Information Technology and Management 6, 97–122, 2005

Resolution of Semantic Heterogeneity in Database Schema Integration Using Formal Ontologies

FARSHAD HAKIMPOUR f.hakimpour@open.ac.uk
Geographic Information Analysis Division, Department of Geography, University of Zurich, Winterthurerstr. 190, Zurich, Switzerland

ANDREAS GEPPERT* geppert@acm.org
Credit Suisse, Zurich, Switzerland

Abstract. This paper addresses the problem of handling semantic heterogeneity during database schema integration. We focus on the semantics of terms used as identifiers in schema definitions. Our solution does not rely on the names of the schema elements or the structure of the schemas. Instead, we utilize formal ontologies consisting of intensional definitions of terms represented in a logical language. The approach is based on similarity relations between intensional definitions in different ontologies. We present the definitions of similarity relations based on intensional definitions in formal ontologies. The extensional consequences of intensional relations are addressed. The paper shows how similarity relations are discovered by a reasoning system using a higher-level ontology. These similarity relations are then used to derive an integrated schema in two steps. First, we show how to use similarity relations to generate the class hierarchy of the global schema. Second, we explain how to enhance the class definitions with attributes. This approach reduces the cost of generating or re-generating global schemas for tightly-coupled federated databases.

Keywords: database integration, schema integration, formal ontologies, semantic heterogeneity

In many domains, the number of data providers and amounts of available data is increasing. Data consumers require consistent views of the data available from heterogeneous data sources. Therefore, integration issues are attracting ever more attention. Data integration refers to combining data in such a way that a homogeneous and uniform view is presented to users. Global schema generation is a critical task performed during data integration and is non-trivial because of *semantic heterogeneity*. We distinguish two types of heterogeneity: structural and semantic heterogeneity. Structural heterogeneity refers to differences among definitions, such as attribute types, formats, and precision. For example, two data sources may represent the same object using different structures. Semantics refers to the interpretation people assign to data (i.e., relating data to what they represent). Thus, semantic heterogeneity refers to *differences in the meaning of data*. For instance, two schema elements (i.e., classes or attributes) in two local data sources can have the same name, but different intended meanings. Both aforementioned kinds of heterogeneity are different from data inconsistency, which refers to conflicting data val-

* Work done while with the University of Zurich.

ues in two or more data sources. That is, multiple data sources may state contradictory facts about the same object.

In data integration, each local database provides a description of the data it is willing to share (the export schema, which is a subset of the local schema). The aim of the integration process is to obtain a global schema which relates and subsumes the export schemas from the local databases. Different interpretations of data cause semantic heterogeneity. If semantic conflicts are not detected and resolved, the use of integrated data leads to invalid results. Even worse, since users do not know about the underlying misinterpretation of the data they use, they will not have a chance to realize that the results are invalid. Adequate and meaningful data integration therefore relies on the detection of discrepancies and similarities between interpretations of schema elements. The goal of the approach presented here is to reduce the number of misinterpretations of terms in database schema definitions.

To that end, this paper shows how formal ontologies can be used to derive global schemas from local schemas for database integration. The approach relies on formal ontologies being available for the local schemas. Ontologies belonging to local schemas are checked for similarities (such as equality or specialization). The knowledge gained about similarity 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 the definition of data mappings, i.e. the mapping of local database entities into global ones.

The remainder of this paper is structured as follows. Section 1 explains notions used in the paper. Section 2 presents an overview of the proposed solution to use ontologies. We also briefly discuss the whole process and its characteristics in comparison to other approaches. In section 3, we propose our definition of semantic similarity relations based on intensional definitions of terms. We then show how a reasoning system can assist in determining similarity relations by means of higher-level ontologies in section 4. Section 5 discusses the integration of schemas into the global schema of a federated database system. In section 6, we explain how this approach supports the data mapping phase. Section 7 introduces relevant related work and a comparison with this work. Section 8 concludes the paper.

1. Background

Relying on common sense is a typical source of semantic heterogeneity; explicit definitions of terms used in schemas are a solution to this problem. The need for explicit and formal definitions of the semantics of the terms led researchers to apply *formal ontologies* [17,39] as a potential solution to semantic heterogeneity. A formal ontology consists of *logical axioms* that convey the meaning of terms for a particular *community* [5,10,33]. Sharing a common set of concerns and values is a particular characteristic of communities [10]. A set of logical axioms defining a term is called an *intensional definition* (denoted by ι). There is *exactly* one intensional definition for each term in a community. An intensional definition approximates an *intensional relation* (also called *conceptual* *relation*, see definition in [16]). Intensional definitions use minimal assumptions about the application domain and explicitly state the specification of a conceptualization. For instance, “Main Street” is a term with the following possible intensional definition:

$$\begin{aligned}&\iota[\mathtt{Main\_Street(x)}] = \\ &\quad \mathtt{Transportation\_Path(x)} \wedge (\mathtt{carPerDay(x)} > \mathtt{2000}) \wedge \\ &\quad (\exists \mathtt{y} : \mathtt{Residential\_Area(y)} \wedge \mathtt{inside(x, y)}).\end{aligned}$$

In contrast to the intension of a term, its *extension* (denoted by ε) refers to the set of instances that are classified under that term at a given point in time.

An ontology for a large number of communities cannot be complete or highly specialized. The more detailed definitions in an ontology for a community, the more difficult it is to reach a consensus within the community or even between communities. Ontologies with general terms and minimum constraints which are used by many communities to develop specialized ontologies are called *higher-level ontologies*. A community can adopt a higher-level ontology and specialize it by adding its own definitions to it. A modular structure of formal ontologies facilitates developing and reusing ontologies. As a result, a specialized ontology cannot remove any constraint or term of a higher-level ontology without agreement of the communities already committed to that ontology [37]. Extending or modifying intensional definitions of an ontology requires an agreement among all communities committed to it.

In this paper, the term *concept* is used to refer to an intensional relation with arity one (e.g., Main_Street(x)) and *relation* is used to refer to an intensional relation of arity greater than one (e.g., inside(x, y)). The notion of relationship in conceptual models (e.g., ER, UML) is comparable with that of relation used here. However, our solution addresses database schemas instead of conceptual models. A (logical) database schema (or simply *schema*) contains the definition of *classes* and *attributes*. Database schemas are concerned with organizing data in databases and are derived from conceptual schemas. Formal ontologies carry more semantics than database schema definitions. However, schema definitions are not independent from the ontological definitions and thus convey part of the knowledge about the ontology of a community.

2. Overview

There are two possible approaches for applying ontologies for data integration:

1. Two or more database schemas are based on the *same common* ontology and this ontology facilitates the data communication between communities (figure 1). The problem is limited to schema integration or finding synonyms and homonyms among the names of schema elements. An example of this approach is On2Broker [12] using the $(KA)^2$ [3] ontology. Note that this solution is applicable whenever the two schemas have already been designed based on the same ontology. Otherwise, building such a common ontology requires resolving conflicts which can be a diffi-

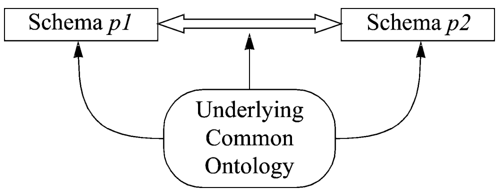


Figure 1. One common ontology shared between two databases.

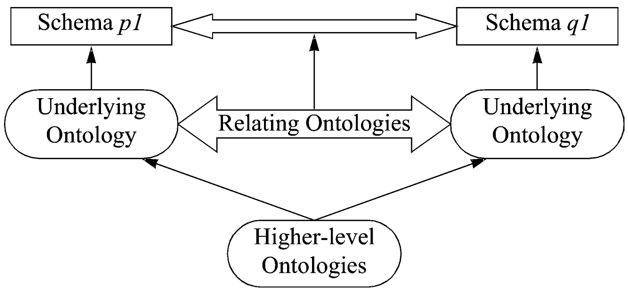


Figure 2. Database schemas based on different ontologies.

cult task. In fact, this solution is not applicable for schemas committing to different ontologies.

2. Database schemas are based on *different* ontologies. In this scenario, two communities using different ontologies aim to interchange or integrate their data (figure 2). The problem is to identify the similarities and differences between concepts defined in the underlying ontologies. Finding similarities in turn requires a minimum of agreement expected from a higher-level ontology. This solution leaves more flexibility to communities to define their ontologies. This approach can be compared to that of SHOE [22] where every Internet page can introduce the ontology its content is based on.

This work presents a solution based on the second scenario by first relating ontologies and then using the detected relations to solve heterogeneity problems.

The use of ontologies in resolving semantic heterogeneity follows two main trends. One uses ontologies for translating queries or their results (as in [12,22,30]), illustrated in figure 3. This approach is suitable whenever schemas are subject to frequent changes (such as DTDs describing XML-documents), many data sources are involved, or the number of involved data sources changes frequently (such as data sources in the Internet). One drawback of this approach is the high processing cost, since for every query, ontologies must be processed to derive required mappings. On the other hand, human supervision to validate extracted similarities is not practical due to the need for immediate action. Lack of human supervision makes this approach less reliable, e.g., as in ontology-based search engines.

The second trend uses ontologies for the generation of global schemas [4,15,19]. It is suitable whenever the schemas do not change frequently. This paper is pursuing

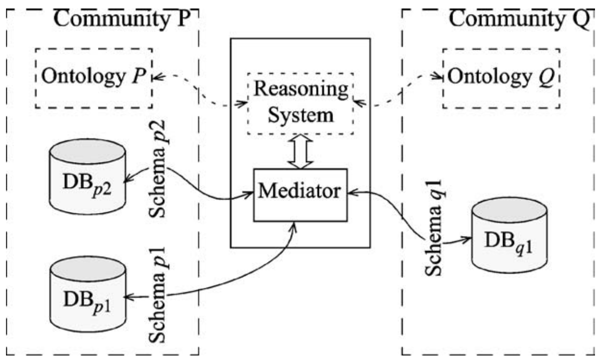


Figure 3. On the fly integration, with local queries committing to a domain ontology and no global schema or global query.

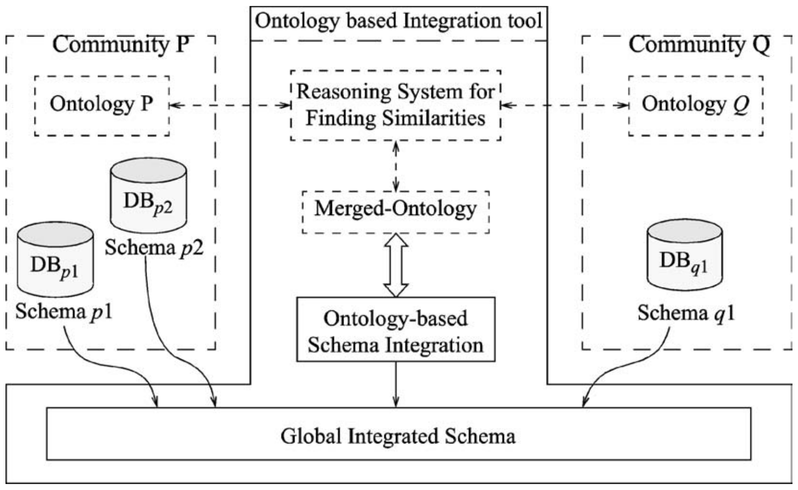


Figure 4. Global schema generation based on a common ontology produced by an integration of domain ontologies.

the second approach as illustrated in figure 4. The method proposed in this paper is based on formal ontologies to generate a global schema. Database schemas are based on the ontology of a community. That is, schema elements (i.e., class and attribute names) refer to terms defined in the ontology of the community. In figure 4, two databases DB$_{p1}$ and DB$_{p2}$ commit to an ontology P by their schemas referring to the terms defined in the ontology P. Such relations can be established either by hard links or by using the same terms as they are defined in the ontology. We consider the former case here (denoted by τ). The link is created by an index that maps every schema element name to a term. These links should be established either during the database design or prior to the integration process, as in case of legacy systems.

Two major tasks should take place before the integration process starts. First, building and formalizing ontologies and second, relating schema elements to the terms defined in the ontologies. The former task yields a set of detailed definitions of terms. It is based on the agreement between members of the community (or users of the database). Building ontologies also requires understanding and adopting definitions from higher-level ontologies. This is a critical and expensive task in the approach. The second task is to relate schema items to the existing ontologies. This task is similar to that of annotating Web pages [21] in the framework of the Semantic Web. However, the structured nature of the data in databases renders this task less complex. The result of this task is an index function (shown by τ, called commitment here) that returns a term in the ontology for the name of every schema item. By using this index function, the name of the schema item does not necessarily need to equal the name of the corresponding term in the ontology. This flexibility can be important when adopting an existing ontology, or when names of schema items do not comply with the terms in the ontology.

In this paper we assume both tasks are already performed and continue with the integration process.

In order to find out whether elements from different schemas are related in some way, we define *similarity relations*. Detection of similarity is based on the intensional definitions of terms in the formal ontologies. Intensional definitions are formulated in Description Logic [1]. In the first phase, the PowerLoom [28] reasoning system is used to find similarity relations. Two ontologies together with the similarity relations existing between their terms are called *merged-ontology*. The result of matching intensional definitions (i.e., a merged-ontology) is used by a schema integrator to build a global schema from local schemas. We establish possible data mappings between the generated global schema and local schemas. This approach is semi-automatic and needs human supervision during schema integration.

3. Similarity relations

An important step during database integration is to determine relations between schema elements from different local databases. Finding meaningful relations is in turn based on a sound understanding of the meaning of the schema elements, i.e., their semantics. To that end, we rely on formal ontologies available for local schemas. Relating schema elements is performed based on *semantic similarity* relations. Possible similarity relations include *equality, specialization* (*or generalization*), *overlapping* and *disjointness* [9].

The similarity relations between two terms ${}^{p}T_i$ and ${}^{q}T_j$ defined in formal ontologies p and q are defined as following [18]:

- ${}^{p}T_i$ is *Equal* to (or synonym of) ${}^{q}T_j$ if and only if both intensional definitions are the same:

$$\big(\iota[{}^{p}T_i] = \iota[{}^{q}T_j]\big). \tag{1}$$

  Equality is denoted by "≡" as in ${}^{p}T_i \equiv {}^{q}T_j$.

- ${}^{p}T_i$ is a *Specialization* (or hyponym) of ${}^{q}T_j$ if and only if the conjunction of the two definitions is the same as the definition of ${}^{p}T_t$ (then ${}^{q}T_j$ is a *Generalization* or hypernym of ${}^{p}T_i$):

$$\big(\big(\iota[{}^{p}T_i] \wedge \iota[{}^{q}T_j]\big) = \iota[{}^{p}T_i]\big). \tag{2}$$

Specialization is represented by "$\leqslant$" as in ${}^{p}T_i \leqslant {}^{q}T_j$.

- ${}^{p}T_i$ is *Overlapping with* ${}^{q}T_j$ if and only if the conjunction of the two definitions is not false for all possible states of the world:

  $$\big(\big(\big(\iota[{}^{p}T_i] \wedge \iota[{}^{q}T_j]\big) = \iota[T_k]\big) \wedge \neg(\iota[T_k] = \mathit{False})\big). \tag{3}$$

  T_k is called a *conjunction* concept or *conjunction* relation. If T_k can be proven to be false in all possible worlds, then the two intensional definitions are *disjoint*. Overlapping is shown by "$\approx$" as in ${}^{p}T_i \approx {}^{q}T_j$.
- ${}^{p}T_i$ is *Disjoint from* ${}^{q}T_j$ if and only if the conjunction of the two definitions is false for all possible states of the world:

  $$\big(\big(\iota[{}^{p}T_i] \wedge \iota[{}^{q}T_j]\big) \equiv \mathit{False}\big). \tag{4}$$

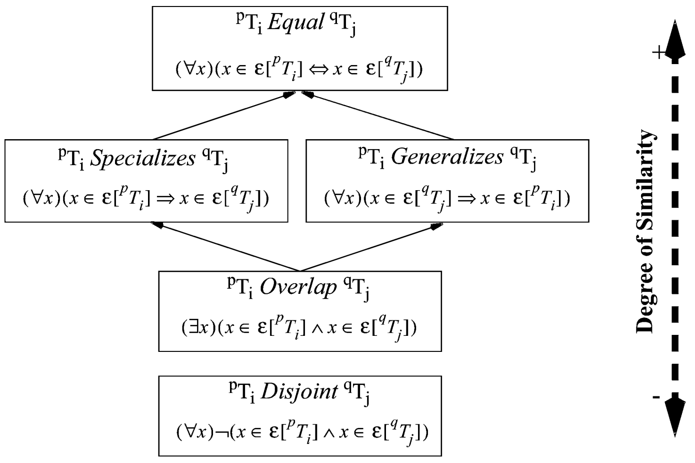
  Disjointness is denoted by "$\neq$" as in ${}^{p}T_i \neq {}^{q}T_j$.

The above similarity relations are results of adopting an approach presented in [8] where relations between spatial regions have been derived. This approach is used to explore possible relations between intensional definition. As presented in table 1, we

Table 1
Deriving similarity relations based on the consistency of conjunction of intensional definitions of terms A and B.[a]

	$\iota[A] \wedge \iota[B]$	$\iota[A] \wedge \neg\iota[B]$	$\neg\iota[A] \wedge \iota[B]$	$\neg\iota[A] \wedge \neg\iota[B]$	
1	■F	■F	■F	■F	Impossible
2	■F	■F	■F	◆T	$\iota[A] = \iota[B] =$ ■F
3	■F	■F	◆T	■F	$\iota[A] =$ ■F, $\iota[B] =$ ■T
4	■F	■F	◆T	◆T	$\iota[A] =$ ■F
5	■F	◆T	■F	■F	$\iota[A] =$ ■T, $\iota[B] =$ ■F
6	■F	◆T	■F	◆T	$\iota[B] =$ ■F
7	■F	◆T	◆T	■F	$\iota[A] = \neg\iota[B]$ (A$\perp$B)
8	■F	◆T	◆T	◆T	A$\perp$B
9	◆T	■F	■F	■F	$\iota[A] = \iota[B] =$ ■T (A $\cong$ B)
10	◆T	■F	■F	◆T	A $\cong$ B
11	◆T	■F	◆T	■F	$\iota[B] =$ ■T (A $\leqslant$ B)
12	◆T	■F	◆T	◆T	A $\leqslant$ B
13	◆T	◆T	■F	■F	$\iota[A] =$ ■T (B $\leqslant$ A)
14	◆T	◆T	■F	◆T	B $\leqslant$ A
15	◆T	◆T	◆T	■F	B $\neq$ A and $\iota[A] \vee \iota[B] =$ ■T
16	◆T	◆T	◆T	◆T	B $\neq$ A

[a] ■F or necessarily false: false for all possible worlds.
◆T or possibly true: true for at least one possible world (¬■F = ◆T).



${}^{p}T_i$ *Equal* ${}^{q}T_j$
$(\forall x)(x \in \varepsilon[{}^{p}T_i] \Leftrightarrow x \in \varepsilon[{}^{q}T_j])$
${}^{p}T_i$ *Specializes* ${}^{q}T_j$
$(\forall x)(x \in \varepsilon[{}^{p}T_i] \Rightarrow x \in \varepsilon[{}^{q}T_j])$
${}^{p}T_i$ *Generalizes* ${}^{q}T_j$
$(\forall x)(x \in \varepsilon[{}^{q}T_j] \Rightarrow x \in \varepsilon[{}^{p}T_i])$
${}^{p}T_i$ *Overlap* ${}^{q}T_j$
$(\exists x)(x \in \varepsilon[{}^{p}T_i] \wedge x \in \varepsilon[{}^{q}T_j])$
${}^{p}T_i$ *Disjoint* ${}^{q}T_j$
$(\forall x)\neg(x \in \varepsilon[{}^{p}T_i] \wedge x \in \varepsilon[{}^{q}T_j])$
+
Degree of Similarity
-

Figure 5. Levels of similarity among definitions in formal ontologies.

use four combinations of conjunctions of the intensional definitions of two terms and their negation. The sixteen combinations result in five major relations. The extensional implications of the similarity relations are shown in figure 5. Specialization and generalization are considered at the same level of similarity (figure 5) since one is just the inverse of the other.

One may recognize finer degrees of similarity than simply disjoint definitions. For instance, railways and highways a more similar (read: less disjoint) than railways and buildings. However, such finer degrees of similarity are not relevant to the integration process.

4. Finding similarities

Finding similarities between intensional definitions in two ontologies requires both ontologies to commit to higher-level ontologies. A reasoning system uses higher-level ontologies as a common reference for finding similarity relations in two ontologies. This approach is not aiming at detecting or resolving conflicts or mismatches between definitions in the 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 merged-ontology. This is because members of one community might not agree with the terms defined in the ontology of the other community, but only to those of the higher-level ontology.

Tables 2 and 3 show simple examples[1] of the logical expressions defined in Description Logic [1] for PowerLoom [28]. Taxonomy trees of the ontologies P and Q are illustrated in figures 6 and 7. The logical expressions define the application domain

[1] Readers who are not familiar with Description Logic can still read only the documentation of the definitions. The documentation lines help to understand the semantics of the terms and helps those not familiar with the German language.

Table 2
Part of intensional definitions of terms in ontology P.

```
(defconcept Highway (?hw)
:documentation "A highway transports cars, has a speed limit of at
least 100, and has at least 4 lanes."
:<<=>> (and (exists (?c Car) (transports ?hw ?c))
            (>= (speed_limit ?hw) 100)
            (>= (cardinality has-lane ?hw) 4)))
(defconcept Street (?s)
:documentation "A street transports cars and is located in a
residential area."
:<<=>> (and (exists (?c Car) (transports ?s ?c))
            (exists (?ra Residential_Area) (inside ?ra ?s))))
(defconcept Road (?r)
:documentation "A road transports cars and is located outside of any
residential area."
:<<=>> (and (exists (?c Car) (transports ?r ?c))
            (forall (?ra Residential_Area) (outside ?ra ?r))))
(defconcept Railway (?f)
:documentation "A transportation path is a Railway if and only if it
transports trains."
:<<=>> (exists (?t Train) (transports ?f ?t)))
(defconcept PavementType (?pt)
:documentation "PavementType is one of the following:  unpaved, tiled
or asphalt.  They are defined as fixed elements of pavement, although
one cannot assign an intensional definition to them.  Their intensional
definitions requires second order logic."
:<<=>> (setof unpaved tiled asphalt))
(deffunction covered-by (?x) :-> (?pt PavementType)
:documentation "covered-by assigns a pavement type to any of the
following concepts:  Street, Highway or Road"
:axioms (or (domain covered-by Street)
            (domain covered-by Road)
            (domain covered-by Road))
:=>>    (or (Street ?x) (Highway ?x) (Road ?x)))
(defconcept Primary_Road (?pr Road)
:documentation "A primary road has a high amount of traffic, that is,
having at least 1500 cars per day running over it."
:<<=>> (>= (vehicle_traffic ?pr) 1500))
(defconcept Secondary_Road (?sr Road)
:documentation "Secondary Road has a low amount of traffic, i.e., less
than 1500 cars travel on it each day."
:<<=>> (< (vehicle_traffic ?sr) 1500))
(defconcept Narrow_Road (?nr Primary_Road)
:documentation "A narrow road is a primary road being less than 25 wide."
:<<=>> (< (width ?nr) 25))
(defconcept Wide_Road (?wr Primary_Road)
:documentation "A wide road is a primary road being at least 30 wide."
:<<=>> (>= (width ?wr) 30))
```

Table 3
Part of intensional definitions of terms in ontology Q.

```
(defconcept Strasse (?r Transportation_Path)
:documentation "Strasse is a subconcept of transportation path that
transports cars.  It is independent of where it is located, unlike
street."
:<<=>> (exists (?c Car) (transports ?r ?c)))

(defconcept Hauptstrasse (?h Strasse)
:documentation "Hauptstrasse (or main road) is a strasse which has a
speed limit of at least 100."
:<<=>> (>= (speed_limit ?h) 50))

(defconcept Nebenstrasse (?n Strasse)
:documentation "Nebenstrasse (or secondary road) is a strasse which has
a speed limit of less than 100."
:<<=>> (< (speed_limit ?n) 100))

(defconcept Schnellstrasse (?ss Strasse)
:documentation "Schnellstrasse (expressway) is a strasse which has a
speed limit of at least 100 and has at least 6 lanes."
:<<=>> (and (>= (speed_limit ?ss) 100)
            (>= (cardinality has-lane ?ss) 6)))

(defrelation strassenkreuzung ((?x Strasse)(?y Strasse))
:documentation "strassenkreuzung is an intersection between two
instances of Strasse."
:<<=>> (and (intersect ?x ?y) (Strasse ?x) (Strasse ?y)))

(defconcept Schienenbahn (?f)
:documentation "A transportation path is a Schienenbahn (Railway) if
and only if it transports trains."
:<<=>> (exists (?t Train) (transports ?f ?t)))
```

ontologies P and Q. Both ontologies are committing to a higher-level ontology for transportation shown in table 4. The transportation ontology in turn is committing to other ontologies as shown in figure 8.

Detection of the similarity relations is a major task of the reasoning system in figure 4. When given the above intensional definitions, a reasoning system (in this example PowerLoom) is able to detect similarity relations such as:

- "Street" defined in ontology P is a *specialization* of "Strasse" in ontology Q;
- "Highway" in ontology P is a *specialization* of "Strasse" in ontology Q;
- "Road" is a *specialization* of "Strasse";
- "Highway" is a *specialization* of "Schnellstrasse";
- "Railway" is *equal* to "Schienenbahn";
- "Railway" is *disjoint* from "Strasse"; etc.

The taxonomy of the tree is illustrated in figure 9.

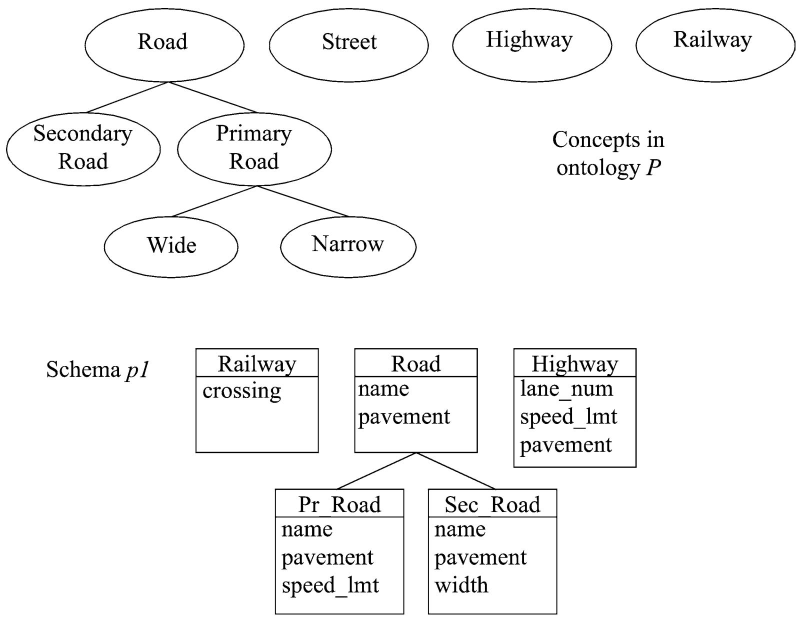


Figure 6. Schema $p1$ and the taxonomy tree of the ontology P.

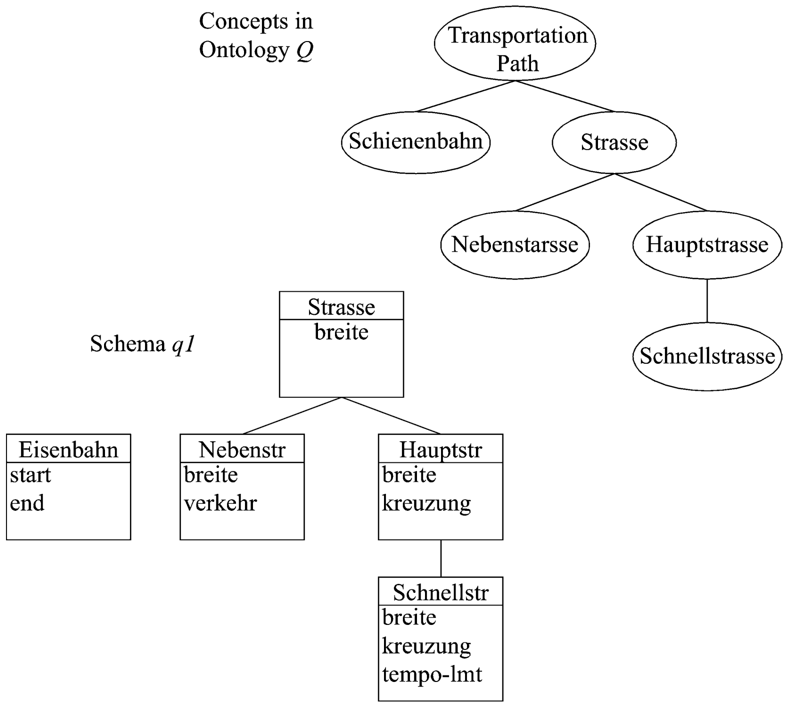


Figure 7. Schema $q1$ and taxonomy tree of ontology Q. Note that the definition of Transportation_Path is adopted from the transportation ontology.

Table 4
Part of a higher-level ontology (the transportation ontology) for both ontologies p and q.

```
(defconcept Transportation_Path (?tp)
:documentation "An individual is a transportation path if and only if
it transports wheeled vehicles.  A transportation path is surrounded by
two sidelines and there exists at least one lane belonging to it."
:<<=>> (exists (?wv Wheeled_Vehicle)
               (transports ?tp ?wv))
:=>>   (exists (?l Lane)
               (has-lane ?tp ?l))
:=>>   (exists (?ls Sideline)
               (left-surrounded-by ?tp ?ls))
:=>>   (exists (?rs Sideline)
               (right-surrounded-by ?tp ?rs)))

(defconcept Lane (?l)
:documentation "no formal definition is given.")

(defrelation has-lane ((?tp Transportation_Path) (?l Lane)
:documentation "has-lane is a relation between transportation path
and lanes.  It concludes that a transportation path is composed of
lanes.  (The definition of the composed-of relation is taken from a
higher-level ontology of aggregation which is not presented here.)")
:=>> (composed-of ?tp ?l))

(defconcept Sideline (?s)
:documentation "A transportation path is surrounded by a left and a
right side line."
:=>> (exists (?tp Transportation_Path)
             (or (left-surrounded-by ?tp ?s)
                 (right-surrounded-by ?tp ?s)))

(deffunction left-surrounded-by ((?tp Transportation_Path))
                                                     :->(?s Sideline)
:documentation "left-surrounded-by is a relation that maps
a transportation path to one sideline.  It concludes that a
transportation path is composed of the sideline."
:=> (composed-of ?tp ?ls))

(deffunction right-surrounded-by ((?tp Transportation_Path))
                                                      :->(?s Sideline)
:documentation "right-surrounded-by is a relation that maps a
transportation path to one sideline.  It concludes that transportation
path is composed of the sideline."
:=> (composed-of ?tp ?ls))

(deffunction transports ((?tp Transportation_Path))
                                              :-> (?vw Wheeled_Vehicle)
:documentation "transports maps every transportation path to one and
only one wheeled vehicle.")
```

Table 4
(Continued)

```
(deffunction belongs-to-path ((?l Lane)) :->(?tp Transportation_Path)
:documentation "belongs-to-path relates a lane to a transportation
path.  It concludes that the lane is part-of the transportation
path.  (The part-of relation is taken from a higher-level ontology of
aggregation not presented here.)"
:=>> (part-of ?tp ?l))

;; The following asserts that every lane belongs to a transportation
path.
(assert (forall (?l Lane) (exists (?tp Transportation_Path)
                                  (belongs-to-path ?l ?tp))))

;; The following asserts that every lane belongs to no more than one
transportation path.
(assert (forall (?l Lane) (= (cardinality (belongs-to-path ?l)) 1)))

;; The following two assertions state that belongs-to-path and has-lane
are inverse relations.
(assert (forall ((?l Lane) (?tp Transportation_Path))
                           (=> (belongs-to-path ?l ?tp)
                               (has-lane ?tp ?l))))

(assert (forall ((?l Lane) (?tp Transportation_Path))
                           (=> (has-lane ?tp ?l)
                               (belongs-to-path ?l ?tp))))

(deffunction vehicle_traffic ((?tp Transportation_Path)) :-> (?n Number)
:documentation "car per day maps every transportation path to one and
only one number.")

(deffunction speed_limit ((?tp Transportation_Path)) :-> (?n Number)
:documentation "speed limit maps every transportation path to one and
only one number.")

(defrelation intersect ((?x Transportation\_Path)
                        (?y Transportation\_Path))
:documentation "intersect relates two transportation path.")

(deffunction distance (?x ?y):->(?n Number)
:documentation "distance relates two individuals to a number.")

(deffunction width ((?tp Transportation_Path)) :-> (?n Number)
:documentation "width is distance between two sides of a transportation
path"
:<<=>> (= (distance (left-surrounded-by ?tp)
                    (right-surrounded-by ?tp)) ?n))
```

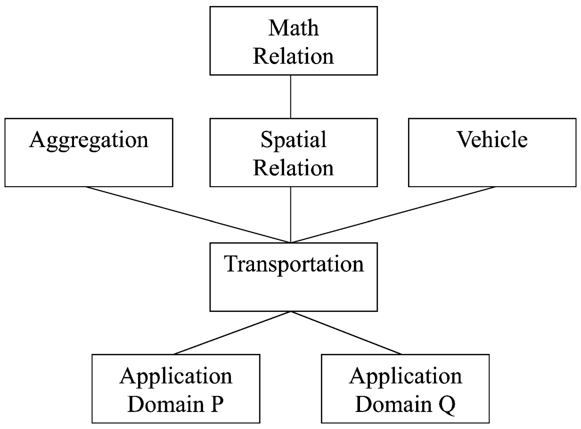


Figure 8. Organization of ontologies P and Q and their higher-level ontologies.

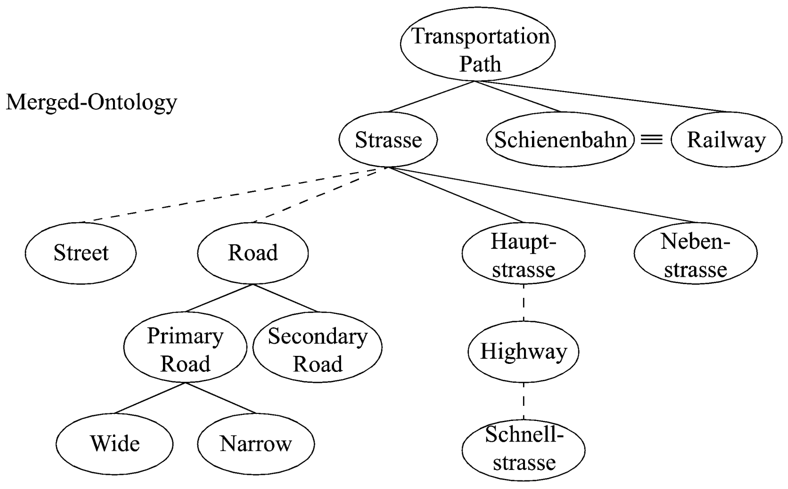


Figure 9. Result of merging formal ontologies by finding specialization similarities.

5. Global schema generation

This section shows how two database schemas (S_{p1} and S_{q1}) based on 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 classes and their 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 definitions in the formal ontology of the community.[2] As an example, class Pr_Road in schema S_{p1} (figure 6) is based on the term "Primary_Road" defined in a formal ontology P. We show this link to a term in ontology P with τ_p, e.g.,

$$\tau_p : (S_{p1}.\textsf{Pr\_Road}) \rightarrow \text{"Primary\_Road" or } \tau_q : (S_{q1}.\textsf{Nebenstr}) \rightarrow \text{"Neben\_Strasse".} \tag{5}$$

"τ" returns exactly one term in the respective ontology. "τ" is introduced to provide the flexibility in naming of schema elements. 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 schema elements in the other local schema.

For every class in the local schema we generate a class in the global schema. The goal is that every class in the local schema is represented by (or can be mapped to) a class in the global schema, which is important for the data mapping. We start by initializing the global schema S_G with the class hierarchy of S_{p1}. The classes of the schema S_{q1} are inserted into the global schema S_G during the following steps. The insertion of classes is performed in a stepwise and top-down manner by starling with root classes in the class hierarchy of the local schemas S_{q1}. For the insertion of any class, the following conditions are checked:

- A class (c) is added only then if no other class exists already in the schema whose concept is *equal* to the one c represents. That is, to add a new class such as "Strasse" to the global schema, the following condition must hold:

$$\forall c \in S_G \colon \neg\big[\tau_q(S_{q1}.\textsf{Eisenbahn}) \equiv \tau_p(S_G.c)\big]. \tag{6}$$

  For example, the class S_{q1}.Eisenbahn is not inserted since a class S_G.Railway based on an *equal* term ("Schienenbahn" $\equiv$ "Railway") is already present in the global schema. In this case, only an alias name Eisenbahn is stored for the existing class (this is needed during global attribute generation and data mapping).

- The specialization similarity of the concepts in the merged-ontology are reflected as a subclass relation in the global schema. Therefore, we establish a subclass (or super class) relation with every class based on a generalized (specialized) concept of the current class. As an example, the class S_{q1}.Strasse is defined as a superclass of S_G.Road and S_G.Highway considering that the following holds according to the merged-ontology:

$$\tau_p(S_G.\textsf{Road}) \leqslant \tau_q(S_{q1}.\textsf{Strasse}), \tag{7}$$
$$\tau_p(S_G.\textsf{Highway}) \leqslant \tau_q(S_{q1}.\textsf{Strasse}). \tag{8}$$

  This step can generate redundant subclass relations. For example, after S_{q1}.Strasse is inserted into the global schema, during insertion of class "S_{q1}.Hauptstr" the subclass

[2] Whether the definition of terms already exists in the formal ontology or 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.

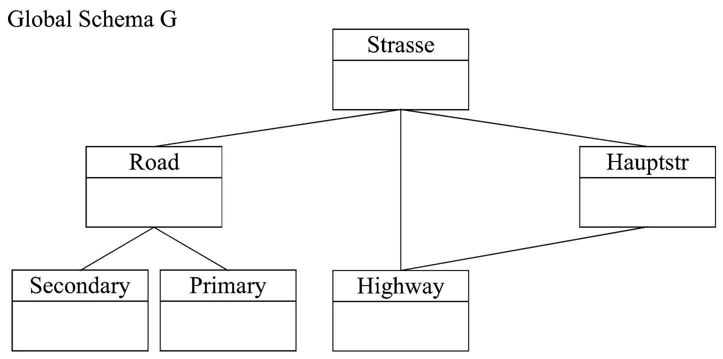


Figure 10. Occurrence of a redundant subclass relation while establishing new a subclass relation.

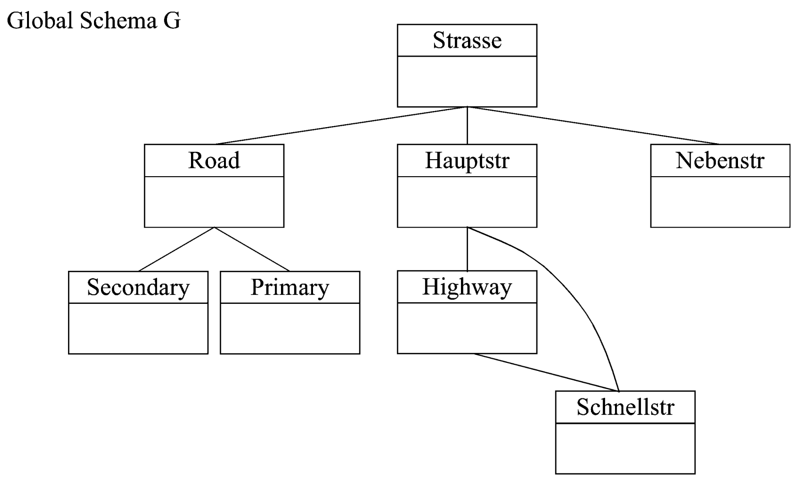


Figure 11. Occurrence of a redundant subclass relation while maintaining an existing subclass relation.

relation between S_G.Strasse and S_G.Highway is redundant (see figure 10). After generation of a subclass (or superclass) relation, such redundant relations are detected and eliminated.

- While inserting a subclass from the local schema (such as S_{q1}.Nebenstr), we maintain its subclass relation with the existing superclasses in the global schema (S_G.Strasse). However, maintaining such relations can cause duplicate relations, just as explained in the previous paragraph. For an example of such a case, see the insertion of S_{q1}.Schnellstr in figure 11.

There are also situations in which new classes are created in the global schema. Both cases need supervision of the database integration administrator:

- New subclass insertion: New classes may be added to the global schema if the base concepts of classes are overlapping. A class based on the conjunction concept of the two overlapping classes is added. As an example, when two classes in the global schema are based on two overlapping concepts:

$$\tau_p(S_{\mathrm{G}}.\mathsf{Nebenstr}) \approx \tau_p(S_{\mathrm{G}}.\mathsf{Sec\_Road}), \tag{9}$$

then a class (say safe_street) based on their conjunction concept can be added to the global schema. This class semantically represents roads with low traffic and a low speed limit. Subclass relations are established with both classes (a case of multiple inheritance). Although such cases often happen during the merging process, many of them are not relevant to applications. In our example, a road with low traffic and a low speed limit may not be of interest. Therefore, there is a need for supervision at this point. The database integration administrator should decide about the necessity of generating such classes. (We suggest creation of the conjunction concepts during the merging process in [18]. However, we did not find it advantageous.)

- New superclass insertion: 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 may also be generated in the global schema. As an example, the class S_{G}.Strasse and S_{G}.Eisenbahn are disjoint. We may add a superclass based on their superconcept "Transportation_Path". This action also needs a human decision. This is, the generation of the new class based on "Transportation_path" should be verified by the database integration administrator. (To further develop the approach, one may consider the logical disjunction of the intensional definitions as a new superclass of the two classes.)

A further extension to this approach is required to support association classes. An assumption we made at the beginning of this section is that classes are based on concepts. In practice, databases can contain classes based on relations. For instance, schema $p1$ may contain a class Crossing with an attribute traffic-light. At the ontology level we need to define a relation between a concept "traffic light" and relation "intersect" (in the Transportation ontology). The Description Logic formalism allows such definitions, and the similarity relation introduced in section 3 can be applied as well. However, supporting such definitions is a very complex task for a reasoning system. The inability of reasoning systems to accept such definitions and use them for reasoning let us rely on the aforementioned assumption for the time being. The final class hierarchy is illustrated in figure 12.

5.2. Filling classes with attributes

All attributes in the database schemas represent binary relations either by pointing to another class or by taking a primitive type such as string or integer. As we assume all classes to be based on concepts, we also assume attributes to be based on binary relation definitions in the respective ontology. For example:

$$\tau_q : (S_{q1}.\mathsf{Hauptstr.kreuzung}) \rightarrow \text{“strassenkreuzung”}, \tag{10}$$

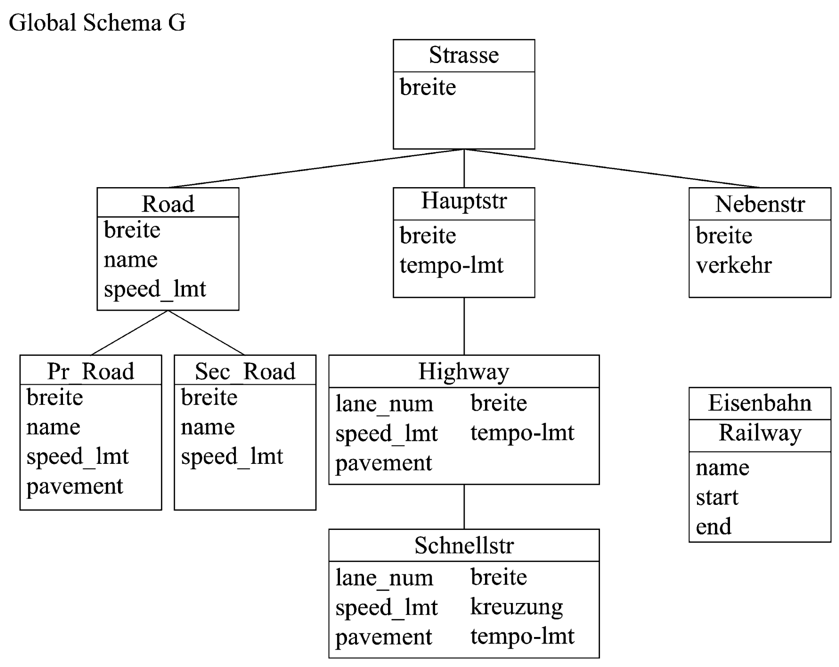


Figure 12. The final global schema.

states that an attribute "kreuzung" in class "Hauptstr" of local schema $q1$ is based on relation "strassenkreuzung" in ontology Q. There is a major constraint imposed while establishing links for attributes. The class of the attribute should not be based on a concept definition which is disjoint from the concept in the domain of its binary relation. For example in the following:

$$\tau_p : (S_{p1}.\text{Highway.pavement}) \rightarrow \text{"covered-by"}, \tag{11}$$

the domain of the "covered-by" relation must not be disjoint from $\tau_p(S_{p1}.\text{Highway})$. This constraint is for instance violated when defining an attribute based on the "covered-by" definition for class $S_{p1}.\text{Railway}$. Since "Railway" is disjoint from the domain of "covered-by", it does not match the semantics defined in the ontology. This constraint ensures that the schema definitions in S_{p1} (and the schema mapping τ_p) agree with the ontology P.

During the generation of attributes, for each attribute in a class of a local schema, we define an attribute in the respective class in the global schema. Thus, each attribute in a class of a local schema has a counterpart in the global schema. However, an *equality* relation might have already been represented by another attribute in the same global class. For example, before we add a new attribute "pavement" to the class "Highway" in the global schema, we check the following:

$$\forall a \in S_{\text{G}}.\text{Highway} : \neg\big[\tau_q(S_{p1}.\text{Highway.pavement}) \equiv \tau_p(S_{\text{G}}.\text{Highway}.a)\big]. \tag{12}$$

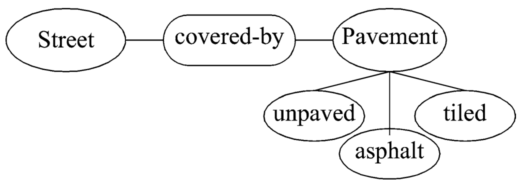


Figure 13. Example of a relation between a concept (street) and subconcepts of another concept (pavement) rather than its instances.

If this constraint is violated, a semantically *equal* (synonym) attribute has already been inserted into the class S_G.Highway in the global schema. We only keep information about the equality of the attributes for the data mapping. Unlike the case of synonym classes where we only keep an alias name, here we maintain both attribute definitions in the class. The reason is that the equality link is based on semantics of the attribute but does not indicate the similarity with respect to representation and data type (such as: unit, structure) of the attribute. We consider a data conversion during data mapping.

The fact that attributes are related by a specialization similarity relation is detected and stored so that it can be used during the data mapping. As an example, while defining attribute S_p.Highway.crossing for class S_G.Highway, the following similarity is detected and we keep the information about the type of relation between the two attributes for the data mapping phase:

$$\tau_q(S_{q1}.\mathsf{Strasse.kreuzung}) \leqslant \tau_p(S_{p1}.\mathsf{Highway.crossing}), \tag{13}$$

considering that:

$$\tau_p : (S_{p1}.\mathsf{Highway.crossing}) \to \text{“intersect”}. \tag{14}$$

We did not find the case of attributes based on overlapping relations relevant to this work. The detection of such cases by the reasoning system is not problematic, though.

The relation “covered-by” in our example is a special case (figure 13). In general, a relation between two concept definitions in ontologies specifies how instances of two concepts are related. However “covered-by” relates instances of a concept (e.g., street) with subconcepts of “Pavement” rather than an instance of “Pavement”. One can define “covered-by” as a relation between “Street” and instances of “PavementType” – as we did. However, we cannot define them like concepts and assign intensional definition to them. Necessary reasoning with such relation definition requires a reasoning system to support higher order logic. Therefore, in our intensional definition of “PavementType” in table 2, “Asphalt” is defined as an instance of “PavementType”. In such a case, attribute values are neither primitive (e.g., numbers or character strings) or objects but enumerations or codes that represent a set of possible individuals or ranges of values (e.g., “Asphalt” in figure 13, or “wide” or “narrow” as an enumeration of the possible values for the attribute width).

6. Data mapping

This section discusses problems arising during data mapping using ontologies. We mainly discuss potential problems encountered during the data mapping phase and discuss possible solutions.

The generated global schema is used for the integration of databases instantiating the local schemas. 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 acquired during the schema integration process. Afterwards, a set of operations performs the mapping of the data.

Classes in two local schemas referring to the same definition are mapped to the same global class by means of alias names. A potential problem may occur during the data mapping whenever both databases are providing instances that represent the same individual in the domain. In our example, one railway route may be stored in both databases DB_{q1} and DB_{q2}. For example, in query "give me all Railways" against the global schema, we must be able to detect those railways that are present in both databases. 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. For instance, for a railway route candidate identification criteria would be its location or its start and end points. Note that the identification criterion may not necessarily be the primary key in one or both systems (but it certainly should be a unique property). In case the identification criterion of a global class evaluates to true for two instances, such inconsistencies can be prevented or at least detected. If an identification criterion cannot be found, inconsistencies can occur and we risk returning two objects representing the same individual, while we cannot realize the redundancy.

In case classes are in a specialization relation in the global schema, all instances of a subclass can be mapped to its superclass, but not the other direction. As an example consider the query "Give me all Highways" (or highways from Germany), which results in the retrieval of data from database DB_{q1}. One solution is to retrieve the S_{G}.Schnellstr (*a.k.a.* substitutability) which is an incomplete result to the query. For the mapping from a superclass to its subclass we need a *classification criterion*, which offers a better result in this case. That means, in order to map instances of the superclass S_{G}.Hauptstr to the subclass S_{G}.Highway, the instances should satisfy a classification criterion. By referring to the intensional definitions in section 4, one can see that if an instance of S_{G}.Hauptstr has a speed-limit greater than 80 and more than four lanes, it can be mapped to class S_{G}.Highway. Finding the classification criteria and implementing the necessary mapping needs human interaction.[3] Most current reasoning systems offer the capability of classifying the instances on the fly. However, the reasoning system (in our case PowerLoom) additionally requires a powerful interface to the database(s). Furthermore, the

[3] Finding classification criteria seems a trivial activity for people but we could not make such query against the reasoning system (PowerLoom). However, PowerLoom is perfectly capable of classifying the instance according to the intensional definitions.

necessary data must be available in the database to render classification possible. In our example, the information on the number of lanes and the speed limit of the highway should be present in the database. In the case of the mapping of Hauptstr to Highway this information is however not available. The elaborations on the identification criterion apply here as well. For example, one road may be classified under both S_G.Hauptstr and S_G.Highway. While mapping S_G.Hauptstr object instances to S_G.Highway the identification criterion must be checked in order to ensure that the mapping is correct.

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 an *equality* similarity, the attribute values are mapped mutually. In case two attributes are related 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 intensional definitions in section 4, one can see that every "strassenkreuzung" relation is an "intersect" relation, but not the other way round. In general there are cases in which attribute values should be mapped by considering a classification criterion. This means that those crosses between "Strasse" can be mapped to "strassenkreuzung".

An attribute mapping often requires a data conversion process (e.g., integer to real or vector to raster). This is because during the integration we did not utilize any knowledge about the data types. There is often a need for further processing steps during data mapping, such as conversion 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). However, work such as [35] can perfectly suit this requirement and satisfy the need for data conversions. We intentionally avoid using any representation knowledge to guarantee that the similarity relations are established independently of their representation. Finally it is worth mentioning that unit conversion is also an issue at the ontology level – e.g., in definition of "width" or "speed limit".[4]

Furthermore, we only consider the mapping of one element of the schema to exactly one other element. More complicated data mappings in which a global class is an aggregation of classes in the local schemas, or an attribute is the result of a calculation over multiple attributes (as presented in [31]) are not treated here. This is because the focus of this work is on how to solve semantic problems in terms of finding similarities between classes and attributes and finding mappings between them. Determining correspondence of data values is out of the scope of this paper.

7. Related work

Hammer and McLeod [20] define a broad range of semantic conflicts and classify them into five categories. We consider *object comparability* in their terminology as semantic heterogeneity. Kim et al. [25,26] present a comprehensive classification of schema

[4] We approached the unit conversion problem by adopting and simplifying the ontology of Phisycal-Quantities present on the Ontoligua [11] library of ontologies. However, related code is not present in the example ontologies, here, for the sake of simplicity.

heterogeneity. They present solutions for several types of schema heterogeneity in relational and object-oriented database systems. Another comprehensive work in this area is presented by Garcia-Solaco et al. [14]. Both approaches address problems of schema heterogeneity, but do not distinguish between schematic and semantic issues. Miller et al. [31] distinguish schema integration from schema mapping. They focus on the mapping of data, taking the integrated schema as given. While we also distinguish phases of schema integration and data mapping, we focus mainly on the schema integration phase. Another effort in the area of data mapping is introduced by Rosenthal and Sciore [35], where they present an architecture for semantic interoperability.

The InfoSleuth project [2,32] (the successor of the Carnot project [36]) is one of the pioneer projects in the area of integration and addresses semantic integrity. Its architecture contains various types of agents, such as user agents, ontology agents, query agents and so forth. A user can select an ontology (via a user agent) from a list of ontologies offered by an ontology agent. Ontologies are built for every user group [13]. InfoSleuth offers the ability to build queries based on a selected ontology and therefore can match information or services with the user queries. The KRAFT project [23] uses shared ontologies as the basis for the mapping between ontology definitions and communication between agents. As a result of this project, [38] presents a set of ontology mismatches and establishes the mapping between a shared ontology and local ontologies. The COIN project [15] presents an architecture for semantic interoperability which inspired parts of this work. The role of the Domain Model in the COIN-architecture can be compared to that of an ontology in our approach. However, their Domain Model is essentially different from an ontology. OBSERVER [30] is a project using ontologies to allow queries against heterogeneous sources. It replaces terms in user queries with suitable terms in target ontologies, by means of inter-ontology relations. The inter-ontology relationship manager in OBSERVER maintains the same relations between ontological definitions as presented by Bergamaschi et al. in [4]. Unlike our approach, the relations are not derived from formalized definitions.

The four levels of similarity relations introduced in this paper are partly mentioned in previous work [9,27]. We present formal definitions both intensionally and extensionally. In this respect, the work of Larson et al. [27] is close to our work. However, we distinguish the ontological characteristics of attributes and their representational and implementational characteristics. For example, characteristics such as domain, uniqueness and cardinality are present in the relation definitions in an ontology, while security constraints or scale are representational characteristics. Elmasri and Navathe [9] introduce the relations using extensional definitions. Kashyap and Sheth [24] present a variety of different issues under the topic of semantic similarity. They present a thorough discussion on criteria for semantic similarity.

[4,29,34] propose schema integration approaches using thesauri. Bergamaschi et al. [4] introduce an approach for deriving similarity relations (synonym, hypernym and hyponym) from schema structures of component databases. The approach supports semiautomatic relation extraction and requires supervision of a domain expert. They introduce an algorithm to integrate schema definitions into a global homogeneous

schema based on the extracted relations. The Cupid project [29] proposes an approach for schema matching. This 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 enhance thesauri with a coefficient for every entry in the thesauri. It also categorizes schema elements into clusters (which in turn is similar to the approach described in [4]). Palopoli et al. [34] rely on the schema definitions of the component databases enhanced by knowledge of domain experts. Their approach requires two dictionaries, a synonymy dictionary like a thesaurus, and an inclusion dictionary extracted from schemas or domain experts. Similar to Cupid [29], domain experts customize both dictionaries by fuzzy coefficients. Furthermore, [7] also uses a thesauri-based approach for query translation which is suitable for Internet search engines querying both structured and semi-structured data.

In the above approaches [4,29,34], domain experts customize a thesaurus for an application-domain. Such thesauri neither help for the communication across application domains, nor do they foster intercommunity communication. For communication between two different application domains, coefficients of the thesauri entries must be set by an expert in both application domains. This is an important shortcoming in comparison to our approach utilizing ontologies. By using ontologies, we establish the *similarity relations* (defined in section 3), while by using a thesaurus synonym and hyponym relations must be provided by domain experts.

8. Conclusion

This paper presents an approach for generating global schemas. This solution uses formal ontologies as a basis for the integration and for resolving heterogeneity problems during the integration of local schemas into one global schema. We only use the intensional definitions in formal ontologies, that is, we rely neither on matching names of schema elements or the use of a thesaurus, nor do we need to consider the structure of the schema definitions [4,29,34].

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

- Completeness: Explicit specification of implicit assumptions in the community facilitates the reasoning process. If the ontologies do not offer detailed specifications, the reasoning system will not be able to detect similarity relations except overlap. If a term is only added to an ontology with no definition (that means no axiom is stated in the definition of the term), the overlap similarity relation with any other term will be detected. If the number of non-overlapping similarity relations detected is small, our approach will result in a low-quality integration, as it will rather be a union of schema definitions than a true integration.
- Accuracy: Accordance of an ontology to the *conceptualization* of the community guarantees that the result of the integration meets the expectations of users. If specifications in ontologies do not comply with the conceptualization then the similarity

relations will not be accurate. In turn, this will result again in a low-quality and imprecise integration.
The accuracy of the formalized ontologies depends on the capabilities of the formalism being used. Apart from the limitation of supporting higher-order logic by a reasoning system shown in figure 13, we faced further limitations during the ontology formalization phase. As can be seen from table 4, a phrase in the definition of "Transportation_Path" states that it transports at least one "Wheeled_Vehicle". The original phrase in the definition of the "Transportation_Path" should state that it *can* transport "Wheeled_Vehicles". That is, "Transportation_Path" possesses the ability to transport a "Wheeled_Vehicle". In other words, there is at least one possible world in which a "Transportation_Path" transports at least one "Wheeled_Vehicle". Such statements can be expressed by modal logic [6]. Although the lack of accuracy in this case did not have a negative impact on our example, supporting modal logic is an essential prerequisite for accurately formalizing 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 to guarantee the agreement of schema definitions with intentional definitions. One such constraint is discussed at the beginning of section 5.2.

Building a higher level ontology that many communities agree upon is a difficult and consequently expensive task. Major tasks are extracting the detailed specifications from members of the community and formalizing these specifications. However, it is a price worth paying to avoid semantic conflicts (which in many cases will be even more expensive). Furthermore, formal ontologies are long-term assets that will remain independent of the application systems. As formal ontologies are becoming more popular, they can also be used for other purposes than database integration.

Note that we considered the integration of classes and attributes only, while methods are not discussed in this paper. Methods are often considered as parametric attributes based on relation definitions of arity higher than two in a formal ontology. This allows us to specify methods that represent an *action* in the formal ontology (actions change the states of the world).

Acknowledgements

Work of Farshad Hakimpour has been funded by Swiss National Science Foundation, project no. 2100-053995.

References

[1] F. Baader et al. (eds.), *The Description Logic Handbook: Theory, Implementation and Applications* (Cambridge University Press, Cambridge, 2003).

[2] R.J. Bayardo et al., Infosleuth: Agent-based semantic integration of information in open and dynamic environments, in: *Proceedings of the ACM SIGMOD International Conference on Management of Data* (1997) pp. 195–206.

[3] V.R. Benjamins and D. Fensel, The ontological engineering initiative (KA)^2, in: *Formal Ontology in Information Systems, Proceedings of FOIS'98*, Trento, Italy, ed. N. Guarino (IOS Press, Amsterdam, 1998) pp. 287–301.

[4] S. Bergamaschi et al., An intelligent approach to information integration, in: *Formal Ontology in Information Systems*, ed. N. Guarino (IOS Press, Amsterdam, 1998) pp. 253–267.

[5] Y.A. Bishr et al., Probing the concept of information communities – A first step toward semantic interoperability, in: *Interoperating Geographic Information Systems*, eds. M. Goodchild et al. (Kluwer Academic, New York, 1999) pp. 55–69.

[6] P. Blackburn, M. de Rijke and Y. Venema, *Modal Logic* (Cambridge University Press, Cambridge, 2001).

[7] R. Domenig and K.R. Dittrich, A query based approach for integrating heterogeneous data sources, in: *Proceedings of 9th International Conference on Information and Knowledge Management* (November 2000).

[8] M.J. Egenhofer and J.R. Herring, Categorizing binary topological relations between regions, lines and points in geographic databases, Technical report, Department of Surveying Engineering, University of Maine, Orono (1991).

[9] R. Elmasri and S.B. Navathe, *Fundamentals of Database Systems*, 3rd Ed. (Addison-Wesley, Reading, MA, 2000) pp. 76–86.

[10] T. Erickson, Social interaction on the net: Virtual community as participatory genre, in: *Proceedings of the 13th Hawaii International Conference on System Science*, eds. J.F. Nunamaker and R.H. Sprague, Vol. 6 (IEEE Computer Society Press, Los Alamitos, CA, 1997) pp. 23–30.

[11] A. Farquhar, R. Fikes and J. Rice, The ontolingua server: A tool for collaborative ontology construction, International Journal of Human Computer Studies 46 (1997) 707–727.

[12] D. Fensel et al., On2Broker: Semantic-based access to information sources at the WWW, in: *Proceedings of the World Conference on the WWW and Internet (WebNet 99)* (October 1999) pp. 366–371.

[13] J. Fowler et al., Agent-based semantic interoperability in InfoSleuth, SIGMODRecord 28(1) (March 1999).

[14] M. Garcia-Solaco, F. Saltor and M. Castellanos, Semantic heterogeneity in multidatabase systems, in: *Object-Oriented Multidatabase Systems: A Solution for Advanced Applications*, eds. O.A. Bukhres and A.K. Elmagarmid (Prentice-Hall, Englewood, Cliffs, NJ, 1996) Chap. 5, pp. 129–202.

[15] C.H. Goh, S. Bressan, S. Madnick and M. Siegel, Context interchange: New features and formalisms for the intelligent integration of information, ACM Transactions on Information Systems 17(3) (1999) 270–290.

[16] N. Guarino, Formal ontology and information systems, in: *Formal Ontology in Information Systems, Proceedings of FOIS'98*, Trento, Italy, June 1998, ed. N. Guarino (IOS Press, Amsterdam, 1998) pp. 3–17.

[17] N. Guarino (ed.), *Formal Ontology in Information Systems* (IOS Press, Amsterdam, 1998).

[18] F. Hakimpour and A. Geppert, Resolving semantic heterogeneity in schema integration: An ontology base approach, in: *Formal Ontology in Information Systems: Collected Papers from the 2nd International Conference, FOIS'01*, eds. C. Welty and B. Smith (ACM Press, New York, 2001) pp. 297–308.

[19] F. Hakimpour and A. Geppert, Global schema generation using formal ontologies, in: *Proceedings of the 21st International Conference on Conceptual Modeling (ER2002)*, October 2002, eds. S. Spaccapietra, S.T. March and Y. Kambayashi, Lecture Notes in Computer Science, Vol. 2503 (Springer, Berlin, 2002) pp. 307–320.

[20] J. Hammer and D. McLeod, An approach to resolving semantic heterogeneity in a federation of autonomous, heterogeneous database systems, Journal for Intelligent and Cooperative Information Systems 2(1) (1993) 51–83.

[21] S. Handschuh and S. Staab, Authoring and annotation of web pages in cream, in: *The 11th International World Wide Web Conference (WWW2002)* (2002).

[22] J. Heflin and J. Hendler, Semantic interoperability on the web, in: *Extreme Markup Languages 2000* (2000).

[23] D. Jones, Developing shared ontologies in multi-agent systems, in: *ECAI'98 Workshop on Intelligent Information Integration*, Brighton, U.K. (August 1998).

[24] V. Kashyap and A. Sheth, Semantic similarities between objects in multiple databases, in: *Management of Heterogeneous and Autonomous Database Systems*, eds. A. Elmagarmid, M. Rusinkiewicz and A. Sheth (Morgan Kaufmann, Los Altos, CA, 1999) Chap. 3, pp. 57–90.

[25] W. Kim, I. Choi, S. Gala and M. Scheevel, On resolving schematic heterogeneity in multidatabase systems, Distributed and Parallel Databases 1(3) (July 1993) 251–277.

[26] W. Kim and J. Seo, Classifying schematic and data heterogeneity in multidatabase systems, IEEE Computer 24(12) (1991) 12–18.

[27] J.A. Larson, S.B. Navathe and R. Elmasri, A theory of attribute equivalence in database with application to schema integration, IEEE Transactions on Software Engineering 15(4) (1989) 449–463.

[28] R.M. MacGregor, H. Chalupsky and E.R. Melz, *PowerLoom Manual*, University of Southern California, http://www.isi.edu/isd/LOOM/PowerLoom/documentation/manual.pdf (1997).

[29] J. Madhavan, P.A. Bernstein and E. Rahm, Generic schema matching with cupid, in: *VLDB 2001, Proceedings of 27th International Conference on Very Large Databases*, Roma, Italy, September 11–14, 2001, eds. P.M.G. Apers et al. (Morgan Kaufmann, Los Altos, CA, 2001) pp. 49–58.

[30] E. Mena, V. Kashyap, A. Illarramendi and A. Sheth, Domain specific ontologies for semantic information brokering on the global information infrastructure, in: *Formal Ontology in Information Systems*, ed. N. Guarino (IOS Press, Amsterdam, 1998).

[31] R.J. Miller, L.M. Haas and M.A. Hernandez, Schema mapping as query discovery, in: *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases*, Cairo, Egypt, September 10–14, 2000, eds. A. El Abbadi et al. (Morgan Kaufmann, Los Altos, CA, 2000) pp. 77–88.

[32] M. Nodine, W. Bohrer and A.H. Hiong Ngu, Semantic brokering over dynamic heterogeneous data sources in InfoSleuth, in: *Proceedings of the 15th International Conference on Data Engineering*, Sydney, Australia, eds. M. Kitsuregawa, L. Maciaszek, M. Papazoglou and C. Pu (March 1999) pp. 23–26.

[33] OGC, *Semantics and Information Communities, The OpenGIS Abstract Specification Topic 14*, OpenGIS Consortium, version 4, edition (April 1999).

[34] L. Palopoli, D. Sacca and D. Ursino, Semi-automatic techniques for deriving interscheme properties from database schemes, Data and Knowledge Engineering 30(3) (1999) 239–273.

[35] A. Rosenthal and E. Sciore, Description, conversion, and planning for semantic interoperability, in: *Database Application Semantics, Proceedings of Conference on Data Semantics, IFIP WG6.2*, eds. R. Meersman and L. Mark (1995) pp. 140–164.

[36] M.P. Singh et al., The Carnot heterogeneous database project: Implemented applications, Distributed and Paralleled Databases 5(2) (April 1997) 207–225.

[37] P.R.S. Visser and Z. Cui, Heterogeneous ontology structures for distributed architectures, in: *ECAI-98 Workshop on Applications of Ontologies and Problem-Solving Methods* (1998) pp. 112–119.

[38] P.R.S. Visser, D.M. Jones, T.J.M. Bench-Capon and M.J.R. Shave, Assessing heterogeneity by classifying ontology mismatches, in: *Formal Ontology in Information Systems, Proceedings of FOIS'98*, Trento, Italy, June 1998, ed. N. Guarino (IOS Press, Amsterdam, 1998) pp. 148–162.

[39] C. Welty and B. Smith (eds.), *Proceedings of the International Conference on Formal Ontology in Information Systems*, Ogunquit, Maine, USA, October 2001 (ACM/SIGART, ACM Press, 2001).