

Global Schema Generation Using Formal Ontologies

Farshad Hakimpour¹

farshad@geo.unizh.ch
Department of Geography
University of Zurich

Andreas Geppert²

geppert@acm.org
Credit Suisse Financial Services
Zurich, Switzerland

Abstract. This paper deals with the problem of handling semantic heterogeneity during schema integration. Semantics refer to the meaning of data in contrast to syntax, which solely defines the structure of schema elements. We focus on the part of semantics related to the meanings of terms used to name schema elements. Our approach does not rely on the names of the schema elements or the structure of the schema. Instead, we present an approach based on formal ontologies presented in a logical language for integrating schemas to produce global schemas. Semantic similarity relations between definitions in formal ontologies are defined and used for merging formal ontologies. We show how similarity relations are discovered by a reasoning system using a higher level ontology. The result of the merging process is used for schema integration. The result of schema integration can be used as a global schema in a tightly coupled federated database. Afterwards, we illustrate how the produced global schema can help for the mapping of data elements.

1 Introduction

Global schema generation is a critical task performed during information system integration. A major obstacle hereby is *semantic heterogeneity*. Semantics refer to the study of the relation between symbols and what they represent – also called *interpretation* of symbols. Communication is generally the main purpose of the symbols and any misinterpretation of symbols representing data during communication between information systems is called semantic heterogeneity. In our work, the terms used for naming schema elements are the symbols. Therefore, our goal is to reduce the number of misinterpretations of such terms.

Explicit and formal definitions of semantics of the terms guided researchers to apply *formal ontologies* [6] as a potential solution of semantic heterogeneity. A formal ontology consists of *logical axioms* that convey the meaning of terms for a particular *community* [2]. A set of logical axioms defining one term is called *intensional definition*³

1. The work of Farshad Hakimpour is funded by Swiss National Science Foundation (Project Number: 2100-053995).

2. Work done while with the University of Zurich

3. Intensional definitions are estimating every *intensional relation* (defined in [6]). For instance, “Faculty” is an intensional relation and its estimation by an intensional definition is: $t[\text{Faculty}(x)] = \text{Staff}(x) \wedge (\exists y: \text{Course}(y) \wedge \text{teaches}(x,y))$

and there is *only* one intensional definition per term for each community. These definitions are using minimal assumptions from the application domain and explicitly state the specification of a conceptualization [6]. Consensus on ontological definitions among members of a community is an important difference between ontologies and conceptual models [3, 11]. Conceptual models are application-dependent and concerned with challenges of computer representation and independence from implementational issues. In contrast, ontologies are based solely on the understanding of the members of a community and help to reduce ambiguity in communication. Note that formal ontologies carry more knowledge than schema definitions in databases. Schemas are mainly concerned with organizing data in databases according to the specification of application requirements. However, schema definitions are not independent from the ontological definitions and vice versa, they convey part of the knowledge about the ontology of a community.

In this paper, we show how formal ontologies can be used to derive global schemas from local schemas during database integration. This approach relies on ontologies for the local schemas being available. These ontologies are merged, giving rise to similarity relations (such as equality or specialization). The knowledge gained about these relations is then used for global schema definitions in such a way that semantic conflicts (at the schema level) can be detected and resolved. The result of merging ontologies is also used for defining data mappings, i.e. the mapping of local database entities into data elements on the global level.

This paper is structured as following. Section 2 discusses related work. In section 3, an overview of the approach is presented. In sections 4, 5 and 6, we introduce the phases of our approach. Section 4 describes similarity relations and illustrates the process of merging ontologies by an example. In section 5, we discuss the generation of global schemas based on a formal merged ontology. Section 6 presents tasks during data mapping. The last section concludes the paper and discusses future work.

2 Related work

Two known pioneer projects using ontologies for information integration and interoperability are KRAFT and COIN. KRAFT uses shared ontologies [18] as a basis for mapping between ontology definitions and communication between agents. It detects a set of ontology mismatches and establishes the mapping between a shared ontology and local ontologies. The COIN [5] project presents a suitable architecture for semantic interoperability. The role of the *Domain Model* in the COIN-architecture can be compared to that of an ontology. However, the Domain Model is rather close to a conceptual model.

Work of Larson et.al in [10] is very close to this work. This paper address some challenges already addressed in [10]. However, an important difference is that we distinguish and emphasize the difference between the ontological characteristics of attributes and its representational and implementational characteristics. For example, characteristics such as domain, uniqueness, cardinality are present in the relation definitions in an ontology while security constraint or scale are representational characteristics.

The work of Kim et al. [9] represents a comprehensive study for the classification of schema heterogeneities. Also, they present solutions for several types of schema heterogeneity in RDBs and OODBs. Another comprehensive work in this area is presented in [4]. Both efforts address problems of schema heterogeneity, but do not distinguish between the schematic and semantic issues while we focus on the semantic problems here.

Miller et. al. in [15] distinguish schema integration from schema mapping. They focus on the mapping of data, taking the integrated schema for granted. Likewise, we consider phases of schema integration and data mapping. However, data mapping in our work is not yet as complete as the one in [15] and needs further investigations.

Bergamaschi et al. [1], Palopoli et al. [16] and Medhavan et al. [13] propose approaches using thesaurus. Bergamaschi et al. introduced a semiautomatic approach assisting domain experts in extracting relations in the thesaurus from schema structure by help of domain experts. Based on the extracted relations they introduce an algorithm to integrate schema definitions into a global homogeneous schema. Palopoli et al. rely on domain-specific knowledge and enhance that with knowledge extracted from the schema definition. Cupid [13] proposes an approach for schema matching. The approach takes both the similarity of the terms in the schema definitions (language similarity) and the structure of the schema into account (structural similarity). Cupid improves the thesaurus with a coefficient for every entry in the thesaurus - similar to [16]. It also categorizes schema elements into clusters - similar to [1]. In contrast, we try to establish the similarity relations (defined in section 4), while by using a thesaurus synonym and hyponym relations are provided by domain experts. For communication between different application-domains, coefficients of the thesaurus entries must be set by experts in both application-domains.

The Inter-Ontology Relationships Manager module in OBSERVER [14] maintains the same relations between ontological definitions as presented in [1]. By means of inter-ontology relations, OBSERVER replaces terms in user queries with suitable terms in target ontologies.

Projects such as SHOE and On2Broker use formal ontologies as the basis for search engines on the Internet. They use ontologies for translating queries. In such cases, data sources are numerous and their schemas (such as DTDs in XML data) are subject to frequent changes - or they even do not have any schema. The drawback of this approach for structured databases is the high processing cost, because for every query, ontologies must be processed to derive required mappings. On the other hand, human supervision to validate translations is not possible, due to the need for immediate action. Lack of human supervision makes this approach less reliable for more precise and strict applications.

In addition to the closely related work we would like to point out three further major areas of research, namely building, formalising and processing ontologies. *Building* refers to the extraction of specifications from a community's conceptualization. Formalization defines the way how we express ontologies instead of using natural languages. Variety of formalisms can be used such as logical languages, RDF, ER-diagrams, UML diagrams, Conceptual Graphs, which can be distinguished with respect to their degree

of expressiveness. *Processing* refers to reasoning with the formalized definitions. This includes finding relations between definitions, unidentified inconsistencies among the definitions, classifying instance data according to the definitions, etc. Every of the three topics are subject to other research related to this work - not in the scope of this paper, though.

3 Overview

Formal ontologies are used in this work to generate a global schema, which then defines the organization of data in a federated database system. In this approach, class and attribute names in database schema definitions must be based on the terms defined in the formal ontology of a community. As illustrated in Fig. 1, terms used in schema $p1$ and $p2$ are based on the definitions in formal ontology p . This is done by linking class and attribute names in the schema definitions to terms in the formal ontology. In order to find out whether elements from different schemas are related, we define *similarity relations*. Detection of similarity is based on intensional definitions of terms in the formal ontologies. In the first phase, a reasoning system is used to *merge* formal ontologies. The result of merging is used by a schema integrator to build a global schema from local schemas. In the last phase, we find the possible meaningful mappings in the generated global schema and by that establish the mapping of data between the (global and local) databases. This approach is semiautomatic and needs human supervision during schema integration.

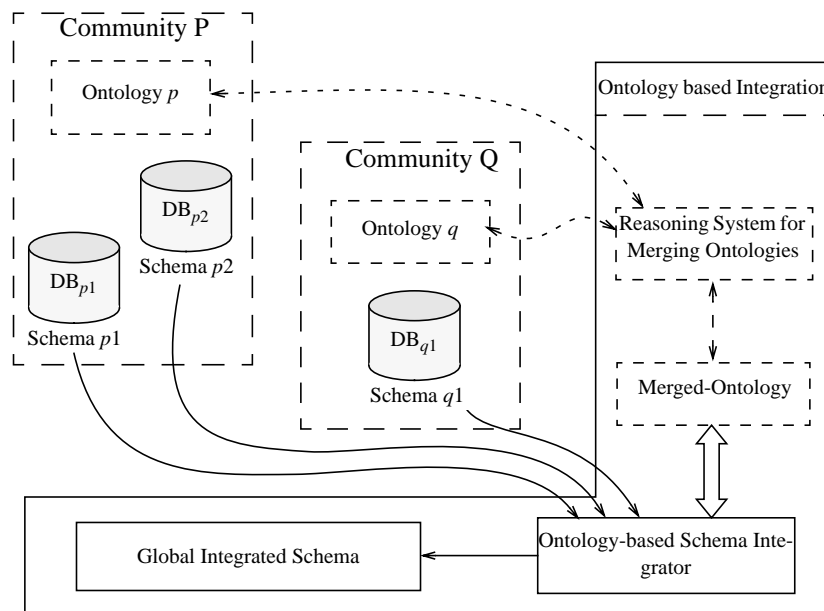


Fig. 1. Global schema generation based on a common ontology produced by integration of ontologies.

4 Merging Ontologies

An important step during database integration is relating schema elements from different local databases. Valid integration relies on a sound understanding of the meaning of the schema elements - this is, it must be found out if schema elements from different schemas refer to the same real world entities, or whether they are different (and if yes, to which degree they differ).

To that end, we rely on ontologies available for local schemas. Schema integration is then based on the merger of these ontologies. Thus, the main task in merging ontologies is to match the intensional definitions from different formal ontologies. The matching is done based on *similarity relations*. By *similarity*, we refer to the particular relations *equality*, *specialization*, *overlapping* and *disjoint* between intensional definitions in two different formal ontologies.

The similarity relations between two terms pT_i and qT_j defined in formal ontologies p and q are defined as following (ι maps a term to its intensional definition) [7]:

- pT_i is *Equal* to (or synonym of) qT_j if and only if both intensional definitions are the same:

$$({}^pT_i = {}^qT_j) \Leftrightarrow (\iota[{}^pT_i] \equiv \iota[{}^qT_j]) .$$

- pT_i is a *Specialization* (or hyponym) of qT_j if and only if the conjunction of the two definition is the same as the definition of pT_i (then qT_j is a *Generalization* or hypernym of pT_i):

$$({}^pT_i \leq {}^qT_j) \Leftrightarrow ((\iota[{}^pT_i] \wedge \iota[{}^qT_j]) \equiv \iota[{}^pT_i]) .$$

- pT_i is *Overlapping with* qT_j if and only if the conjunction of the two definition is not false for all possible states of the world:

$$({}^qT_j \sim {}^pT_i) \Leftrightarrow (((\iota[{}^pT_i] \wedge \iota[{}^qT_j]) \equiv \iota[T_k]) \wedge \neg(\iota[T_k] \equiv False)) \text{ }^1 .$$

T_k is called *conjunction* concept or *conjunction* relation here.

The matching in this phase requires both ontologies to commit to a specific ontology called *higher-level ontology* [17]. A reasoning system will use the higher-level ontology as a common reference in two ontologies for matching.

Note that this approach is not aiming at detecting or resolving mismatches between definitions in ontologies. Consequently, we do not use the term *integrating ontologies* to avoid the wrong impression that any of the communities should agree with or commit to the result of ontology merging. This is because members of one community should not necessarily agree with the terms defined in the ontology of the other communities.

The following example shows parts of the ontologies which are used for a sample merging process. The reasoning system finds the specialization relation between concepts as shown in Fig. 2².

1. If T_k can be proven to be false in all possible worlds then the two intensional definitions are *disjoint*.

Table 1: Higher level ontology for both ontologies p and q

```

(defconcept Person (?p)
:documentation "An individual is a Person if and only if it
is identified by a Social_Security_Number."
:<<=>> (exists (?x Social_Security_Number)
(is-identified-by ?p ?x)))
(deffunction identifies ((?x Social_Security_Number)
:-> (?p Person))
(deffunction is-identified-by ((?p Person)
:-> (?x Social_Security_Number)
:<<=>> (identifies ?x ?p))
(defrelation makes ((?p Person)(?q Quantity))
(defconcept Quantity (?q)
:<<=>> (and (exists (?u Unit) (= (measured-in ?q) ?u))
(exists (?v Number) (= (value-of ?q) ?v)))

```

Table 2: Part of ontology p

```

(defconcept Staff (?x)
:documentation "A Staff is hired by university of zurich; and
if an individual is a Staff then:
- it is identified by a Socoal_Security_Number;
- it earns a salary; and
- its age is greater than 14."
:<<=>> (is-hired-by ?x university_of_zurich)
:=> (and (exists (?y Social_Security_Number)
(= (is-identified-by ?x) ?y))
(exists (?z Salary) (= (earns ?x) ?z))
(and (> (value-of (age ?x)) 14)
(= (measured-in (age ?x)) year))))
(deffunction earns ((?x Staff) :-> (?y Quantity)
:=> (makes ?x ?y))
(defconcept Salary (?z Quantity)
:documentation "Salary is a type of Quantity. A Quantity is
a Salary if and only if a Staff is paid by the Quantity."
:<<=>> (exists (?x Staff) (earns ?x ?z)))
(defconcept Faculty (?x Staff)
:documentation "Faculty is a Staff who teaches at least one
Course."
:<<=>> (exists (?y Course) (teaches ?x ?y)))
(defconcept Professor (?x Faculty))
(defconcept Lecturer (?x Faculty))
(assert (forall (?x Professor) (not (Lecturer ?x))))

```

Table 3: Part of ontology q

```

(defconcept Citizen (?c Person))

```

-
2. Note that Fig. 2, 3 or 4 are only simplified illustrations of ontologies as a means for further discussion. Moreover, even the presented logical axioms are only build to show the capability of this approach.

```

(defconcept Alien (?a Person))
(assert (forall (?x Citizen) (not (Alien ?x))))
(defrelation is-paid ((?q Quantity)(?p Person))
: <=> (makes ?p ?q))
(defconcept Wage (?w Quantity)
:documentation "Wage is a type of Quantity. If and only if a
Quantity is paid to a Person then it is a Wage."
: <=> (exists (?x Person) (is-paid ?w ?x))

```

Detection of the similarity relations is a major task of a reasoning system. When given the above intensional definitions, a reasoning system (this example is prepared by PowerLoom [12]) can detect the following similarities (also shown in Fig. 4):

- “Staff” defined in ontology q is a specialization of “Person”;
- “Salary” is a specialization of “Wage”; and
- “earns” is a specialization of “makes”.

5 Global Schema Generation

We now show how two schemas (S_{p1} and S_{q1}) based on two different ontologies (p and q) can be integrated into a global schema (S_G). Schema integration is done in two main phases: global class derivation and global attribute derivation. In the first phase the class hierarchies are generated, and in the second phase classes are enhanced by attributes.

5.1 Class Integration

All the classes in the local schemas must be based on concept definitions in the community’s formal ontology. In other words, the names of all schema elements used in schema definitions are uniquely referring to definition in the formal ontology of the community¹. As an example, class “Resident” in schema S_{q1} (Fig. 3) is based on the term “Person” defined in a formal ontology p. We show this link to a term in ontology p with τ_p — e.g.,

$$\tau_q: (S_{q1}.Resident) \rightarrow \text{“Person”} \text{ or } \tau_p: (S_{p1}.Prof) \rightarrow \text{“Professor”}.$$

" τ " returns only one term in the respective ontology. If the database designer does not link a schema element to a term in the ontology the integration process will not be able to relate it to other schema elements in other schemas.

The global class derivation is done as follows:

- For every class in the local schema we create a class in the global schema. This guarantees that every class in the local schema is represented by a class in the global schema, which is important for the data mapping phase. The creation of classes is done top-down by starting from superclasses in the class hierarchy of the lo-

1. Whether the term definition already exists in the formal ontology or it is added during database design; or if the links are established just before the integration process or earlier are not in the scope of this paper. We take them for granted here and focus on the integration process.

cal schemas. The subclass relation with the existing superclasses in the global schema is established while a subclass is added.

- During the insertion of the classes, if an *equal* concept has already been represented by another class in the global schema, a new class is not added. That is, to add a new class such as “Resident” to the global schema the following should hold:

$$\forall c \in S_G: \neg[\tau_q(S_{q1}.Resident) = \tau_p(S_G.c)].$$

For example, the class “Resident” should not be added if a class based on the term “Person” is already present in the global schema. In this case, only an alias name “Resident” is stored for the existing class (this is needed during both global attribute generation and the data mapping phases).

- The specialization relation of the concepts in the merged ontology is also reflected as a subclass relation in the global schema. Therefore, we establish a subclass relation with every class based on a generalized concept of the current class - with the condition that the subclass relation cannot be inferred from the class hierarchy in S_G . As an example, the class “ $S_G.S_{p1}.Faculty$ ” is defined as a subclass of “ $S_G.S_{p1}.Resident$ ” considering that the following holds according to the merged-ontology:

$$\tau_p(S_G.Faculty) \leq \tau_q(S_G.Resident).$$

- New classes based on the generated conjunction concepts (during merging ontologies) may be added to the global schema. As an example, if two classes in the global schema are based on two conjunction concepts, such as:

$$\tau_p(S_G.Lecturer) \sim \tau_p(S_G.Student),$$

then a class based on conjunction concept may be added to the global schema. In such a case a class based on the conjunction concept (e.g., teaching assistant) can be added to the global schema. The subclass relation must be established with both classes (multiple inheritance). However, such cases of the conjunction similarity relation happen very often during the merging process, although many of

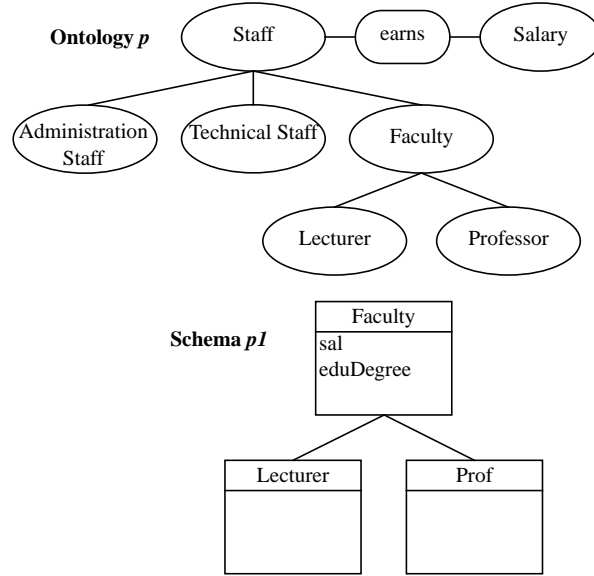


Fig. 2. Schema $p1$ and ontology p .

them are not relevant to applications — e.g., foreign lecturer, native lecturer, foreign professor, etc. Therefore, there is a need for supervision at this point to revise these cases and decide about the necessity of creating such classes. (This task can be done also during merging ontologies, if affecting the merged-ontology by the application view is not an issue).

- If two classes refer to two overlapping or disjoint concepts, while the corresponding concepts have a common superconcept, a class based on the common superconcept is defined in the global schema. As an example, if the class “ S_{q1} .Resident” did not exist in schema $p1$ in Fig. 3 (the schema contained only two classes foreigner and citizen), a class based on the concept “Person” will be added to the global schema.

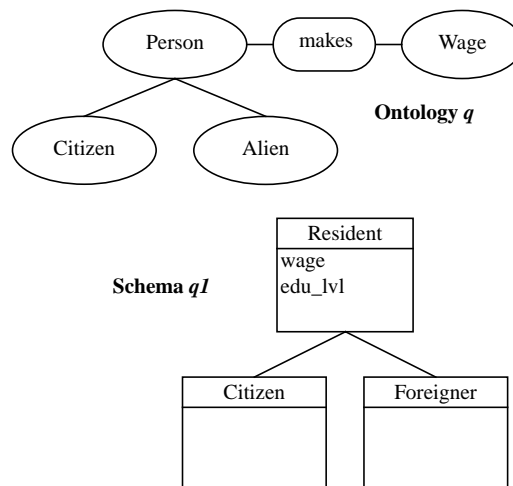


Fig. 3. Schema $q1$ and ontology q

5.2 Filling Classes with Attributes

All attributes in the schemas represent binary relations (either by pointing to another instance of a class or by taking a value of a primitive type such as string or integer value). Since we consider all classes as being based on concepts, attributes must be based on binary relation definitions in the respective ontology — e.g.,

$$\tau_p: (S_{p1}.Faculty.sal) \rightarrow \text{“earns”} \text{ or } \tau_q: (S_{q1}.Faculty.wage) \rightarrow \text{“makes”}.$$

This means that an attribute name in a local schema is linked to a binary relation definition in the ontology. Note that this does not imply every binary relation should be represented in the schema definition - e.g., social security number is not linked to any attribute in any of the classes S_{q1} .Resident or S_{p1} .Staff in schema definitions. The class of the attribute should be based on a concept definition which is not disjoint from the concept in the domain of its binary relation. As an example, the domain of $\tau_p(S_{p1}.Faculty.sal)$ must not be disjoint from $\tau_p(S_{p1}.Faculty)$. This constraint ensures that the τ_q mapping and schema definitions comply (or commit) to the community’s ontology.

During the derivation of attributes for each class in local schema, we define an attribute in the respective class in the global schema. This confirms that each attribute in the local schemas has a counterpart in the global schema. However, an *equal* relation might have already been represented by another attribute in the same global class. For example, before we add a new attribute “wage” to the class “Resident” in the global schema, we check the following:

$$\forall a \in S_G.Resident: \neg[\tau_q(S_{q1}.Resident.wage) = \tau_p(S_G.Resident.a)].$$

If this constraint yields false, we consider the two attributes as semantically equal and keep an equality link between them for the data mapping phase. Unlike the case of classes where we only keep an alias name, here we maintain both attribute definitions in the class. The main reason is that the equality link does not indicate the similarity in the data type, unit, structure, etc. of the value of the attribute. We will deal with this matter in the data mapping phase.

The same is done in case attributes are based on relations in specialization similarity. This case also should only be detected in this phase and will be used during data mapping. As an example, while defining attribute “sal” for class “Staff”, the following similarity is detected and will be kept for the data mapping phase:

$$\tau_p(S_G.\text{Staff.sal}) \leq \tau_q(S_G.\text{Staff.wage}).$$

We did not find the case of attributes based on conjunction relation relevant to our work. Detection of such cases is not problematic, though. (As an example one can imagine two attribute $S_{q1}.\text{Resident.father}$ and $S_{p1}.\text{Staff.colleague}$.)

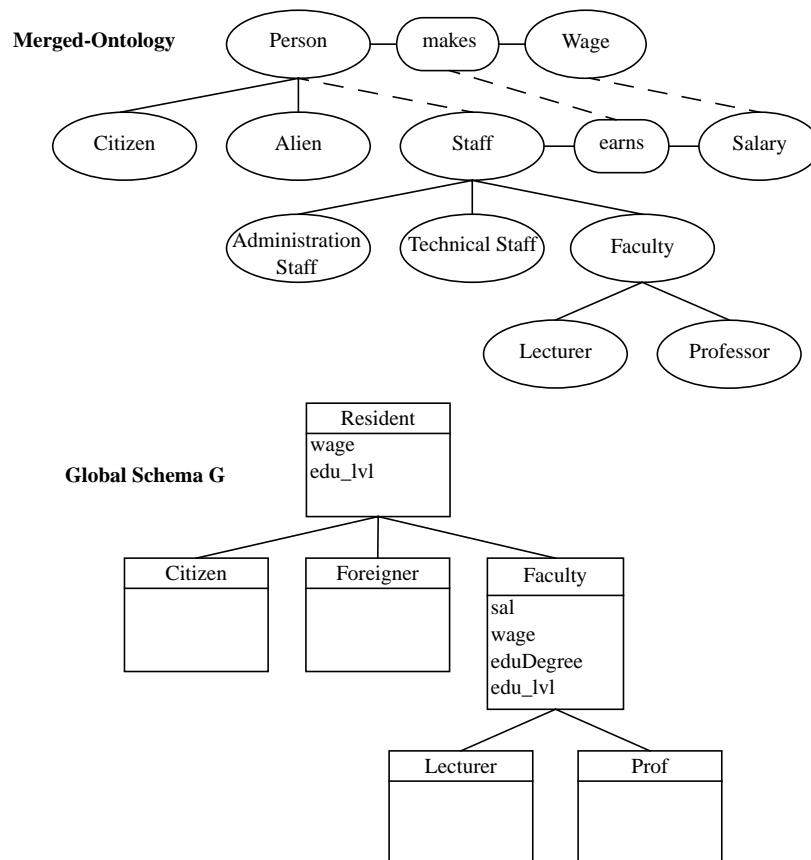


Fig. 4. Result of merging parts of formal ontologies by finding a specialization similarity and creation of class hierarchy based on merged ontology.

In general, relation definitions in ontologies can not only relate instances of two concepts, they additionally may relate two concepts; or instances of a concept with sub-concepts of another. An example of this case is shown in Fig. 5. In such a case an attribute based on the relation is not filled in with a value (e.g., a number or a character string) or an instance of a concept, but with a code that represents a set of individuals or a range of values (e.g., “M.sc. degree” in Fig. 5 or “tall” for an attribute height).

6 Data Mapping

The generated global schema can be used for the integration of two databases instantiating the schemas S_{p1} and S_{q1} . The instances of classes in the local databases are mapped to those of the global schema and vice versa. This mapping of instances is straightforward and relies on the information kept during the integration process. Afterwards, a set of operations performs the mapping of the data at the global schema.

Classes in two schemas referring to the same definition are mapped to the same class by means of alias names. In case classes are in a specialization relation in the global schema, all instances of the subclass can be mapped to the superclass, but not the other direction. For the mapping from a superclass to its subclass, we need a *classification criterion*. As an example, instances of the subclass “S_G.Faculty” are mapped to instances of the superclass “S_G.Resident”. However, in order to map instances of the superclass “S_G.Resident” to the subclass “S_G.Faculty”, these instances should satisfy a classification criterion. By referring to the definitions in section 4, one can see that if an instance of “S_G.Resident” is a “Staff” and “teaches” a “Course” it can be mapped to class “S_G.Faculty”. A reasoning system finds such criteria by referring to the intensional definitions. The reasoning system can classify the instances during the data mapping in the global schema.

A problem occurs during data mapping when two instances (from two databases) are classified under one class in the global schema, while they represent the same individual in the domain. To deal with this problem, we need an *identification criterion* to recognize if two objects in the underlying databases represent different individuals. This criterion must be present in both local class definitions — such as `social_security_number` in our example. The identification criterion may not necessarily be the primary key in one or both systems, though.

Attributes of a class in a local schema are mapped directly to their counterpart in the global schema. A set of rules map attributes in the global schema. In case two attributes are linked by equal similarity, the attribute values will be mapped mutually. In case two attributes are linked by a specialization similarity, the value of the specialized attribute can be mapped to the generalized one, but not the other way round. By looking at the definitions in section 4, one can see that every “earns” relation is a “makes” relation but not the other way round. Yet, one can argue that in this example (Fig. 3), all values of attribute “S_G.Resident.wage” can be mapped to “S_G.Faculty.sal” and that this is what happens during data mapping. However, in general there are cases in which attribute values should be mapped by considering a classification criterion. (Attributes referring to sibling and sister relations are simple examples of this case.)

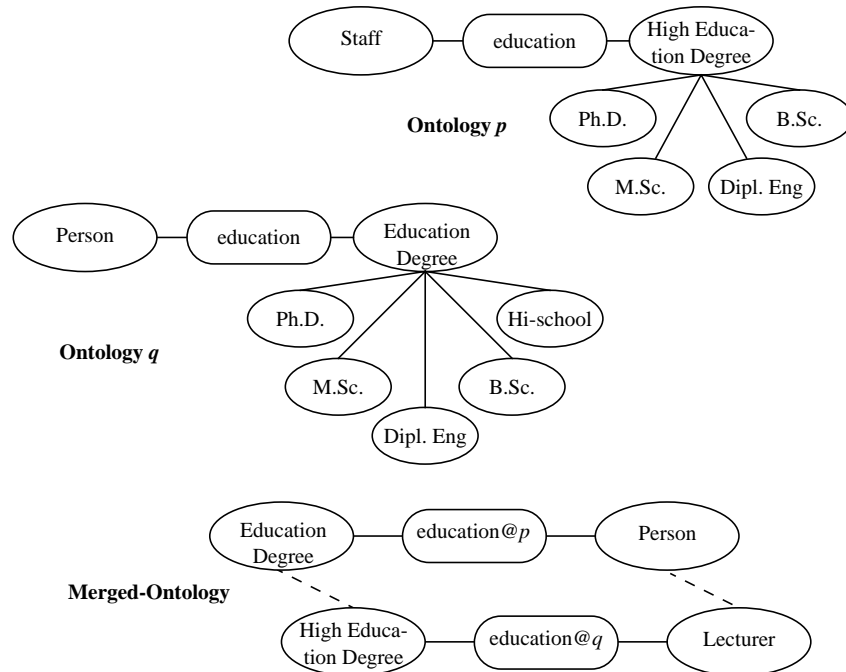


Fig. 5. Example of relations between instances of a concept and subconcepts of another.

An attribute mapping often requires a data conversion process (e.g., integer to real). This is because during the integration we do not employ any knowledge about the data types. There is often need for further processes for data mapping — a simple example is change of units. If the changes are not due to the structural differences, a detailed ontology can eventually help in some cases (such as unit conversion).

Furthermore, we only consider mapping of one element of the schema to only one other element. More complicated data mappings in which a global class is an aggregation of classes in the local schemas or an attribute may be the result of calculation over many attributes from local schemas (as presented in [15]) are not treated here. However, the focus of this work has been to solve semantic problems in terms of finding similarities between classes and attributes and finding mappings between them, but not correspondence of data values.

7 Conclusion

In this paper, we present an approach to generate global schemas. This solution uses formal ontologies as a basis for the integration and for resolving heterogeneity problems during the integration of local schemas. We only use the intensional definitions in formal ontologies, that is, we do not use the matching of the terms used to name schema elements. This approach also does not rely on or consider the structures expressed by the schema.

The quality of the formal ontologies plays an important role here. There are two important quality measures for the success of this approach:

- accordance of an ontology with the *conceptualization* (defined in [6]) of the community and
- details of explicit specifications of implicit assumptions in the community are important during building ontologies.

Another factor that plays an important role in the success of this approach is the commitment of the schema definitions to the community's formal ontology. There are constraints that should apply to the schema definitions. One such constraint is discussed in the beginning of section 5.2. Such constraints should be clearly specified.

Building a higher level ontology that many communities agree upon is a difficult task. This makes ontologies expensive to build. However, it is a price to avoid semantic conflicts that can be even more expensive. Furthermore, formal ontologies are long term assets that will remain independent of the application systems. As ontologies are becoming more popular they can be used for other purposes other than database integration.

Although we considered the integration of classes and attributes, methods are not discussed in this paper. Methods as parametric attributes based on the relation definitions of arity higher than two in a formal ontology. However, we can only expect those methods that are representing an *action* to be based on a definition in the formal ontology (actions change the states of the world). That is, methods that are results of implementation (such as triggers) may not have a counterpart in the communities' ontology. This is a general discussion that may apply rarely to attributes which are only used for implementational reasons (such as object id) and do not have a counterpart in the ontology.

References

- [1] S. Bergamaschi, S. Castano, S. De Capitani di Vimercati, S. Montanari, and M. Vincini. An intelligent approach to information integration. In Nicola Guarino, editor, *Formal Ontology in Information Systems*, pages 253–267. IOS Press, 1998.
- [2] Y. A. Bishr, H. Pundt, W. Kuhn, and M. Radwan. Probing the concept of information communities - a first step toward semantic interoperability. In M. Goodchild, Max Egenhofer, R. Fegeas, and C. Kottman, editors, *Interoperating Geographic Information Systems*, pages 55–69. Kluwer Academic, 1999.
- [3] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems*. Addison-Wesley, third edition, 2000.
- [4] Manuel Garcia-Solaco, Felix Saltor, and Malu Castellanos. Semantic heterogeneity in multidatabase systems. In Omran A. Bukhres and Ahmed K. Elmagarmid, editors, *Object-oriented Multidatabase Systems: A Solution for Advanced Applications*, chapter 5, pages 129–202. Printice-Hall, 1996.
- [5] Cheng Hian Goh, Stephane Bressan, Stuart Madnick, and Michael Siegel. Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transaction on Information Systems*, 17(3):270–290, 1999.

- [6] Nicola Guarino. Formal ontology and information systems. In Nicola Guarino, editor, *Formal Ontology in Information Systems, Proceedings of FOIS'98*, pages 3–17, Trento, Italy, June 1998. IOS Press, Amsterdam.
- [7] Farshad Hakimpour and Andreas Geppert. Resolving semantic heterogeneity in schema integration: An ontology base approach. In Chris Welty and Barry Smith, editors, *Proceedings of International conference on Formal Ontologies in Information Systems FOIS'01*. ACM Press, October 2001.
- [8] R. J. Bayardo Jr., W. Bohrer, R. Brice, A. Cichochi, J. Fowler, A. Helal, V. Kashyap, T. Ksiez, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. woelk. InfoSleuth: Agent-based semantic integration of information in open and dynamic environments. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 195–206, 1997.
- [9] Won Kim, Injun Choi, Sunit Gala, and Mark Scheevel. On resolving schematic heterogeneity in multidatabase systems. *Distributed and Parallel Databases*, 1(3):251–277, July 1993.
- [10] James A. Larson, Shamkant B. Navathe, and Ramez Elmasri. A theory of attribute equivalence in database with application to schema integration. *IEEE Transactions on Software Engineering*, 15(4):449–463, April 1989.
- [11] Pericles Loucopoulos. Conceptual modelling. In Pericles Loucopoulos and Robert Zicari, editors, *Conceptual Modelling, Databases, and CASE: an Integrated View of Information system development*, chapter Introduction, pages 1–26. John Wiley & Sons, Inc., 1992.
- [12] Robert M. MacGregor, Hans Chalupsky, and Eric R. Melz. *PowerLoom Manual*. University of Southern California, <http://www.isi.edu/isd/LOOM/PowerLoom/documentation/manual.pdf>, November 1997.
- [13] Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In Peter M. G. Apers, Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Kotagiri Ramamohanarao, and Richard T. Snodgrass, editors, *VLDB 2001, Proceedings of 27th International Conference on Very Large Databases, September 11-14, 2001, Roma, Italy*, pages 49–58. Morgan Kaufmann, September 2001.
- [14] E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Domain specific ontologies for semantic information brokering on the global information infrastructure. In Nicola Guarino, editor, *Formal Ontology in Information Systems*. IOS press, 1998.
- [15] René J. Miller, Laura M. Haas, and Mauricio A. Hernández. Schema mapping as query discovery. In Amr El Abbadi, Michael L. Brodie, Sharma Chakravarthy, Umeshwar Dayal, Nabil Kamel, Gunter Schlageter, and Kyu-Young Whang, editors, *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, pages 77–88. Morgan Kaufmann, 2000.
- [16] Luigi Palopli, Domenico Sacca, and Domenico Ursino. Semi-automatic technique for deriving interscheme properties from database schemes. *Data and Knowledge Engineering*, 30(3):239–273, 1999.
- [17] Pepijn R. S. Visser and Zhan Cui. Heterogeneous ontology structures for distributed architectures. In *ECAI-98 Workshop on Applications of Ontologies and Problem-solving Methods*, pages 112–119, 1998.
- [18] Pepijn R.S. Visser, Dean M. Jones, M.D. Beer, T.J.M. Bench-Capon, B.M. Diaz, and M.J.R. Shave. Resolving ontological heterogeneity in the KRAFT project. In *10th International Conference and Workshop on Database and Expert Systems Applications DEXA'99*. University of Florence, Italy, August 1999.