target schema* the schema resulting from the application of the primitive; we will call *source concepts* the concepts of the source schemata and *target concepts* the concepts of the target schemata. In practice, integration and abstraction primitives are often applied together to obtain an integrated target schema at a coarser LOD. We will refer to this primitive as *integration-abstraction*; observe that this primitive is more general than abstraction, and the application of an abstraction primitive can be considered a special case of integration-abstraction applied to a single schema. As a result, this iterative integration-abstraction process leads to consider schemata at progressively coarser LOD, that is, that are progressively more abstract.

For the details of the methodology to carry out this iterative integration process we refer to [11]. In order to provide a more detailed example of schema integration-abstraction, we consider the Customs domain highlighted in Figure 1; the conceptual schema of the Custom domain is obtained by means of integration-abstraction mechanisms applied to three basic schemata, namely Custom Agencies, Custom Declarations and Item Categorization. Figure 2 represents a simplified version of this example (the size of the schemata is reduced to illustrate the main points addressed in this paper); broken-lined arrows represent subclass relationships, thick A-arrows represent

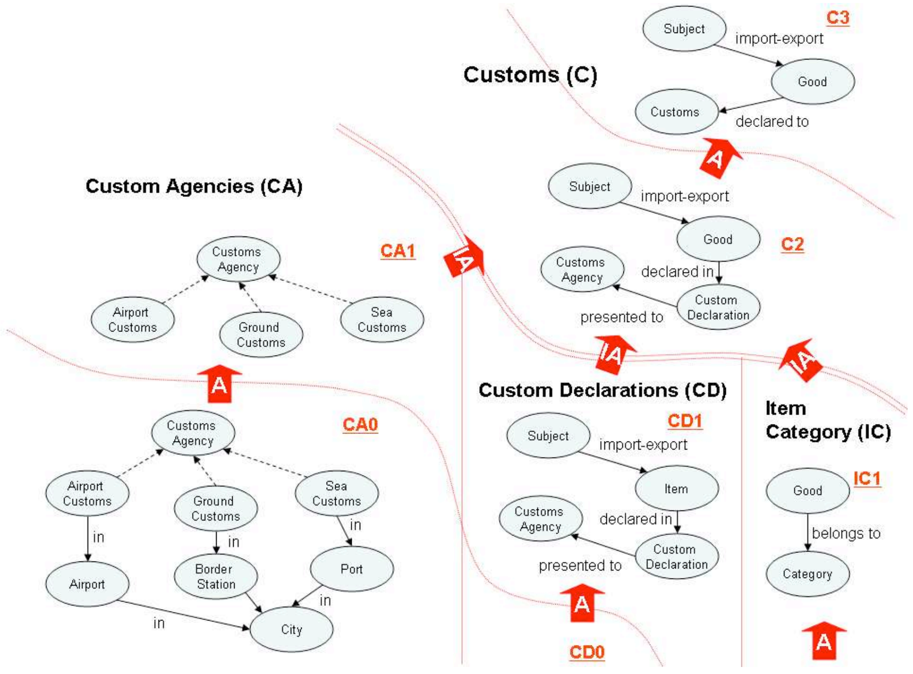


Fig. 2: Examples of integration and abstraction of ER conceptual schemata

schema abstraction primitives, and thick IA-arrows represents integration-abstraction primitives; four LODs are represented (only the schema CA0 - Custom Agency at level 0 - is drawn as representative of the bottom level; the schemata CD0 and IC0 are omitted); the schemata are represented as CCR models. An abstraction primitive is applied to CA0, producing the schema CA1 (where the locations the custom agencies are associated with are discarded). The schemata CA1, CD1 and IC1 are integrated-abstracted in the schema C2. The schema C2 is abstracted into the schema C3, where the two concepts Custom Agency and Custom Declaration are replaced by the concept Customs, in order to express, at a more abstract level, that Goods are declared to Customs.

The question Q2 addressed in the paper concerns the suitability of available lightweight ontology Web languages to represent these kind of integration-abstraction primitives. Assuming to represent each source schema as a Web ontology, standard import and namespace mechanisms in OWL and RDFS easily allows for the integration of schemata at the architectural level (e.g. to define that C2 integrates the three schemata CA1, CD1 and IC1 one could define a Web ontology C2 importing CA1, CD1 and IC1). The question therefore focuses on the representation of the mappings between the schema concepts: are the available language suitable to represent the set of significant relations occurring between the entities of the source and the target schemata? As an example, is it possible to trace out that the specific locations associated with the three types of custom agencies in the schema C0 of Figure 2 are *forgot* in CA1?

3 Semantics for Conceptual Schema Integration Based on CCR Models

In order to better define the problem, we introduce the notion of schema integration-abstraction framework and an abstract CCR formalism.

Given a set of schemata Σ, a *schema integration-abstraction framework* can be formalized by a schema integration-abstraction operation $f^{\Sigma} : \mathcal{P}(\Sigma) \longmapsto \Sigma$; as an example, the schema integration-abstraction framework based on Figure 2 is defined by a f^{Σ} such that $f^{\Sigma}(\{CA0, CD0, IC0\}) = C1$, and $f^{\Sigma}(\{C1\}) = C2$. In the following we assume that the schema integration-abstraction frameworks are *partitive*, that is, each schema is integrated-abstracted in only one schema).

The abstract CCR formalism is defined through a language (CCR language from now on) that represents concepts, binary relationships between concepts and subconcept relationships; moreover, since the notion of inverse property is quite intuitive for binary directed relationships, the notion of inverse property is introduced (scalability w.r.t. reasoning is not addressed in this paper); finally, specific relationships to represent inter-schema integration-abstractions mappings are introduced. In order to avoid misunderstandings w.r.t. to technical notions from standard ontology Web languages (in particular in the next sections when a translation-based semantics is provided), we adopt the following conventions inspired by the Entity Relationship model: concepts in the CCR formalisms will be called *entities* and subconcept relationships will be called *generalization* relationships.

Formally, the CCR syntax is defined as follows.

Definition 1. ***Concept-to-Concept Relationship (CCR) Alphabet****. An CCR alphabet $\mathcal{A} = (\Sigma, E, R, gen, IA, i^{inv})$, is a tuple where: Σ is a set of schema names, E is a set of entity names, R is a set of binary relationship names, gen is the generalization relation symbol, IA is a set of names of integration-abstraction relations, $i^{inv} = R \rightarrow R$ associates relation names with names of their inverse relations, and the sets Σ, E, R, gen, IA are pairwise disjoint.*

Definition 2. CCR **language and** CCR **schemata.** *Given a CCR alphabet $\mathcal{A} = (\Sigma, E, R, gen, IA, i^{inv})$, a CCR language $\mathcal{L}^{CCR}$ based on $\mathcal{A}$ is the set of sentences having the form:*

- *intra-schema $\mathcal{L}^{CCR}$ sentences*
  - $S{:}r(S{:}e, S{:}f)$;
  - $S{:}r^{-}(S{:}e, S{:}f)$;
  - $gen(S{:}e, S{:}f)$;
- *inter-schema $\mathcal{L}^{CCR}$ sentences*
  - $ia * (S{:}e, S'{:}f)$

where $S \in \Sigma$, $r \in R$,$\{e, f\} \in E$, and $ia* \in IA$, and (r^{-} is a short for $i^{inv}(r)$). Given a language $\mathcal{L}^{CCR}$ defined over an alphabet $\mathcal{A}$, an CCR schema S is a set of intra-schema sentences $\Phi \subseteq \mathcal{L}^{CCR}$. Statements having the form $S{:}r(S{:}e, S{:}f)$ and $S{:}r^{-}(S{:}e, S{:}f)$ are called CCR patterns; a CCR pattern whose relation is r is called CCR r-pattern. Given a schema integration-abstraction framework defined on Σ by a *structuring function f^{Σ}, a set of inter-schema $\mathcal{L}^{CCR}$ sentences Ψ is valid iff for every S and S' such that $ia * (S{:}e, S'{:}f) \in \Psi$ there exist a set of schemata $T \subseteq \Sigma$ such that $S \in T$ and $f^{\Sigma}(S) = S'$.*

As an example, the schema CD1 of Figure 2) is conceptually represented in the CCR language by means of the following statements:
$CD1{:}import - export(CD1{:}Subject, CD1{:}Good)$,
$CD1{:}declared\_in(CD1{:}Good, CD1{:}Cus.\_Decl.)$,
$CD1{:}presented\_to(CD1{:}Cus.\_Decl., CD1{:}Cus.\_Agency)$ (in order to avoid long names the following abbreviations are used in the paper: "Cus." for Custom, "Decl." for declaration, "Adv." for adventure). In the following, when the schema that intra-schema $\mathcal{L}^{CCR}$ sentences refer to is clear from the context, or not relevant, the schema reference will be avoided for sake of clearness (the more compact notation $r(e, f)$ will be used to denote CCR patterns).

One of the peculiar characteristics of CCR schemata is the *Multiple Use of Relationship Names* (MURN) in a same schema; as an example, more than one *in* relationships are represented in the CA0 schema of Figure 2. Many conceptual modeling languages, e.g. the ER language, formally assume the Single Use of Relationship Names (SURN). SURN means that a schema such as CA0 of Figure 2 cannot be represented and specific different names for each of the involved relationship need to be introduced (e.g. $in\#1$, $in\#2$, etc.). SURN can be defined more formally as follows.

Definition 3. ***SURN condition and SURN assumption.*** *Given a schema $S = \varphi_1, ..., \varphi_n$ defined over a language $\mathcal{L}^{CCR}$, the SURN property holds for S iff there not exist two CCR patterns $r(e_1, e_2)$ and $r'(f_1, f_2)$ in S such that $r = r'$. We call a SURN-schema a schema for which the SURN property holds. Given a set Σ of CCR schamas, the SURN assumption holds for Σ iff every schema $S \in \Sigma$ is a SURN-schema.*

We call MURN-schemata the CCR schemata for which the SURN property is not required to hold, and we call MURN the relaxation of the SURN assumption for a set of schemata. Observe that SURN-schemata are MURN-schemata, while the converse does not hold.

3.1 Representing MURN CCR schemata with sound semantics

In the following we exploit a DL notation defined for OWL ($\mathcal{SHOIQ}^{\mathcal{D}}$) and RDFS (based on $DL-Lite$), as defined respectively in [1] and [7]; the DL-Lite axioms $\exists R \subseteq C$ and $\exists R^{-} \subseteq C$ (equivalent to $\exists R.\top \subseteq C$ and $\exists R^{-}.\top \subseteq C$ in $\mathcal{SHOIQ}^{\mathcal{D}}$) represent that C is respectively the domain or the range of the DL role R, where R^{-} denotes the inverse role of R.

The representation of MURN schemata is a crucial but often overlooked issue in light-weight ontology modeling. First, consider the abstract nature of the models represented in the schemata. SURN forces a proliferation of relationship names for relationships with a unique intuitive meaning (e.g. three relationship names in_1,in_2, and in_3 would be needed - under SURN - in the CA0 schema of Figure 2). In the context addressed in this paper this point is particularly relevant: besides the amount of schemata to represent, many generic relationships are used (e.g. has, use, is related to, part of, and so on), and particularly in the more abstract schemata. Second, there are many references to MURN schemata in the literature: SURN is not assumed in the early semantic nets and is often violated even for languages such as ER for which it is supposed to hold (e.g. see examples in [13]). Third, in our past experiences in the design of repositories of ER schemata SURN was not adopted by the designers: the SURN assumption was systematically violated in the CPA repository (e.g. the "related to" relationship is used up to 7/8 times in a same schema, which consists of less than 20 entities).

Representing MURN CCR schemata by means of a light-weight language such as RDFS is not possible. RDFS easily maps to CCR under the assumption that CCR patterns are represented by domain and range restrictions; e.g. $in(Aiport\_Cus., Airport)$ of CA0 in Figure 2 is interpreted as the assertion that $Aiport\_Cus.$ and $Airport$ are respectively domain and range of the relationship in. However, according to the semantics of RDFS, multiple domain and range assumptions have a conjunctive interpretation; which means that in the CA0 schema the domain of in consists of the intersection of all the concepts the arcs labelled as in start from ($Aiport\_Cus.$,$Airport$, and so on). As a result, RDFS semantics does not capture the intended semantics of CCR patterns in MURN schemata.

However, there can be other possible DL-Lite and OWL-DL interpretations of CCR patterns that conflict with specific CCR pattern combinations, as shown in Figure 3. In particular, it is remarkable that participation constraints that can be represented in DL-Lite do not allow for (0,n) cardinality restrictions, which when representing abstract schemata can be assumed as default cardinality restrictions (they impose the lighter possible constraints on the underlying data models).

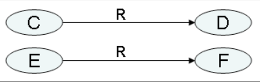
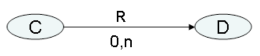
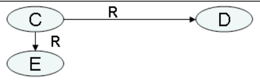
CCR pattern DL formalization			**Conflicting CCR pattern**
$\{\exists R \sqsubseteq C,$ $\exists R^{-} \sqsubseteq D\}$	RDFS DL-Lite	**Role-typing** First elements in a relation R are of type C, second elements in a relation R are of type D	
$C \sqsubseteq \exists R$	DL-Lite	**Existential Restriction** All instances of C have a filler for the relation R	
$C \sqsubseteq \exists R.D$	OWL-DL	**Qualified Existential Restriction** All instances of C have a filler for the relation R of type D	
$C \sqsubseteq \forall R.D$	OWL-DL	**Qualified Universal Restriction** Second elements in a relation in R whose first elements are of type C are of type D	

Fig. 3: Possible CCR pattern interpretations and conflicting CCR patters

To address the above problems we propose a new semantic interpretation for CCR patterns based on an automatic deterministic translation of CCR light-weight schemata into OWL-DL ontolgies. As usual, $\mathcal{L}^{CCR}$ entities are interpreted as OWL-DL concepts and $\mathcal{L}^{CCR}$ relationship names are interpreted as OWL-DL properties. Our interpretation is characterized by three main assumptions. A very light interpretation of the first entity in a CCR R-pattern as a concept included in the domain of R, and of the second entity in the CCR R-pattern as a concept included in the range of R (*domain/range inclusion*). A first epistemic closure that states that the domain (range) of a relationships R is the disjunction of all the concepts corresponding to the entities occurring as first (second) elements in any CCR R-pattern (*domain/range global union*). A second epistemic closure that captures the conditional constraint represented in a CCR pattern of the form $r(e, f)$, that is, that when the first element of a tuple in r is of type e, then the second element of the tuple is of type f; of course we need to consider MURN and adequately treat multiple conditional range (domain) restrictions (e.g. when two CCR patterns $r(e, f)$ and $r(e, f')$ are considered); the strategy is analogous to domain/range global union but it is conditional to specific domain/range concepts *(domain/range conditional union)*. Observe that domain/range inclusion and domain/range global union can be represented by defining the domain/range to be equivalent to the union of all the concepts occurring in the domain/range global union specifications (*domain/range global equivalence*).

As an example, consider the relation in and the in-related patterns of schema CA0. The concepts $Aiport\_Cus.$, $Sea\_Cus.$, $Ground\_Cus.$, $Airport$, $Border\_Station$, $Port$ are subconcepts of the in domain (domain inclusion), which in CA0 consists of the union of these concepts (domain global union); the concepts $Airport$, $Border\_Station$, $Port$, and $City$ are subconcepts of the in range (range inclusion), which in CA0 consists of the union of these concepts (range global union); moreover, given a tuple $< x, y > \in in$: if $x \in Aiport\_Cus.$, then $y \in Airport$; if $x \in Sea\_Cus.$, then $y \in Port$; if $x \in Ground\_Cus.$, then $y \in Border\_Station$; analogous conditional interpretations occur for the other in-patterns in the schema; as for domain conditional union, let us focus on the three in-patterns $in(Airport, City)$, $in(Border\_Station, City)$, and $in(Port, City)$: in this case the interpretation is that, given a tuple $< x, y > \in in$, if $y \in City$, then $x \in Airport \sqcap Border\_Station \sqcap Port$.

The formal conceptual semantics for $\mathcal{L}^{CCR}$ can be defined translating $\mathcal{L}^{CCR}$ schemata into OWL-DL ontologies, where entities are represented by OWL concepts, relationships by OWL properties and CCR patterns by restrictions on properties, according to the mappings defined in Table 1. In the table we adopt the following compact notation: $r(e, \{f_1, ..., f_k\})$ represents the set of $\mathcal{L}^{CCR}$ assertions where e occurs as a first element in a relation r, $r(\{e_1, ..., e_h\}, f)$ represents the set of $\mathcal{L}^{CCR}$ assertions where f occurs as second element in a relation r, and $r(\{e_1, ..., e_h\}, \{f_1, ..., f_k\})$ represent the set of all the $\mathcal{L}^{CCR}$ assertions about a relation r where one element of the first set occurs as first element and one element in the second set occurs as second element.

Observe that based on the epistemic closures, this semantic provides an interpretation of schemata that is relative to an epistemic state. If the schemata are changed, the semantics should be computed again. This is consistent with the aim of this paper: we do not propose to design CCR models with the OWL-DL language (we would not be consistent with our assumptions). The semantics proposed is aimed at providing formal translations at the back-end level for front-end CCR models, that is, in a transparent way to the designers. In this paper we claim that our proposals allows for the more freedom in the design of CCR models without conflicts with possible CCR patterns and without giving up a set-theoretic semantics.

Table 1: Translation from binary $\mathcal{L}^{CCR}$ MURN schemata to $OWL - DL$ ontologies

$\mathcal{L}^{CCR}$		$\mathcal{SHOIQ}^{\mathcal{D}}$ (OWL-DL)	Intuitive Semantics
e	$\mapsto$	C^e	Concepts
r	$\mapsto$	P^r	Properties
$r(\{e_1, ..., e_h\},$	$\mapsto$	$\exists R.\top \equiv C^{e_1} \sqcup ... \sqcup C^{e_h},$	domain global equivalence
$\{f_1, ..., f_k\})$		$\exists R^{-}.\top \equiv C^{f_1} \sqcup ... \sqcup C^{f_k}$	range global equivalence
$r(\{e_1, ..., e_k\}, f)$	$\mapsto$	$C^f \sqsubseteq \forall R^{-}.(C^{e_1} \sqcup ... \sqcup C^{e_k}),$	domain conditional union
$r(e, \{f_1, ..., f_k\})$	$\mapsto$	$C^e \sqsubseteq \forall R.(C^{f_1} \sqcup ... \sqcup C^{f_k});$	range conditional union
$gen(e, f)$	$\mapsto$	$C^e \sqsubseteq C^f$	

3.2 Loose integration of CCR schemata

A set of source schemata are integrated-abstracted in order to provide a target schema that accounts for the knowledge represented in the source schemata at a coarse LOD. The integration-abstraction primitive is based on the application of a set of abstraction mechanisms that, given a set of source concepts, provide an abstract target concept that represent the source concepts. The issue addressed in this paper is related to the representation of the relationships (or mappings) that might occur between the source concepts and the target concept that abstracts them.

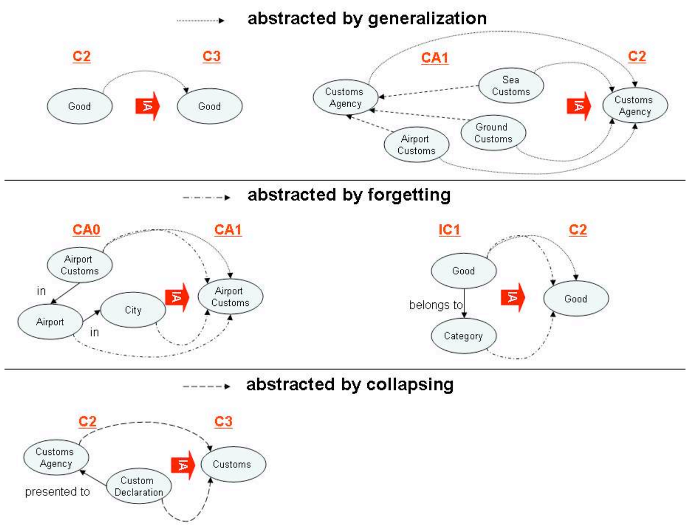


Fig. 4: Examples of the three classes of integration-abstraction relationships

Different kinds of abstraction mechanisms can be applied as to a set of entities, resulting in different kinds of integration-abstraction relations. In this paper we focus on three abstraction mechanisms for CCR schemata, namely *abstraction by generalization*, *abstraction by forgetting* and *abstraction by collapsing*; these abstraction mechanisms are used in the case study described in Figure 2 and have been acknowledged in the literature, althought sometimes under different naming (see Section 4 for details). Figure 4 provides some examples of the application of three mechanisms taken from the case study represented in Figure 2; sets of source concepts are replaced in the target schema by one concept (observe that these sets might trivially consists of one schema, and that the target concept might have the same name of the source schema).

We discuss the semantics of these abstraction mechanisms by introducing three different integration-abstraction relations - and the respective inverse relations - needed to represent the respective abstraction mechanisms. These relations provide the characterization of the IA set in a $\mathcal{L}^{CCR}$ language:

1. $abstract{-}by{-}generalization$ ($a\text{-}generalize$ for short), and the inverse $abstracted-by-generalization$ ($a\text{-}generalized$ for short). This relation represents generalizations between sets of entities of different schemata with standard subsumption semantics; looking at the right-most example in the top-most section of Figure 4), the entities $Cus.\_Agency$, $Sea\_Cus.$, $Airport\_Cus.$, and $Ground\_Cus.$ of CA1 are $a\text{-}generalized$ by the entity $Cus.\_Agency$ of C2 (i.e. $a\text{-}generalized(C1{:}sea\_cus., C2{:}cus.\_agency)$, and so on).
2. $abstract - by - forgetting$ ($a\text{-}forget$ for short), and the inverse $a\text{-}forgot$. It represents abstractions of source entities that are "sunk" in a more abstract target entity, discarding some details in the source representations; looking at the right-most example in the middle section of Figure 4), the entities $Good$ and $Category$ of IC1 are $a\text{-}forgot$ in the entity $Good$ of C2 (i.e. $a\text{-}forgot(IC1{:}category, C2{:}good)$ and $a\text{-}forget(IC1{:}good, C2{:}good)$).
3. $abstract - by - collapsing$ ($a\text{-}collapse$ for short), and the inverse $a\text{-}collapsed$. It represents abstraction mechanisms in which the target concept has a different meaning w.r.t. all the source concepts; looking at the example in the bottom-most section of Figure 4), the entities $Cus.\_Agency$ and $Cus.\_Decl.$ of schema C2 are $a\text{-}collapsed$ in the entity $Customs$ of schema C3 (i.e. $a\text{-}collapsed(C2{:}cus.\_agency, C3{:}cus)$ and $a\text{-}collapsed(C2{:}cus\_decl., C3{:}cus)$).

Intuitively, $a\text{-}collapse$ and $a\text{-}forget$ are quite similar, but $a\text{-}forget$ relations are polarized on an entity: there exists one entity in the source schema whose instances can be considered also instances of the abstract entity. This can be modeled by introducing also an abstraction by generalization relation for such an entity: e.g. in Figure 4 the entity $Good$ of C2 $a\text{-}generalize$ the entity $Good$ of IC1, which is represented by the sentence $a\text{-}generalize(C2{:}good, IC1{:}good)$. Generalizations cannot be established for $a\text{-}collapse$, where for none of the source entities it can be assumed that instances are also instances of the target entity; as an example, the entity $Customs$ in schema C3 represents the general notion of customs as public institutions; custom declarations and custom agencies are not "customs" according to this meaning. For this reason, the intuitive meaning of $a\text{-}collapse$ includes "part of"-like aggregation and the grouping relations introduced in [14] (customs as public institution are composed of other entities, among which custom declarations and custom agencies), and is almost equivalent to unfolding relations as introduced in [15]. Observe that this shift in meaning might occur even when only one concept is $a\text{-}collapsed$ into another concept (e.g. imagine that custom agencies are represented by a concept named Custom).

How do available light-weight semantic Web languages behave w.r.t. the representation of the above mechanisms? First, consider that only abstraction by generalizations can be natively codified as relations, namely subsumption relations ($\sqsubseteq$), between two concepts. These relations can easily be interpreted as subsumptive mappings traditionally used in data integration [2]. If we consider abstraction by forgetting, subsumption relations between the source concept the forgetting mechanism is polarized on can be represented but the information about the other forget source concept is lost. If we consider abstraction by collapsing, none of the source concept can be mapped with a subsumption relation to the target concept. The representation of integration-abstraction relations is not straightforward.

In order to overcome these problems one could introduce specific relations, e.g. $a\text{-}forgot$ and $a\text{-}collapsed$ to be used in more complex axioms; as an example, consider to represent that $Cus.\_Agency$ of C2 is $a\text{-}collapsed$ in $Customs$ of C3. Here different options are available: (A) $Cus.\_Agency$ and $Customs$ are respectively domain and range of $a\text{-}collapsed$; (B) $Cus.\_Agency \sqsubseteq \forall a\text{-}collapsed.Customs$; (C) $Cus.\_Agency \sqsubseteq \exists a\text{-}collapsed.Customs$. The option (A) cannot be adopted because $a\text{-}collapsed$ relations have more than one concept as domain, as clearly represented in Figure 4 (the label "0,n" in the figure refers to intended interpretations of a CCR pattern $r(c, d)$ as [O,n] cardinality constraints between two classes C and D, e.g. in the UML model [14]); observe that the same argument applies to the inverse relations w.r.t. range restrictions. Unfortunately (A) is the only option available assuming RDFS or DL-Lite; hence the negative answer to Q1.

Moreover, assume to represent these multi-layered mappings (*multi-layering*) by means of one ontology that import all the CCR schemata and defines their mappings. The resulting ontology is clearly based on MURN, which means that more complex strategies are needed to represent integration-abstraction relations in the context addressed in this paper. This is another argument against the adoption of the option (A).

The option (B) is safe against the above arguments, but does not capture the strong commitments in the definition of the integration-abstraction relations. If option (C) is applied to $a\text{-}collapse$ and $a\text{-}forget$, we have a case similar to qualified universal restriction depicted in Figure 3. However, this does not occur if we consider their inverse relations $a\text{-}collapsed$ and $a\text{-}forgot$ because a set of source concepts are $a\text{-}collapsed$ and $a\text{-}forgot$ into at most one schema. We therefore propose a solution based on the functional interpretation of the relations $a\text{-}collapsed$ and $a\text{-}forgot$, and the exploitation of inverse property axioms (last two rows of Table 2). Formally, this interpretation is represented in Table 2.

4 Related Works

CCR models largely overlap with simple semantic nets whose nodes represent concepts (and not instances). CCR models are isomorphic to Concept Maps [16] and to RDFS

Table 2: Semantics for $\mathcal{L}^{CCR}$ ia-relations

$\mathcal{L}^{CCR}$		$\mathcal{SHOIQ}^{\mathcal{D}}$ (OWL-DL)
$a\text{-}generalized(S{:}e, S'{:}f)$	$\mapsto$	$C^{S:e} \sqsubseteq C^{S':f}$
$a\text{-}forgot(S{:}e, S'{:}f)$	$\mapsto$	$C^{S:e} \sqsubseteq \forall a\text{-}forgot.C^{S':f}$
$a\text{-}collapsed(S{:}e, S'{:}f)$	$\mapsto$	$C^{S:e} \sqsubseteq \forall a\text{-}collapsed.C^{S':f}$
$a\text{-}generalize(S{:}e, S'{:}f)$	$\mapsto$	$C^{S':f} \sqsubseteq C^{S:e}$
$a\text{-}forget$	$\mapsto$	$a\text{-}forget \equiv a\text{-}forgot^{-}$
$a\text{-}collapse$	$\mapsto$	$a\text{-}collapse \equiv a\text{-}collapsed^{-}$

under the interpretation given in Section 3 (if we assume not to consider property hierarchies, not relevant to the claim of the paper); moreover, tools like Semantic MediaWiki [8] and MoKi [9], which make the user specify global or local domain/range restrictions through quantifier and cardinality-free forms or shortcuts, are based on a front-end design language isomorphic to CCR models. CCR patterns in Semantic MediaWiki are based on RDFS [8], which means that, in theory, only SURN schemata can be represented. The interpretation of CCR patterns in MoKi is not clear from [9]; there are reason to believe that their interpretation is based on qualified existential range restriction (the third top-most interpretation represented in Figure 3)

The work more related to our proposal the translation from concept maps to OWL ontologies proposed in [6]. The proposed transformation covers more complex CCR models than the one covered in this paper (e.g. it considers also instances as part of the maps). They interpret Concept Maps *propositions* (analogous to CCR patterns) as domain/range global union, and also refer to WordNet to disambiguate between instances and concepts. However, they do not introduce any conditional domain/range union semantics, and therefore they do not capture specific conditional dependencies represented in the Concept Map propositions (see Section 3). Finally, the interpretation they provide for the specification of multiple ranges for a same property looks counterintuitive; e.g. the proposition $(Activity, hasType, \{Air\_Adv, Sea\_Adv\})$, is translated into the axiom $Activity \sqsubseteq \exists hasTypeAir\_Adv. \sqcap \exists hasTypeSea\_Adv.$, from which it can be derived that an activity has always two types.

The approach to schema integration in the large based on integration-abstraction primitive is based on the approach introduced in [11]. However, that approach was based on Entity Relationship schemata, while here we discuss how to exploit the approach in a semantic Web framework. Moreover, the classification of the three abstraction mechanisms, the relations to represent them, and their semantics are new contribution of this paper. This approach to schema integration is very close to traditional techniques for data integration, where the concepts of local schemata are mapped to the concepts of a global schema [2]. At a schema-level, we differ from traditional approach because we do not consider only subsumption-based mappings, which are the mappings that most of the techniques for ontology alignment provide [17], but more in general abstraction-based mappings; moreover we adopt a multi-layered integration approach because of the large amount of schema considered. As argued in the paper, nor RDFS or DL-Lite provide provide specific language constructs to model different kinds of integration-abstraction relationships.

Abstractions in conceptual modeling have been studied to support database design [18], database comprehension and schema summarization [19], formal characterizations of generic relationships [14], and, recently, theories of ontology granularity [15]. Abstraction based on *forgetting* has been applied to Web ontologies [20]. As for conceptual database design, abstraction primitives are exploited to refine or abstract conceptual schemata in top-down and bottom-up database design methodologies [18]. As for database comprehension, several papers address the problem of dominating complexity of large schemata by means of schema clustering techniques (see [21], [13] and references therein). Abstraction are exploited also in [19] to make flat conceptual schemata more comprehensible; the conceptual modeling language used in [19] is Object-Role Modeling (ORM), which is more expressive than ER. All the above mentioned approaches do not explicitly define the abstraction relations between the clusters of entities and their abstract representatives in terms of set-theoretic semantics; instead, the abstraction mechanisms are defined in terms of operations carried out on the schemata.

Generic relationships and their semantics in conceptual models are analyzed in [14]; some of these generic relationships, i.e. aggregation, generalization and grouping can be interpreted as or are related to abstraction relations between concepts. The exploitation of abstraction to enhance comprehension of ontologies and conceptual models has been also proposed in [15]. Three main types of abstractions representing three abstraction mechanisms are introduced: (i) the relation is remodeled as a function; (ii) multiple entities and relations fold into a different type of entity; (iii) semantically less relevant entities and relations are deleted. The primitives used in this paper for ER conceptual schemata overlap with the abstraction types discussed in [14],[15] and [20]. Forgetting in CCR schemata is very close to *deletion* in ontologies as defined in [15].

5 Conclusions

In this paper we consider a context where the representation and integration of semantic Web models (or schemata) is exploited to provide large organizations with an integrated view of the information they manage. Discussing a case study in the eGovernment domain and previous works of colleagues, we assume to adopt at the front-end level light weight graph-based Concept-to-Concept Relationship (CCR) representations. We therefore claim that, in the context of schema representation and integration in the large, light-weight semantic Web languages such as RDFS and DL-Lite (i) cannot be used to provide the semantics of individual CCR models and (ii) are not sufficient to provide appropriate semantics for the definition of the loose mappings needed for model integration. We therefore propose a new interpretation of CCR models semantics; moreover, based on the identification of three abstraction mechanisms exploited in the integration process, we define three main classes of integration-abstraction relations and their semantics.

The approach and the translations defined in the paper allow for the reuse of the methodology and the schemata in the repositories described in [10] in a semantic Web framework. Current research is aimed to develop effective and user friendly graphical interface to browse and edit multi-layered repositories of schemata.

References

1. Staab, S., Studer, R.: Handbook on Ontologies (International Handbooks on Information Systems). SpringerVerlag (2004)
2. Noy, N.F.: Semantic integration: a survey of ontology-based approaches. SIGMOD Rec. **33** (2004) 65–70
3. Baumeister, J., Reutelshoefer, J., Puppe, F.: Engineering on the knowledge formalization continuum. In: SemWiki'09: Proceedings of 4th Semantic Wiki workshop. (2009)
4. Paslaru, E., Simperl, B., Tempich, C., Sure, Y.: Ontocom: A cost estimation model for ontology engineering. In: Proceedings of the 5th International Semantic Web Conference ISWC2006. (2006)
5. Hepp, M.: Possible ontologies: How reality constrains the development of relevant ontologies. IEEE Internet Computing **11** (2007) 90–96
6. Simón, A., Ceccaroni, L., Rosete, A.: Generation of OWL ontologies from concept maps in shallow domains. (2007) 259–267
7. Calvanese, D., Lembo, D., Lenzerini, M., Rosati, R.: DL-Lite: Tractable Description Logics for ontologies. In: In Proc. of AAAI 2005. (2005) 602–607
8. Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. Web Semant. **5** (2007) 251–261
9. Ghidini, C., Kump, B., Lindstaedt, S.N., Mahbub, N., Pammer, V., Rospocher, M., Serafini, L.: Moki: The enterprise modelling wiki. In Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E.P.B., eds.: ESWC. Volume 5554 of Lecture Notes in Computer Science., Springer (2009) 831–835
10. Batini, C., Barone, D., Garasi, M., Viscusi, G.: Design and use of ER repositories: Methodologies and experiences in egovernment initiatives. In Embley, D.W., Olivé, A., Ram, S., eds.: ER. Volume 4215 of Lecture Notes in Computer Science., Springer (2006) 399–412
11. Batini, C., Battista, G.D., Santucci, G.: Structuring primitives for a dictionary of entity relationship data schemas. IEEE Trans. Softw. Eng. **19** (1993) 344–365
12. Noy, N.F., Musen, M.A.: Specifying ontology views by traversal. In McIlraith, S.A., Plexousakis, D., van Harmelen, F., eds.: International Semantic Web Conference. Volume 3298 of Lecture Notes in Computer Science., Springer (2004) 713–725
13. Tavana, M., Joglekar, P., Redmond, M.A.: An automated entity-relationship clustering algorithm for conceptual database design. Inf. Syst. **32** (2007) 773–792
14. Dahchour, M., Pirotte, A., Zimányi, E.: Generic relationships in information modeling. J. Data Semantics IV **3730** (2005) 1–34
15. Keet, C.M.: Enhancing comprehension of ontologies and conceptual models through abstractions. In: AI*IA '07: Proceedings of the 10th Congress of the Italian Association for Artificial Intelligence on AI*IA 2007, Berlin, Heidelberg, Springer-Verlag (2007) 813–821
16. Coffey, J.W., Hoffman, R.R., Cañas, A.J.: Concept map-based knowledge modeling: perspectives from information and knowledge visualization. Information Visualization **5** (2006) 192–201
17. Euzenat, J., Shvaiko, P.: Ontology matching. Springer-Verlag, Heidelberg (DE) (2007)
18. Batini, C., Ceri, S., Navathe, S.B.: Conceptual database design: an Entity-relationship approach. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA (1992)
19. Campbell, L.J., Halpin, T.A., Proper, H.A.: Conceptual schemas with abstractions making flat conceptual schemas more comprehensible. Data Knowl. Eng. **20** (1996) 39–85
20. Wang, Z., Wang, K., Topor, R.W., Pan, J.Z.: Forgetting concepts in DL-Lite. In Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M., eds.: ESWC. Volume 5021 of Lecture Notes in Computer Science., Springer (2008) 245–257
21. Sousa, P., de Jesus, L.P., Pereira, G., e Abreu, F.B.: Clustering relations into abstract er schemas for database reverse engineering. Sci. Comput. Program. **45** (2002) 137–153