AER^{IA}: Extending SKOS for the practical, yet well-founded, representation and integration of Web schemas in the large

Matteo Palmonari
Department of Computer Science, Systems and Communication, University of Milan - Bicocca, Milan, Italy
Email: palmonari@disco.unimib.it

Abstract— **Multi-layered repositories of schemas 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 schema 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. In this paper we present a language that aims to achieve a good balance between expressivity and ease of use, namely the binary subset of AER^{IA}, and we define its conceptual syntax, its semantics, and its concrete syntax based on a SKOS extension. The SKOS extension we propose allows for representing light-weight Web schemas to support schema integration and abstraction by (i) extending SKOS with the capability to represent arbitrary relationships between concepts, (ii) extending SKOS with relations representing abstraction-based mappings among concepts, namely, inter-schema generalization, forgetting and collapsing relations. Since the proposed approach is compliant with available tools developed for semantic Web languages, tool support to design multi-layered repositories of Web schemas is provided. The approach is illustrated by means of a case study discussed throughout the paper.**

I. Introduction

One of the challenges that large organizations need to address is to improve the governance of the data they manage [1]. When information sources are multiple (e.g. hundreds of databases) and large in size (e.g. in databases with hundreds/thousands of tables), locating the relevant data, capturing their semantics and grasping an overall view of the available information become difficult; these tasks are indeed crucial in order to use the sources across different applications targeted to data integration, document management, or service provision [1]–[3]. The problem is even more relevant in digital ecosystems, such as large virtual enterprises, and local clusters of Small-Medium Enterprises, where the set of schemas to consider may change over time.

In addition to providing solutions for the management of data at the technical level, *domain experts*, *data architects* and *data managers* need models and tools to manage conceptual-level representations of data sources (e.g. by editing, browsing, and querying such representations), and to understand the relationships among their concepts. In other words, conceptual metadata management is an important building block of data governance, in particular when semantic interoperability need to be taken into account [3].

Two main problems should be addressed by up-to-date metadata management approaches. First, in the contemporary Web-centric and network-based context that organizations need to deal with [1], the explicit representation, reuse and exchange of metadata *through the Web* is crucial; we refer to this issue as *knowledge share and Web compliance*. Second, in order to tackle the information overload, conceptual metadata management initiatives should provide organizations with a comprehensive and semantically - as much as possible - integrated view of the data sources and the knowledge they manage; we refer to this challenge as *conceptual representation and integration*.

The approach we propose to address the above problems is based on multi-layered repositories of light-weight and Web-compliant schemas, *Web schemas* for short, structured according the primitives of abstraction. Data schemas are represented as graphs and are progressively integrated by exploiting abstraction and integration mechanisms; as a result of this integration process, a repository of schemas organized into levels of abstraction is created, where the higher is the abstraction level of a schema, the more information sources the schema represents. The schemas and the mappings between the concepts of different schemas are represented by means of semantic Web languages, which make the schemas Web-compliant and support knowledge sharing. The multi-layered repository provides a comprehensive representation of the information managed by an organization, addressing the problem of conceptual representation and integration.

In this paper we elaborate on and consolidate previous works that addressed specific parts of the above described global picture: structured data dictionaries based on the Extended Entity Relationship (EER) model [4] and organized into levels of abstraction have been introduced in [5] and described in [6]. However these works did not address the problem of making data dictionaries Web compliant,

This paper is based on "Abstract ERIA: a web language for conceptual metadata integration and abstraction in the large", by M. Palmonari and C. Batini, which appeared in the Proceedings of the International ACM Conference on Management of Emergent Digital EcoSystems, Lyon, France, October 27-30, 2009. © 2009 ACM.

which is the main objective of this paper.

Very simple knowledge organization systems such as SKOS [7] allow for the Web-compliant definition of terminologies, namely taxonomies and thesauri, but cannot represent specific relationships between concepts. Semantic Web technologies and languages such as RDF, RDFS and OWL provide knowledge sharing and logical modeling capabilities based on ontologies [8], and techniques to achieve data and schema integration [9]. However, when the goal is to provide domain experts, data architects and data stewards with effective models and tools to manage conceptual metadata, the representation of schemas and concept mappings with Web ontologies rise a significant problem. In fact 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 [10]; on the other hand, as shown in [11], light-weight languages such as RDFS and DL-Lite [12] are not enough expressive to represent Web schemas with the flexibility required in this context, and cannot represent the generic integration-abstraction relations between concepts needed to define the mappings.

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. [13]) can be considered as front-end concrete languages for light-weight ontology design.

The AER^{IA} language has been introduced to preserve some of the advantages of ontology Web languages, such as having a well defined semantics, but keeping the simplicity of lexical knowledge systems such as thesauri [14]. However, the original approach to the definition of AER^{IA} presented the major drawback of introducing yet another semantic Web language (although based on standards such as RDF and OWL-DL at the syntactic level); furthermore, the relationship between lexical knowledge systems such as thesauri and AER^{IA} was has not been previously discussed. In this paper we improve our previous proposal by bridging this gap and representing the binary AER^{IA} subset as a SKOS extension. With binary AER^{IA} we denote the subset of AER^{IA} that considers only binary relationships between concepts, and which basically maps to CCR models, addressing the more interesting language subset to support conceptual metadata management. The SKOS extension introduced in this paper supports the representation of light-weight Web schemas and their integration by means of abstraction primitives. Moreover, by defining AER^{IA} as a SKOS extension, it is possible to improve the schema navigability by exploiting traditional OWL reasoning services.

Furthermore, the above approach is fully compliant with state-of-the-art tools developed in the semantic Web, which provide tool support to design and publish on the Web the multi-layered repositories of Web schemas.

As a results of the proposed approach, SKOS is extended with:

- the capability to represent arbitrary relationships between concepts mapped to SKOS' "related" semantic relations;
- the capability to represent abstraction-based mappings among concepts, by means of inter-schema *generalization*, *forgetting* and *collapsing* relations

Observe that , where SKOS as no set theoretic semantics, the SKOS extension based on AER^{IA} has a formal semantics in terms of translation to OWL ontologies.

The paper is organized as follows: Section II introduces the main problems arising when representing and integrating large sets of schemas and discusses a case study used in the rest of the paper. The overall approach to schema integration based on abstraction, and the notion of multi-layered repository of Web schemas is discussed in Section III. Section IV formally defines the conceptual syntax and the semantics of the AER^{IA} language. The SKOS extension defined to support the representation of schemas and schema mappings and the practical approach to design repositories of schemas based on such extension are presented in Section V. Related works is discussed in Section VI, and conclusions end the paper.

II. Schema Representation and Integration for Conceptual Metadata Management

Schema representation and integration in the large supports the management of conceptual metadata by providing large organizations and networked enterprises with an integrated view of the information managed. Several experiences with structured repositories of ER conceptual schemas related to the most relevant databases of the Italian central public administrations are described in [6]. A Central Public Administration (CPA) repository of has been created to organize the conceptual metadata of a large set of public institutions; the CPA repository progressively integrates through nine levels of abstraction approximately 400 hundreds conceptual schemas, including approximately 5.000 entities and hundreds of relationships, representing at the conceptual level logical of the information sources (*basic schemas*). The benefits of exploiting structured repositories of schemas 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 [6]. 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.

Consider that even dozens of heterogeneous information sources provide a large set of sources considering the critical effort needed in ontology matching and schema integration. However, to give an idea of the approach

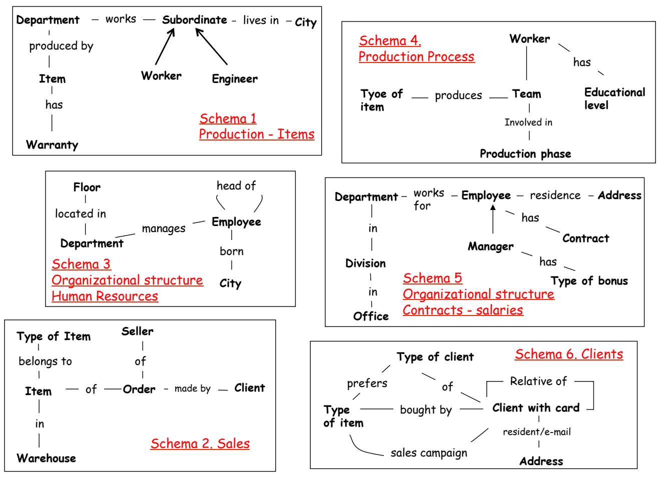


Figure 1: The case study with six source schemas

developed in this paper we present a case study with a even smaller number of sources belonging to a company in the eCommerce domain. We assume that the company has six data sources containing information about: the produced items (Schema 1), the production process (Schema 4), sales (Schema 2) and clients (Schema 6), the organizational structure of the company (Schema 3) and the contracts and the salaries of the employees (Schema 5). Figure 1 represents six conceptual schemas of the mentioned data sources.

Conceptual schemas of the sources are represented by a graphical simplified notation of the Extended Entity-Relationship model [4]: entities are represented by names in bold, and relationships are represented by names in plain text; lines represent the participation of entities to the relationships; arrows represent generalization relations between entities. Observe that these schemas are equivalent to CCR models.

There are a number of interesting issues that emerge from the conceptual schemas represented in Figure 1. First, although the schemas are semantically heterogenous, some schemas share a significant number of concepts with other schemas because they refer to data that are semantically related; as an example, Schema 1 and Schema 4 share similar concepts such *Worker* and *Items* (actually, *Type of Item* in Schema 04. Schemas that are semantically close can be clustered and integrated by exploiting integration and abstraction primitives as defined in [5]; the process is repeated recursively until a global schema at a desired level of abstraction is obtained. The application of this approach to the case study introduced in this section is shown in Figure 2. In the figure, Schema 3 and Schema 5 are integrated into a schema representing the "Organizational structure" of the company; Schema 2 and Schema 6 are integrated into a schema representing the information about "Clients and sales"; Schema 1 and Schema 4 are integrated into a schema representing the information about "Production"; the obtained schemas are then integrated into a unique global schema. Observe that each integrated schema does not represent every concept occurring in the schema it integrates, but provide a representation at an higher level of abstraction.

Second, relationships having the same intuitive semantics - and therefore identified by the same name - can occur more than once in each schema; as an example, consider the relationship "*in*" in Schema 5. We call Multiple Use of Relationship Names (MURN) this feature of conceptual schemas. MURN is not legal for ER models, but is often used in practice (e.g. see the schemas in [16]), and is particularly reasonable when dealing with high-level conceptual representations, and when these representations are used as media to bridge the gap between the data sources and the human experts. If we consider popular tools for conceptual modeling such as the IBM InfoSphere Data Architect[1], these tools allow for the representation of MURN ER schemas, although the application warns the user with a message when he/she uses a relation label to connect more than two concepts.

Third, some concepts are similar across the schemas but not straightforwardly linkable through subsumption-based relations (subclass or generalization); as an example, consider *Item* of Schema 1 and *Type of Items* of Schema 4, or *Address* of Schema 6 and *City* of Schema 3. The deep semantic integration of these schemas based on concept mappings grounded on subsumption is possible, but might require very demanding modeling efforts. At large scale, the mappings' precision can be given up to support the semantic integration of many schemas considering looser mappings among concepts.

In the following, we refer to the above three issues respectively as *progressive integration by abstraction*, *MURN compliance*, and *loose mapping support*.

III. Integration by Abstraction: Basic Principles

In our approach, abstraction acts as a key driver to manage the complexity of large and heterogeneous collection of schemas. Among the issues introduced in the previous section, *progressive integration by abstraction* deals with the approach adopted to integrate a large collection of schemas, while *MURN compliance* and *loose mapping support* concern the language adopted to represent the schemas and to integrate them. We therefore identify two main and interrelated principles adopted to support schema integration by means of abstraction representations and primitives:

- the construction of repositories of schemas at different levels of abstraction;
- the adoption a light-weight approach to Web schema representation based on AER^{IA}, a language supporting MURN schemas and loose mappings.

In this section we better characterize the notion of progressive integration by abstraction on the Web, by describing multi-layered repositories of schemas and their

[1]http://www-01.ibm.com/software/data/optim/data-architect/features.html?S_CMP=wspace

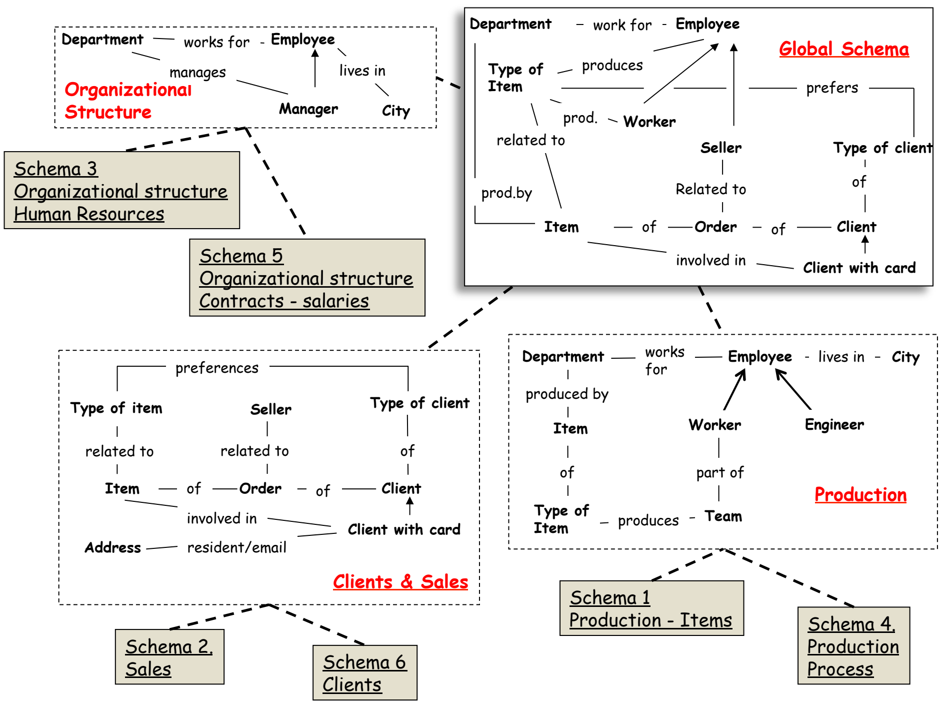


Figure 2: The progressive integration of the six source schemas

structure, and we provide an overview of the approach to the representation of Web schemas and schema mappings.

A. *Multi-layered Repositories of Web Schemas*

The structure of a repository of Web schemas can be described by defining by the notion of schema integration-abstraction framework.

Given a set of Σ, a Schema Integration-Abstraction Framework can be formalized by introducing a schema integration-abstraction function $f^{\Sigma} : \mathcal{P}(\Sigma) \longmapsto \Sigma$

Definition 1: **Schema Integration-Abstraction Framework** Given a set of schemas Σ, and a schema integration-abstraction function f^{Σ}, the *Schema Integration-Abstraction Framework* $\mathcal{F}^{\Sigma}$ for f^{Σ} (SIAF, for short) is the graph of f^{Σ}. When $f^{\Sigma}(\Delta) = T$ holds for some $\Delta \subseteq \Sigma$, we call: *integration-abstraction operation* (IA operation, for short) the application of f^{Σ} to Δ, *source schemas* every schema $S \in \Delta$, and *target schema* the schema T. We call *schema-level integration-abstraction relation*, a binary relation between schemas $R^{SIA} \subseteq (\Sigma \times \Sigma)$, where $R^{SIA}(S, T)$ holds if and only if there exists some $\Gamma \in \Sigma$ such that $f^{\Sigma}(\Gamma) = T$ and $S \in \Gamma$.

We call *base schema* every schema S such that there does not exist any T such that $R^{SIA}(S, T)$ holds. We call *upper schema* every schema S such that there not exist any T such that $R^{SIA}(S, T)$ holds. We call *top schema* an upper schema which is unique for $\mathcal{F}f^{\Sigma}$, that is such that there does not exist any schema T with $T \neq S$, which is an upper schema for $\mathcal{F}f^{\Sigma}$.

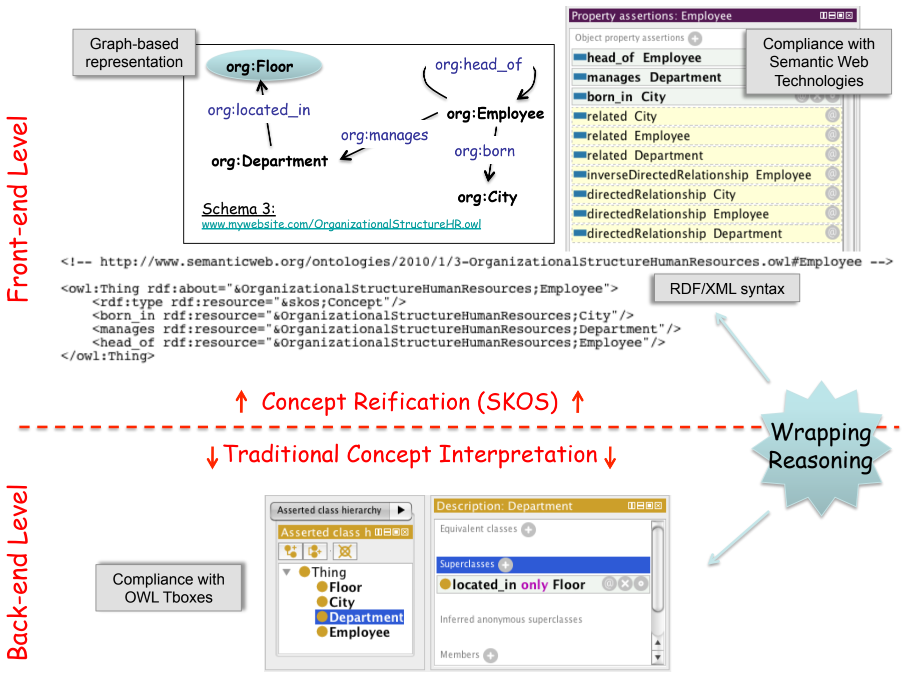
The following properties of SIAFS can be defined. We call *partitive SIAF*, a SIAF where each schema is integrated-abstracted in only one schema, that is, where if $R^{SIA}(S, T)$ and $R^{SIA}(S, T')$, then $T = T'$. Given a schema S, and a partitive SIAF $\mathcal{F}f^{\Sigma}$ we call *upward integration-abstraction path* for S a sequence of schemas $S_0, ..., S_n$ such that $S_0 = S$, S_n is an upper schema for $\mathcal{F}f^{\Sigma}$, and for every S_i with $0 \leqslant i \leqslant n$, $R^{SIA}(S_i, S_{i+1})$ holds. We call *structurally balanced SIAF* a SIAF such that the upward integration-abstraction path for every base schema S has a same length l. We call *complete SIAF* a SIAF which has a top schema.

We call repository of Web schemas, or repository of schemas for short, any collection of schemas that instantiate a schema integration-abstraction framework. We call multi-layered repositories of schemas, or structured repositories of schemas, a repository of schemas based on a partitive SIAF.

At the core of the construction of a repositories of Web schemas there is the capability of representing individual schemas, and the relations between source and target schemas of IA operations. An approach to schema integration based on repositories of Web schema adopts therefore the well-known model usually adopted in virtual data integration, where a set of schemas are integrated into a global view by defining mappings between local sources and the global view [9]. These mappings are defined by making the relationship between elements of the source and the target schemas explicit.

B. *The AER^{IA} Approach*

The AER^{IA} language provides light-weight modeling primitives to represent Web schemas and loose schema mappings. Light-weight semantic Web languages such as RDFS and DL-Lite (equivalent to the OWL2 QL


Graph-based representation
org:Floor
org:located_in
org:head_of
org:Employee
org:manages
org:Department
org:born
org:City
Schema 3:
www.mywebsite.com/OrganizationalStructureHR.owl
Property assertions: Employee
Object property assertions
head_of Employee
manages Department
born_in City
related City
related Employee
related Department
inverseDirectedRelationship Employee
directedRelationship City
directedRelationship Employee
directedRelationship Department
Compliance with Semantic Web Technologies
RDF/XML syntax
Front-end Level
Concept Reification (SKOS)
Traditional Concept Interpretation
Wrapping Reasoning
Back-end Level
Compliance with OWL Tboxes
Asserted class hierarchy
Thing
Floor
City
Department
Employee
Description: Department
Equivalent classes
Superclasses
located_in only Floor
Inferred anonymous superclasses
Members

```
<!-- http://www.semanticweb.org/ontologies/2010/1/3-OrganizationalStructureHumanResources.owl#Employee -->

<owl:Thing rdf:about="&OrganizationalStructureHumanResources;Employee">
    <rdf:type rdf:resource="&skos;Concept"/>
    <born_in rdf:resource="&OrganizationalStructureHumanResources;City"/>
    <manages rdf:resource="&OrganizationalStructureHumanResources;Department"/>
    <head_of rdf:resource="&OrganizationalStructureHumanResources;Employee"/>
</owl:Thing>
```

Figure 3: The approach to represent light-weight Web schemas with the SKOS extension based on AER^{IA}

profile[2]) cannot provide MURN compliance and loose mapping support because of their conjunctive semantics for role typing (see Section VI for a in-depth discussion of this topic). The approach to the representation of Web Schemas and of loose mappings between concepts of different schemas is based on decoupling the front-end and the back-end level, and on exploiting abstraction-based mappings, thus extending the traditional notion of subsumption-based mappings.

The front-end level is the level at which users are expected to consider when they design the repository. At this level, Web schemas can be represented as Web compliant graphs, where every resource is identified by a URI. In order to design the Web schemas, end users exploit an extension of SKOS; although SKOS is an OWL-DL ontology, concepts are considered instances of such an ontology (concept reification); the design of the repository therefore reduces to the assertion of Abox statements, i.e. relations between OWL instances.

The back-end level provides a formal interpretation of the Web schemas in terms of OWL-DL ontologies, where concepts are interpreted as ontology concepts, and schema constraints are interpreted as Tbox axioms; however, end users do not perceive this level and are not required to master the logical complexity intrinsic to Tbox-level axioms; since the translation is deterministic AER^{IA} schemas can be exported as Web ontologies compliant with the OWL-DL model. Figure 3 show the relationships between the front-end and the back-end level describing how a Web schema looks like at each level.

Abstraction-based mappings represent the relations between concepts that result from the application of abstraction primitives to a schema (abstraction by generalization, collapsing and forgetting); these mappings are used for the loose integration of Web schemas.

In the following we introduce the AER^{IA} language conceptual syntax and semantics, and we provide a SKOS extension to concretely represent AER^{IA} schemas on the semantic Web.

IV. The AER^{IA} Language

The AER^{IA} language provides primitives to represents schema concepts - called entities in the language - and their mutual relationships, attributes of the concepts, generalization relations, and inverse relations. As for expressivity, AER^{IA} can be considered a formalism to represent abstract binary EER schemas, where: $[0, n]$ is considered as default cardinality restriction; relationships are assumed to be directed; MURN schemas are allowed.

The AER^{IA} language is defined by a conceptual syntax and a semantics semantics based on the translation of $\mathcal{L}^{AER^{IA}}$ schemas to OWL ontologies.

A. AER^{IA} Conceptual Syntax

Formally, the AER^{IA} syntax is defined as follows.

Definition 2: **Abstract Entity Relationship with Integration-Abstraction** (AER^{IA}) **Alphabet**. An AER^{IA} alphabet $\mathcal{A} = (\Sigma, E, A, R, gen, inv, att, IA)$, is a tuple where: Σ is a set of schema names, E is a set of entity names, A is a set of attributes, R is a set of binary relationship names, gen, inv and att are respectively the generalization relation, the inverse relation, and the attribute relation symbols, IA is a set of names of integration-abstraction relations, and the sets $\Sigma, E, R, gen, inv, att, IA$ are pairwise disjoint.

[2]http://www.w3.org/TR/owl2-profiles/

Definition 3: AER^{IA} **language and** AER^{IA} **Schema.** Given a AER^{IA} alphabet $\mathcal{A} = (\Sigma, E, R, gen, inv, att, IA)$, a AER^{IA} language $\mathcal{L}^{AER^{IA}}$ based on $\mathcal{A}$ is the set of sentences having the form:

- intra-schema $\mathcal{L}^{AER^{IA}}$ sentences
  - $S{:}r(S{:}e, S{:}f)$;
  - $gen(S{:}e, S{:}f)$;
  - $inv(S{:}p, S{:}r)$;
  - $att(S{:}e, S{:}a)$;
- inter-schema $\mathcal{L}^{AER^{IA}}$ sentences
  - $ia*(S{:}e, S'{:}f)$

where $S \in \Sigma$, $\{p, r\} \in R$,$\{e, f\} \in E$, $a \in A$, and $ia* \in IA$. The *attribute locality condition* holds on a set of $\mathcal{L}^{AER^{IA}}$ sentences iff there does not exist any a such that $att(S{:}e, S{:}a)$ and $att(S{:}f, S{:}a)$, with $e \neq f$ (attributes are local to entities). Given a language $\mathcal{L}^{AER^{IA}}$ defined over an alphabet $\mathcal{A}$, an AER^{IA} schema S is a set of intra-schema sentences $\Phi \subseteq \mathcal{L}^{AER^{IA}}$ on which the attribute locality condition holds. Statements having the form $S{:}r(S{:}e, S{:}f)$ are called AER^{IA} *patterns*; a AER^{IA} pattern whose relation is r is called AER^{IA} r-pattern.

When the schema that intra-schema $\mathcal{L}^{AER^{IA}}$ sentences refer to is clear from the context, or not relevant, the more compact notation $r(e, f)$ will be used to denote AER^{IA} patterns, omitting the schema reference.

As an example of schema representation, consider Schema 3 in Figure 1. Let $S03$ be the name of Schema 3; the latter is conceptually represented in the AER^{IA} language by means of the following statements: $S03{:}located\_in(S03{:}Department, S03{:}Floor)$, $S03{:}head\_of(S03{:}Employee, S03{:}Employee)$, $S03{:}manages(S03{:}Employee, S03{:}Department)$, $S03{:}born\_in(S03{:}Employee, S03{:}City)$. The schema can then be enriched by stating that a relation is the inverse of another one: as an example the relation $S03{:}managedBy$ is introduced and defined as the inverse of $S03{:}manages$ with the following statement: $inv(S03{:}manages, S03{:}managedBy)$.

An example of generalization is shown in Schema 05 of Figure 1. In this schema $Manager$ is generalized in $Employee$, which is represented in AER^{IA} by the triple: $gen(S05{:}Employee, S03{:}Manager)$ (represented in the figure by a dashed arrow from the more specific to the more general concept). In this schema there are also two examples of MURN: the relationships has and in occur in two AER patterns each. Attributes are not considered in the examples for sake of clarity.

In the approach based on AER^{IA}, mappings represent the relations between two concepts as the result of the application of three main abstraction mechanisms acknowledged in the literature (see Section VI for details) and exemplified in Figure 4: *abstraction by generalization*, when an entity or a generalization hierarchy is abstracted into another entity; *abstraction by forgetting*, when a group of entities is abstracted into an entity, in a way such that the new entity represents an element of the group while the other entities are discarded; *abstraction by collapsing*, when one or more entities are collapsed into a more abstract entity which acts as a representative for the group. Figure 4) represents the mapping between Schema 05 (S05) and the more abstract schema, namely, Human Resources Global Schema (HRG), in which S06 and S03 hare integrated (depicted in Figure 3).

Based on the analysis of these abstraction mechanisms, AER^{IA} introduces the three relations to define loose mappings between concepts:

1) $abstract - by - generalization$ ($a\text{-}generalize$ for short), and the inverse $abstracted - by - generalization$ ($a\text{-}generalizedIn$ for short). This relation represents generalizations between sets of entities of different schemas with standard subsumption semantics; as an example, a generalization-based mapping from Figure 4 is represented by the triple $a\text{-}generalize(HRG{:}Manager, Sal{:}Manager)$.
2) $abstract - by - forgetting$ ($a\text{-}forget$ for short), and the inverse $a\text{-}forgotIn$. It represents abstractions of source entities that are "sunk" in a more abstract target entity, discarding some details in the source representations; as an example, a forgetting-based mapping from Figure 4 is represented by the triples $a\text{-}forget(HRG{:}Department, Sal{:}Department)$, $a\text{-}forget(HRG{:}Department, Sal{:}Division)$, and $a\text{-}forget(HRG{:}Department, Sal{:}Office)$.
3) $abstract - by - collapsing$ ($a\text{-}collapse$ for short), and the inverse $a\text{-}collapsedIn$. It represents abstraction mechanisms in which the target concept has a different meaning w.r.t. all the source concepts; as an example, a collapse mapping is represented by the triple $a\text{-}collapse(HRG{:}City, Sal{:}Address)$ as shown in Figure 4.

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 $Department$ of HRG $a\text{-}generalize$ the entity $Department$ of Sal, which is represented by the sentence $a\text{-}generalize(HRG{:}Department, Sal{:}Department)$. The intuitive meaning of $a\text{-}collapse$ includes "part of"-like aggregations and the grouping relations introduced in [17], and is almost equivalent to unfolding relations as introduced in [18]; in general, collapsing is based on metonimy: usually a concept representing a whole collapses concepts representing parts, but the opposite is also possible (under metonimy, a part can represent the whole, as in the above example, where city represents an address).

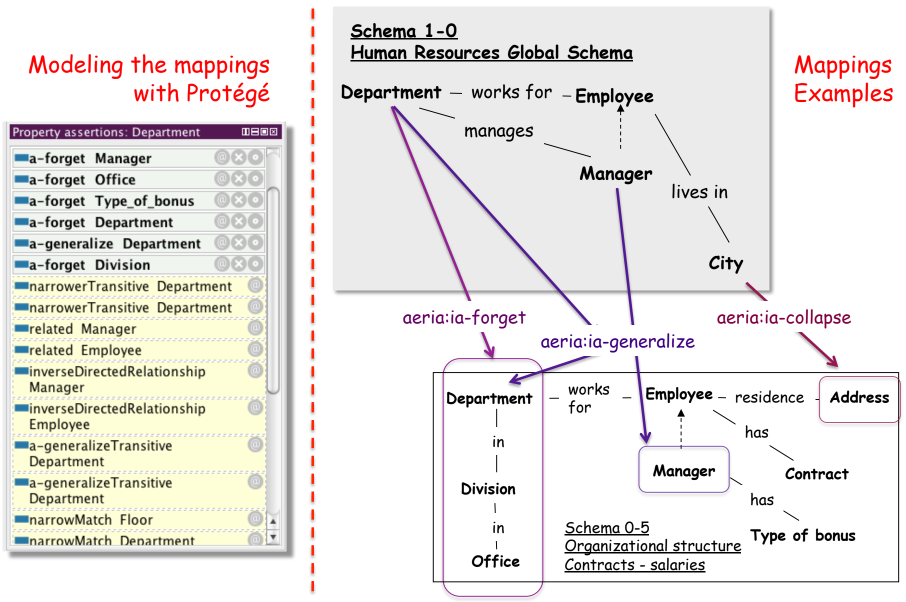


Figure 4: Examples of inter-schema mappings with the AER^{IA} constructs

	$\mathcal{L}^{AERIA}$		$\mathcal{SHOIQ}^{\mathcal{D}}$ (OWL-DL)	Intuitive Semantics
1	e	$\mapsto$	C^e	concept
2	r	$\mapsto$	P^r	property
3	$gen(e,f)$	$\mapsto$	$C^e \sqsubseteq C^f$	generalization
4	$inv(p,r)$	$\mapsto$	$P^p \equiv P^{r-}$	inverse property
5	$r(\{e_1,...,e_h\},$	$\mapsto$	$\exists R.\top \sqsubseteq C^{e_1} \sqcup ... \sqcup C^{e_h}$,	global domain union
6	$\{f_1,...,f_k\})$		$\exists R^-.\top \sqsubseteq C^{f_1} \sqcup ... \sqcup C^{f_k}$	global range union
7	$r(\{e_1,...,e_k\},f)$	$\mapsto$	$C^f \sqsubseteq \forall R^-.(C^{e_1} \sqcup ... \sqcup C^{e_k})$,	domain conditional union
8	$r(e,\{f_1,...,f_k\})$	$\mapsto$	$C^e \sqsubseteq \forall R.(C^{f_1} \sqcup ... \sqcup C^{f_k})$;	range conditional union
9	a	$\mapsto$	P^a	attribute
10	$att(e,a)$	$\mapsto$	$\exists P^a \sqsubseteq C^e$	attribute domain

TABLE I.: Translation from binary AER^{IA} schemas to $OWL-DL$ ontologies

B. AER^{IA} Semantics

Providing an account for MURN schemas is one of the main challenges In order to define the semantics of AER^{IA}, . 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 S05 of Figure 4 cannot be represented and specific different names for each of the involved relationship need to be introduced (e.g. $in\#1$, $in\#2$, etc.). The capability to represent MURN schemas is important when dealing with large knowledge bases and gross-grained representations, and is used in practice, even breaking up formalisms semantics; in fact, SURN lead to unnecessary (in these contexts) multiplication of relationship names, in particular when representing generic relationships between concept such as location, relatedness and relationships such as $partof$, has, and so on.

The formal semantics for MURN schemas is based on the translation of AER^{IA} schemas into OWL-DL ontologies. Translation rules for $\mathcal{L}^{AER^{IA}}$ intra-schema sentences are represented in Table I. In the following we exploit the DL notation defined in [8] for OWL and RDFS constructs, respectively based on $\mathcal{SHOIQ}^{\mathcal{D}}$ and [12] $(DL-Lite)$; 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 and the range of the DL role R, where R^- denotes the inverse of R; the only exceptions to the standard notation is the more compact first-order-like form $r(x,y)$, instead of $< x, y >: r$ for Abox relational assertions. Moreover, in the table we adopt the following compact notation: $r(e,\{f_1,...,f_k\})$ represents the set of $\mathcal{L}^{AER^{IA}}$ assertions where e occurs as a first element in a AER^{IA} r-pattern (see Def. 3). $r(\{e_1,...,e_h\},f)$ represents the set of AER^{IA} r-patterns where f occurs as second element in the pattern, and $r(\{e_1,...,e_h\},\{f_1,...,f_k\})$ represent the set of all the AER^{IA} r-patterns where one element of the first set occurs as first element, and one element in the second set occurs as second element.

The translation rules represented in Table I capture the following assumptions (examples are based on Schema 3 of Figure 1):

- **Concepts, relations, generalization and inverse properties**. Concepts are represented by OWL-DL concepts, relations are represented by OWL-DL properties, and the generalization relation is interpreted as the subsumption relation between concepts (rows 1-3 of Table I); e.g. the OWL-DL concepts

$Floor$, $City$, $Dep.$ and $Emp.$ represent the concepts of Schema 3 depicted in in Figure 1; inverse properties are defined by axioms as usual in OWL-DL (row 4).

- **Global domain/range union semantics**. AER^{IA} patterns specify all the possible domains (ranges) for the relations: an AER^{IA} pattern $r(c, d)$ specifies that c and d are respectively possible domain (range) for r (rows 5-6). The domains/ranges specified in a schema are the only possible ones for such relation, that is, the domain (range) of a relationship consists of the union of the all the possible domains/ranges. In schemas that are SURN schemas such as Schema 1 in Figure 1, this reduces to plain RDFS domain/range restrictions; instead, in a MURN schema such as Schema 5 of Figure 1, the domain of in consists of $Dep. \sqcup Division$, its range consists of $Division \sqcup Office$.
- **Conditional semantics**. Additional constraints capture the conditional semantics specified by AER^{IA} patterns. A schema specifies all the possible ranges (domains) for a relation r given a specific domain (range), by exploiting universal qualified domain/range restrictions of OWL (rows 7-8): e.g. the additional constraint $Dep. \sqsubseteq \forall located\_in.Floor$ is introduced in Schema 5 refining the global domain/range union semantics.
- **Attributes**. Attributes are interpreted as properties of OWL concepts (row 9); when a concept has an attribute, it is the domain of the property representing the attribute (observe that attributes are considered local to concepts, and therefore there is no possibility that more than one domain is specified for an attribute (row 10).

Despite the recent attention that approximate schema mapping models has received in the research community [19], mappings in data integration are traditionally interpreted according to subsumption-based semantics (subconcept or equivalent concept relations) [9]. The AER^{IA} language extends the notion of schema mapping by modeling concept mappings based also on abstraction relations whose semantics cannot be reduced to subsumption (forgetting and collapsing).

Therefore, the semantics for integration-abstraction relations, i.e. for $\mathcal{L}^{AER^{IA}}$ inter-schema sentences, follows the principles introduced for intra-schema AER^{IA} sentences, and is formally described in Table II; in particular, abstraction by generalization relations are translated into subsumption statements, while other relations are treated as directed relationships; however, since a set of source concepts can be mapped only to one target concept, i.e. it cannot happen that $a\text{-}forgot(e, f)$ and $a\text{-}forgot(e, f')$ with $f \neq f'$, statements with $a\text{-}forgotIn$ and $a\text{-}collapsedIn$ can be safely tranlated in OWL by qualified universal restrictions (second and third rows of Table II).

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

TABLE II.: Semantics for $\mathcal{L}^{CCR}$ ia-relations

V. Concrete Representation of AER^{IA} by Extending SKOS

A. Extending SKOS to Represent Web Schemas

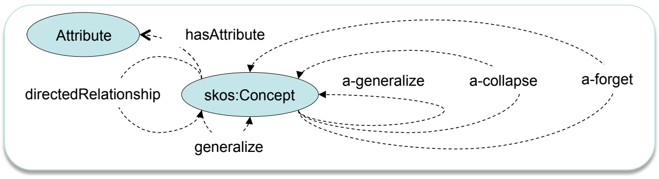


Figure 5: The $AER^{IA-SKOS}$ ontology

The concrete Web-compliant syntax to represent AER^{IA} schemas and AER^{IA} schema mappings is based on a SKOS extension, namely the $AER^{IA-SKOS}$ ontology whose core elements are represented in Figure 5. In the following we discuss such extension and we show how $AER^{IA-SKOS}$ schemas, i.e. AER^{IA} schemas represented extending SKOS, can be practically designed and linked together in multi-layered repositories of schemas.

The concrete representation of AER^{IA} is based on the definition of an extension of the SKOS core ontology [3] defined as follows:

- AER^{IA} entities are represented by the concept $skos{:}concept$
- the concept $Attribute$ is introduced to represent schema attributes
- the properties $directedRelationship$, $hasAttribute$, and $generalize$, with a respective inverse property each are introduced and are characterized as shown in Table III;
- the property $generalize$ in introduced as subproperty of $skos{:}narrower$

The properties $directedRelationship$ and its inverse are subproperties of $skos{:}related$; $generalize$ is subproperty of $skos{:}narrower$; instead, $hasAttribute$ is independent from SKOS. Domain specific relations between concepts are introduced as subproperties of $directedRelationship$. AER patterns are therefore represented as triples $< C, R, D >$ where (1) C and D are instances of $skos{:}concept$ and R is a subproperty of $directedRelationship$, or (2) C and D are instances of $skos{:}concept$ and R is a $generalize$, or (3) C is a $skos{:}concept$, D is an $Attribute$, and R is $hasAttribute$.

The above described approach, where AER^{IA} core relations are considered subproperties of SKOS proper-

[3]http://www.w3.org/2004/02/skos/core/owl-dl/skos-core-owl-dl.owl

aeria property	Domain	Range	superproperty	inverse property
directedRelationship	Concept	Concept	skos:related	inverseDirectedRelationship
hasAttribute	Concept	Attribute	no	attributeOf
generalize	Concept	Concept	skos:narrower	generalizedIn

TABLE III.: Mappings to SKOS

ties, allows to exploit standard reasoning services to infer SKOS terminologies from each schema of the repository. On the other hand, SKOS properties and annotations can be used to enrich the representation of schemas with semantic relations (e.g. synonimity) and textual descriptions (e.g. glosses).

B. Extending SKOS to Represent Schema Mappings

The AER^{IA} abstraction-based mappings are compliant with the semantic of narrower/broader relations as they are usually interpreted in thesauri and in SKOS terminologies. In fact narrower/broader relations in thesauri account for semantic relations between two entities such as narrower/broader generic (the relation between a concept and its superconcept), narrower/broader partitive (the relation between a concept-part and a concept-whole), and narrower/broader instiantive (the relation between an instance and the concept the instance is member of). Although the relations representing the three types of narrower/broader relations are deprecated in the current version of SKOS, it is clear that SKOS introduces the capability of representing mappings that cannot be interpreted according to the subsumption semantics.

SKOS introduces two particular relations to represent narrower/broader relations between mapped concepts, namely $skos{:}narrowMatch$ and its inverse $skos{:}broadMatch$.The three AER^{IA} integration-abstraction relations are therefore introduced as subproperties of $skos{:}narrowMatch$ as shown in Table IV.

aeria property	skos superproperty	inverse property
a-generalize	narrowMatch	a-generalizedIn
a-forget	narrowMatch	a-forgetIn
a-collapse	narrowMatch	a-collapsedIn

TABLE IV.: Mappings to SKOS

The above described approach allows to exploit standard reasoning services to infer SKOS terminological relations based on the defined mappings; such inferences are shown by the elements on yellow background in the screenshot from the Protégé[4] ontology editor depicted in the left-hand side of Figure 4.

C. Web Schema Design and Integration in Practice

The representation of Web schemas through the above described SKOS extension allows to exploit existent tools like Protégé for designing individual Web schemas and multi-layered-repositories. A repository of schemas can be designed instantiating an SIAF according to the following basic principles:

- the AER^{IA} ontology imports the SKOS ontology;
- each schema is represented by an ontology and imports the AER^{IA} ontology;
- schema entities are introduced as instances of *skos:concept*; schema relationships are introduced as subproperties of AER^{IA} directed relationships;
- when a set of schemas Σ are integrated into a more abstract schema A; each schema in Σ is imported by A; statements representing schema mappings based on the AER^{IA} ia-relations are inserted in A.

The approach based on SKOS allows also to exploit reasoning services supported by state-of-the-art OWL-DL reasoners such as Pellet[5] to effectively compute implicit semantic relations between concepts, even at the front-end level. In particular, the transitive closure of generalization relations and generalization mappings, inverse relations, and SKOS superproperties can be computed.

Observe that the SKOS extension proposed is autonomous w.r.t. the approach based on multi-layered repositories; in fact, it provides SKOS with the capability of representing arbitrary labeled relations between schemas, and forgetting and collapsing-based schema mappings. Nonetheless this is achieved by a small extension.

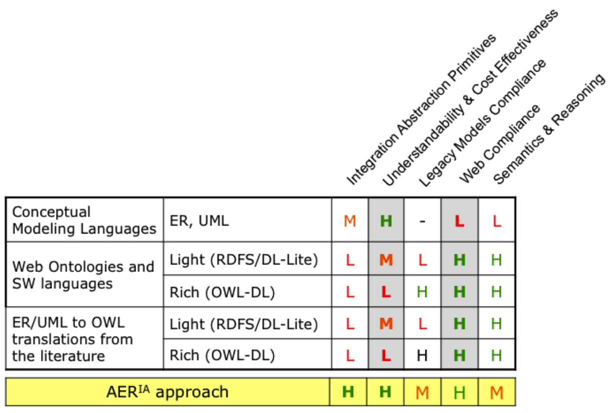
The approach to schema representation and integration in the large with the $AER^{IA-SKOS}$ language is practical, yet well-founded. The approach is *practical* for the following reasons: it builds on a well-known and widely adopted semantic Web language like SKOS; relations between concepts are defined through AER^{IA} patterns, and schema editing reduces to stating RDF triples; querying schemas and schema repositories based on $AER^{IA-SKOS}$ can be performed by means of SPARQL queries. The approach is yet well-founded because the language introduced has a well-defined semantics based on a standard such as OWL-DL, which provide a sound interpretation also for MURN schemas.

VI. Related Works

Different approaches to metadata management specifically addressing conceptual metadata have been proposed in the literature: remarkably, a system to manage conceptual models represented in different languages, among which UML and ER, and Web ontologies represented in RDFS and OWL-DL, is presented in [3]; a preliminary framework based on a network of models is introduced in [20]. In this paper, we exploit [5] as reference metadata

[4]http://protege.stanford.edu/

[5]http://clarkparsia.com/pellet/



		Integration Abstraction Primitives	Understandability & Cost Effectiveness	Legacy Models Compliance	Web Compliance	Semantics & Reasoning
Conceptual Modeling Languages	ER, UML	M	H	-	L	L
Web Ontologies and SW languages	Light (RDFS/DL-Lite)	L	M	L	H	H
	Rich (OWL-DL)	L	L	H	H	H
ER/UML to OWL translations from the literature	Light (RDFS/DL-Lite)	L	M	L	H	H
	Rich (OWL-DL)	L	L	H	H	H
AER^{IA} approach		H	H	M	H	M

Figure 6: A comparison of the AER^{IA} approaches with other approaches proposed in the literature

management framework because it is the only one that explicitly addresses the problem of schema integration and abstraction. The latter paper presents also a methodology for the design of multi-layered repositories of schemas that has been applied for several years several times to integrate large sets of schemas [6] and that can be adopted also for repositories of Web schemas as defined in this paper.

In the following we compare the main semantic Web and conceptual modeling languages available in the literature for conceptual schema representation and integration according to five comparison criteria:

- the capability of the languages to explicitly represent abstraction relationships needed in a schema integration process (Criterion 1);
- the understandability of the languages in communities of professionals with little formal background, and the cost-effectiveness when large sets of schemas and concepts need to be considered (Criterion 2);
- the capability to reuse legacy conceptual schemas based on languages like ER and UML (Criterion 3):
- the capability to support model reuse and exchange through the Web by relying on accepted semantic Web standards (Criterion 4);
- the support to reasoning based on formal and well defined semantics (Criterion 5).

Results of the comparison are summarized in Figure 6. In the following we provide a detailed analysis by emphasizing the the positive (+) and negative (-) position of these solutions w.r.t. to the above introduced criteria.

Semantic Web languages such as RDFS and OWL have the advantage of coupling Web compliance (+ Criterion 4) and semantics (+ Criterion 5), and are supported by a set of Web-based technologies and tools. However they also have some drawbacks: expressiveness influences a variety of costs discussed in ontology engineering [21]; Tbox-level axioms makes it difficult for people with little background in logic and knowledge representation to master OWL semantics and use it in the large in a cost-effective way [10] (- Criterion 2). RDFS is simpler and easier than OWL, as proved by the number of RDFS ontologies actually published on the Web [10]. Modeling relationships with many non overlapping concepts as range - or domain - (e.g. resources are used by services *and* actors) conflicts with the RDF conjunctive interpretation of multiple range/domain restrictions; [22] showed that a further consequence of this problem is that integration-abstraction relations that do not reduce to subsumption cannot be represented in RDFS (the same argument applies to OWL fragments corresponding to DL-Lite) (- Criterion 1).

Conceptual modeling languages such as ER and UML are well known in the professional community (+ Criterion 2) [23], but are not natively Web compliant (- Criterion 4) (this argument applies also to the Telos language [24]), or their semantics is not formal (UML) or not exploitable to perform reasoning (ER) (- Criterion 5). However, one might choose a conceptual modeling language such as ER at the front-end level and then exploit translations to semantic Web languages. A number of translations from ER to DLs and OWL have been proposed [4], [12], [25], [26]. A first approach translates ER in description logics (DL) such as $\mathcal{SHOIQ}^{\mathcal{D}}$ or extended $DL\!-\!Lite$ (corresponding respectively to OWL-DL or subsets) and is based on (ii) the representation of ER relationships as concepts (OWL classes), and (ii) the introduction of ER-roles, represented as DL-roles (OWL properties), to represent the participation of entities to relationships [25], [26], [4]; we will call this approach the *relationship reification approach* because relationships are reified into concepts (and hence objects at the extensional level). A second approach translates ER diagrams in $DL-Lite$ knowledge bases [12], is based on the representation of relationships as DL roles (OWL properties), and will be called the *relationship-as-role approach*.

W.r.t. the specific problem context we are addressing, the relationship-as-role approach has the main drawback of producing rather complex ontologies from ER schemas; the number of ontological entities are significantly multiplied, generating in particular a large amount of properties with little intuitive semantics to represent ER roles (linkages between ER entities and ER relationships).

The relationship-as-role approach based on DL-Lite provides more compact and intuitive representations, but covers ER dialects that are by far less expressive and cannot represent MURN schemas.

CCR models largely overlap with simple semantic nets whose nodes represent concepts (and not instances) and are clearly covered by the AER^{IA} language. CCR models are isomorphic to Concept Maps [27] and almost equals RDFS (if we assume not to consider property hierarchies, not relevant to the claim of the paper); moreover, tools like Semantic MediaWiki [28] and MoKi [13], 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 [28], which means that, in theory, only

SURN can be represented. The interpretation of CCR patterns in MoKi is not clear from [13]; there are reason to believe that their interpretation is based on qualified existential range restriction.

Coming to conceptual schema integration, none of the above solutions provides specific language constructs to model the different kinds of integration-abstraction relations (- Criterion 1) that this paper introduces; ER and UML support only intra-schema abstraction-related relations such as generalization and is-a. Description logics, which can be taken ad the reference formal languages that ontology-driven data integration techniques are based on [29], provides as built-in primitives to straight represent relations between concepts, only subsumption relations. Other mappings based on weaker abstraction mechanisms not reducible to subsumption need to be represented by means of Tbox axioms.

The approach to schema integration in the large based on integration-abstraction primitives is based on the approach introduced in [5]. However, that approach was based on Entity Relationship , 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 are mapped to the concepts of a global schema [9]. 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 [30], 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 [31], database comprehension and schema summarization [32], formal characterizations of generic relationships [17], and, recently, theories of ontology granularity [18]. Abstraction based on *forgetting* has been applied to Web ontologies [33]. As for conceptual database design, abstraction primitives are exploited to refine or abstract conceptual schemas in top-down and bottom-up database design methodologies [31]. As for database comprehension, several papers address the problem of dominating complexity of large schemas by means of schema clustering techniques (see [16], [34] and [23]). Abstraction are exploited also in [32] to make flat conceptual schemas more comprehensible; the conceptual modeling language used in [32] 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 schemas.

Generic relationships and their semantics in conceptual models are analyzed in [17]; 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 [18]. 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 overlap with the abstraction types discussed in [17], [18] and [33]. Forgetting in CCR is very close to *deletion* in ontologies as defined in [18].

VII. Conclusions

Multi-layered repositories of schemas, based on the Web-compliant representation and integration of conceptual models provide organizations dealing with a large amount of data sources with an integrated view on the information managed. In this paper we presented an extension of SKOS to represent multi-layered repositories of schemas supporting both Web-compliance and conceptual integration of the information source.

The proposed approach provides the AER^{IA} language with a Web-compliant concrete syntax based on semantic Web languages to support the design of compact schemas and of asbtraction-based concept mappings at the front-end level, hiding thus logical details that often prevented semantic Web languages from being used by people with limited formal background. Moreover, since the syntax is based on a OWL-DL ontology, automatic reasoning techniques support inference of new relationships and model completion. However, at the back-end level, the language is provided with a set theoretic semantics that make the modeling choices explicit and can be exploited to derive Web ontologies based on the designed conceptual schemas.

Current research focuses on the study of automated techniques to support the process of schema integration by abstraction, in order to support the user in the design of the integration process. Furthermore, techniques to extract AER^{IA} schemas from other conceptual model languages and format are under investigation.

References

[1] Kumar, S.: Data governance: An approach to effective data management. White paper, Satyam Computer Services, Ltd. (2008)

[2] SAS: The value of integrated metadata: Sas®open metadata architecture. Technical report, SAS, SAS Campus Drive, Cary, NC 27513 USA (2004)

[3] Hauch, R., Miller, A., Cardwell, R.: Information intelligence: metadata for information discovery, access, and integration. In: SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM (2005) 793–798

[4] Artale, A., Calvanese, D., Kontchakov, R., Ryzhikov, V., Zakharyaschev, M.: Reasoning over extended er models. In: Proc. of the 26th Int. Conf. on Conceptual Modeling (ER 2007). Volume 4801 of Lecture Notes in Computer Science., Springer (2007) 277–292

[5] Batini, C., Di Battista, G., Santucci, G.: Structuring primitives for a dictionary of entity relationship data schemas. IEEE Trans. Softw. Eng. **19** (1993) 344–365

[6] 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

[7] Miles, A., Matthews, B., Wilson, M., Brickley, D.: Skos core: simple knowledge organisation for the web. In: DCMI '05: Proceedings of the 2005 international conference on Dublin Core and metadata applications, Dublin Core Metadata Initiative (2005) 1–9

[8] Staab, S., Studer, R.: Handbook on Ontologies (International Handbooks on Information Systems). SpringerVerlag (2004)

[9] Noy, N.F.: Semantic integration: a survey of ontology-based approaches. SIGMOD Rec. **33** (2004) 65–70

[10] Hepp, M.: Possible ontologies: How reality constrains the development of relevant ontologies. IEEE Internet Computing **11** (2007) 90–96

[11] Palmonari, M., Batini, C.: Representing and integrating light-weight semantic web models in the large. In: Proceedings of the 5th International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2009). (2009)

[12] Calvanese, D., Lembo, D., Lenzerini, M., Rosati, R.: Dl-lite: Tractable description logics for ontologies. In: In Proc. of AAAI 2005. (2005) 602–607

[13] 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

[14] Palmonari, M., Batini, C.: Abstract ER^{IA}: a web language for conceptual metadata integration and abstraction in the large. In: MEDES '09: Proceedings of the International Conference on Management of Emergent Digital EcoSystems, New York, NY, USA, ACM (2009) 117–125

[15] Kashyap, V., Borgida, A.: Representing the umls semantic network using owl: (or "what's in a semantic web link?"). In Fensel, D., Sycara, K.P., Mylopoulos, J., eds.: International Semantic Web Conference. Volume 2870 of Lecture Notes in Computer Science., Springer (2003) 1–16

[16] Castano, S., De Antonellis, V., Fugini, M.G., Pernici, B.: Conceptual schema analysis: techniques and applications. ACM Trans. Database Syst. **23** (1998) 286–333

[17] Dahchour, M., Pirotte, A., Zimányi, E.: Generic relationships in information modeling. J. Data Semantics IV **3730** (2005) 1–34

[18] 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

[19] Madhavan, J., Jeffery, S.R., Cohen, S., (luna Dong, X., Ko, D., Yu, C., Halevy, A., Inc, G.: Web-scale data integration: You can only afford to pay as you go. In: In Proc. of CIDR-07. (2007)

[20] Mei, J., Xie, G.T., Zhang, L., Liu, S., Schloss, R.J., Pan, Y., Ni, Y.: Umrr: Towards an enterprise-wide web of models. In Bizer, C., Joshi, A., eds.: International Semantic Web Conference (Posters & Demos). Volume 401 of CEUR Workshop Proceedings., CEUR-WS.org (2008)

[21] Paslaru, E., Simperl, B., Tempich, C., Sure, Y.: Ontocom: A cost estimation model for ontology engineering. In: In Proceedings of the 5th International Semantic Web Conference ISWC2006. (2006)

[22] Palmonari, M., Batini, C.: Representing and integrating light-weight semantic web models in the large. In: Proceedings of the 5th International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2009). (2009)

[23] Tavana, M., Joglekar, P., Redmond, M.A.: An automated entity-relationship clustering algorithm for conceptual database design. Inf. Syst. **32** (2007) 773–792

[24] Mylopoulos, J., Borgida, A., Jarke, M., Koubarakis, M.: Telos: Representing knowledge about information systems. ACM Trans. Inf. Syst. **8** (1990) 325–362

[25] Calvanese, D., Lenzerini, M., Nardi, D.: Unifying class-based representation formalisms. J. of Artificial Intelligence Research **11** (1999) 199–240

[26] Xu, Z., Cao, X., Dong, Y., Su, W.: Formal approach and automated tool for translating er schemata into owl ontologies. In Dai, H., Srikant, R., Zhang, C., eds.: PAKDD. Volume 3056 of Lecture Notes in Computer Science., Springer (2004) 464–475

[27] 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

[28] Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. Web Semant. **5** (2007) 251–261

[29] Calvanese, D., Giacomo, G.D., Lenzerini, M., Nardi, D., Rosati, R.: Description logic framework for information integration. In: KR. (1998) 2–13

[30] Euzenat, J., Shvaiko, P.: Ontology matching. Springer-Verlag, Heidelberg (DE) (2007)

[31] Batini, C., Ceri, S., Navathe, S.B.: Conceptual database design: an Entity-relationship approach. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA (1992)

[32] 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

[33] 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

[34] 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