

Semantic Conflict Resolution Ontology (SCROL): An Ontology for Detecting and Resolving Data and Schema-Level Semantic Conflicts

Sudha Ram, *Member, IEEE*, and Jinsoo Park, *Member, IEEE*

Abstract—Establishing semantic interoperability among heterogeneous information sources has been a critical issue in the database community for the past two decades. Despite the critical importance, current approaches to semantic interoperability of heterogeneous databases have not been sufficiently effective. We propose a common ontology called **Semantic Conflict Resolution Ontology (SCROL)** that addresses the inherent difficulties in the conventional approaches, i.e., federated schema and domain ontology approaches. SCROL provides a systematic method for automatically detecting and resolving various semantic conflicts in heterogeneous databases. SCROL provides a dynamic mechanism of comparing and manipulating contextual knowledge of each information source, which is useful in achieving semantic interoperability among heterogeneous databases. We show how SCROL is used for detecting and resolving semantic conflicts between semantically equivalent schema and data elements. In addition, we present evaluation results to show that SCROL can be successfully used to automate the process of identifying and resolving semantic conflicts.

Index Terms—Heterogeneous databases, ontology, semantic conflict resolution, semantic modeling.

1 INTRODUCTION

THE concept of ontology, which originates from philosophy, has been widely employed by several research communities. In the artificial intelligence (AI) community, ontologies have been used to capture domain knowledge for knowledge-based systems. The knowledge is typically represented in the knowledge-based system's representation language using the vocabulary provided by an ontology. In the distributed artificial intelligence (DAI) community, which includes research on distributed problem solving (DPS) and multiagent systems (MAS), ontologies have been accepted as an effective means to facilitate collaboration and communication among agents [38]. The need for ontologies has also been addressed in the information retrieval area to facilitate semantic information searching. Other areas, such as natural language processing (NLP), utilize ontologies to facilitate natural language generation and interpretation [20]. The database community is not an exception. In particular, research on distributed, heterogeneous databases has begun to exploit ontologies in order to support semantic interoperability.

In this paper, we present an ontology called SCROL (**Semantic Conflict Resolution Ontology**) that can be used to identify and resolve semantic conflicts among heterogeneous databases. Consider a state tax administrator, Mary Beth, who is interested in understanding how property tax assessments differ across various counties within her jurisdiction. To explore this problem, she may have to access many individual county databases simultaneously. However, a major impediment is that taxes are captured in different ways in each database. The Pima county database stores yearly tax amounts for each property, while Pinal county stores tax rates as a percentage of property value. Maricopa county, on the other hand, stores the monthly amount owed on each property. To be able to properly compare property taxes, Mary Beth has to understand these differences (also known as *semantic conflicts*) and know how to resolve them before she can attempt to compare the taxes. Similarly, there are a host of other semantic differences which need to be properly identified and resolved before databases can be used effectively. SCROL is intended to tackle this problem of recognizing and resolving semantic conflicts among multiple databases. While there are other tools and techniques such as Clio [21] and Cupid [18] that also provide semantic interoperability, our approach is different. These other systems will point out to the user that the tax amount in each database is similar. However, the user has the burden of understanding how they are similar and yet different, and then either manually resolve the differences, i.e., converting from one value to another, or calling on additional procedures to convert from one to another.

- S. Ram is with the Department of Management Information System, Eller College of Business and Public Administration, The University of Arizona, Tucson, AZ 85721. E-mail: ram@bpa.arizona.edu.
- J. Park is with the College of Business Administration, Korea University, 1, 5-Ga Anam-Dong, Sungbuk-Gu, Seoul, 136-701, Korea. E-mail: jinsoo.park@computer.org.

Manuscript received 6 Dec. 2000; revised 22 Dec. 2001; accepted 17 Oct. 2002.

For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 113254.

Our methodology using SCROL *automatically* identifies that the tax amounts in the three databases are similar, recognizes *how* they are similar and yet different, and then, converts them automatically so that the user is provided with a unified answer.

In order to make the *contents* of SCROL as general as possible and to enable it to be widely reused by a broad range of different application domains, the concepts represented in SCROL are neither data or application driven, nor are they domain specific. Since the main objective of our ontology is to facilitate detection and identification of various semantic and schematic conflicts, our ontology, unlike the Cyc project [16], is not intended to accumulate a massive knowledge base of human consensus knowledge. This goal enables us to develop a very simple yet flexible ontology. In fact, most currently existing ontologies cannot identify and accurately classify semantic conflicts [23]. Thus, our goal is to provide a robust mechanism for automating (to the extent possible) the semantic conflict identification and resolution process. SCROL has a simple structure, yet the results of our evaluation show that it is powerful enough to tackle a large variety of different types of semantic conflicts.

Semantic interoperability is the ability to integrate data sources developed using different vocabularies and different perspectives on data. To achieve semantic interoperability, systems must be able to exchange data in such a way that the precise meaning of the data is readily accessible and the data can be translated by any system into a form that it understands [31]. Establishing semantic interoperability among heterogeneous and disparate information sources has been a critical issue meriting active research within the database community for the past two decades [34]. Kashyap and Sheth identified an important issue for achieving semantic interoperability in a multidatabase environment [13]. This concerns the identification of semantically related data in different database systems and the subsequent resolution of the schematic differences among the semantically related data. A key aspect of identifying semantically similar data in different databases involves making semantics explicit [4]. Semantic similarity depends on the context in which a data object is used, and the contextual representation of a data object concerns how the data object is used [12]. Therefore, context is a critical element for capturing and representing similarities of data objects. Several techniques allow us to determine semantically similar objects. These include semantic modeling approaches, formal logic-based approaches, classifications of terminology, formal languages, knowledge-based systems, and the use of a shared ontology [34].

The problem of semantic interoperability has generally been tackled by one of two approaches: the federated schema approach and the domain ontology approach. The federated schema approach attempts to construct a federated (or global) schema and establish mappings between the federated schema and the participating local schemas. However, the drawback of this approach is that it was not designed to be independent of particular schemas or applications [32]. Thus, as new schemas or applications join a community or as potential conflicts emerge, one may

have to significantly modify the federated schema. On the other hand, the domain ontology approach uses a machine understandable definition of concepts and relationships between concepts so that there is a shared common understanding within a community. Its knowledge is domain specific, but independent of particular schemas and applications. However, one needs additional tools to actually capture and represent the knowledge (i.e., specifications and mappings) needed to resolve semantic conflicts.

Our hybrid approach is based on the use of an ontology that explicitly captures knowledge about different types of semantic conflicts and ways to resolve them. However, our ontology is domain independent. It is also independent of particular schemas and applications and, hence, provides a way to easily add new applications and schemas without having to significantly modify the federated schema.

Our paper is organized as follows: Section 2 explains the rationale behind and justification for using a common ontology approach. In Section 3, our common ontology, called **Semantic Conflict Resolution Ontology (SCROL)**, is defined. Section 4 highlights various features of the proposed semantic conflict detection and resolution methodology. In this section, we show the entire process of instantiating SCROL. The implementation of SCROL is based on the classification scheme of semantic conflicts, which provides a foundation for constructing SCROL. We also describe the ontology mapping and relationship knowledge, which are derived from the representation of SCROL. Section 5 presents examples demonstrating the use of SCROL. In addition, empirical results are discussed to evaluate the usefulness of SCROL. In Section 6, our approach is compared with previous approaches and our contributions are summarized. Finally, our future research directions are addressed in Section 7.

2 NEED FOR A COMMON ONTOLOGY TO FACILITATE SEMANTIC INTEROPERABILITY

In this section, we first examine the concept of *ontology*, a term commonly used in the fields of AI and knowledge management (KM). We then argue that the design of a common ontology is needed to facilitate interoperability in multiple heterogeneous systems. Ontology has been defined by various researchers as:

- The specification of a representational vocabulary for a shared domain of discourse, which may include definitions of classes, relations, functions, and other objects [9].
- A concept system in which all concepts are defined. Concepts are interpreted in a declarative way, as standing for the sets of their instances. This concept system is limitative in the sense that concepts can only be used if they are defined in the ontology. Definitions of concepts are formal where possible and informal otherwise [39].
- A model of some portion of the world, which is described by defining a set of representational terms [19].
- A means of achieving consistent communication between agents in multiagent systems [28].

- A collection of concepts and interconnections to describe information units [11].

We use the term *common ontology* as a vocabulary of representational terms (concepts) with agreed-upon definitions in the form of human readable text and machine-enforceable, declarative constraints (agent readable format) on their well-formed use [7].

Semantic interoperability requires resolving various *context-dependent* incompatibilities, i.e., semantic conflicts [23]. The *context* refers to the knowledge that is required to reason about another system for the purpose of answering a specific query [24]. Therefore, it is important to provide contextual knowledge of domain applications in order to ensure semantic interoperability. Siegel et al. [36] argue that the most basic requirement of the use of context for heterogeneous databases is the existence of common metadata vocabularies, so that any system in the enterprise can use such a common vocabulary to develop rules, i.e., context knowledge describing data semantics. In this approach, terminology outside of this common vocabulary must be translated to the common vocabulary; otherwise, the comparison of data semantics will not be possible [36]. Therefore, we believe that this approach is more practical than trying to agree upon broad-based standards for databases. Moreover, it is important to have automatic ways of comparing and manipulating the common vocabulary in order for a context knowledge representation to be useful for semantic interoperability among heterogeneous databases [13].

Consequently, in our framework, the common vocabulary that represents context knowledge is captured in the form of a common ontology, SCROL, which provides a systematic way of automatically detecting and dynamically resolving various semantic conflicts found in heterogeneous databases. In addition, all schema components captured by a common semantic data model, called USM* [29], are mapped to SCROL. The use of SCROL, in which all data objects have been mapped, has several advantages.

- It facilitates sharing and reuse [39].
- Mappings can be associated with each database and application, and can be applied by mediators [10].
- An administrator constructing or maintaining mappings needs to consider only his or her own data objects, not those in any other database or application programs [10].

Thus, a common ontology-based manipulation of complex and heterogeneous databases is one of the most desirable solutions for achieving semantic interoperability. There are, however, four important issues that need to be addressed. In working with a common ontology, Kahng and McLeod [11] discuss four of these.

- Contents: The contents of the common ontology are heavily affected by the semantic conflict types to be resolved.
- Construction and maintenance: The initial construction of a common ontology prior to any information sharing is a challenging problem. Furthermore, it is very important to allow the evolution of the common ontology.

- Mapping: Mapping from an information source to the common ontology is typically the most labor-intensive and time-consuming process, and is mostly carried out by domain experts.
- Relevance: The similarities and differences between two data objects from different databases or the relevance of exported information to a given request needs to be determined at some point within the information sharing activities.

The first and second issues can be resolved by establishing agreements on the meaning of the terms used in a common ontology, i.e., ontological commitment [9]. In order to make the contents of the common ontology as general as possible, and thus usable in various environments, we use a comprehensive classification framework of semantic conflicts [30], which gives clear guidelines for capturing semantic conflicts in various heterogeneous databases using a well-defined set of relationships between concepts to characterize application domains. The classification framework has been encoded into the common ontology, SCROL, and is described in more detail in Section 4.1. After its initial construction, we allow evolution of the common ontology. Since semantic conflict taxonomy is possibly, an approximation based on an agreement among participants, a system designed to solve the semantic conflicts is necessarily incremental and iterative [23]. Therefore, our approach (i.e., initial comprehensive construction and then the adoption of incremental and iterative evolution of the common ontology) is preferred because it allows the system to fine-tune and accommodate more contents as the system grows, while at the same time allowing the participating systems that may have evolving schemas to remain autonomous.

After the agreement has been reached, the next step (the third issue) is to establish mappings between information sources and the common ontology. Although the establishment of a comprehensive taxonomy and the mapping process may be difficult and time-consuming, being sharable and reusable by multiple heterogeneous environments probably justifies the extra effort in the design. The mapping is encoded in terms of the ontology mapping knowledge, as described in Section 4.3. We approach the fourth issue by specifying a structure of SCROL (Section 3) that allows the designer to describe how concepts are related to each other by specifying relationships between them in terms of the ontology relationship knowledge (Section 4.3). It provides initial linguistic links between concepts. In addition, the definitions of relationships in SCROL clearly lead to a single semantic interpretation of a given concept via semantic transformation between different contexts (Section 4.2). Detailed descriptions are given in later sections.

3 SCROL CONSTRUCTS AND DEFINITIONS

Gruber [8] maintains that formal ontologies are viewed as design artifacts, formulated for specific purposes and evaluated against object design criteria. He suggests five design criteria for ontologies whose purpose is knowledge

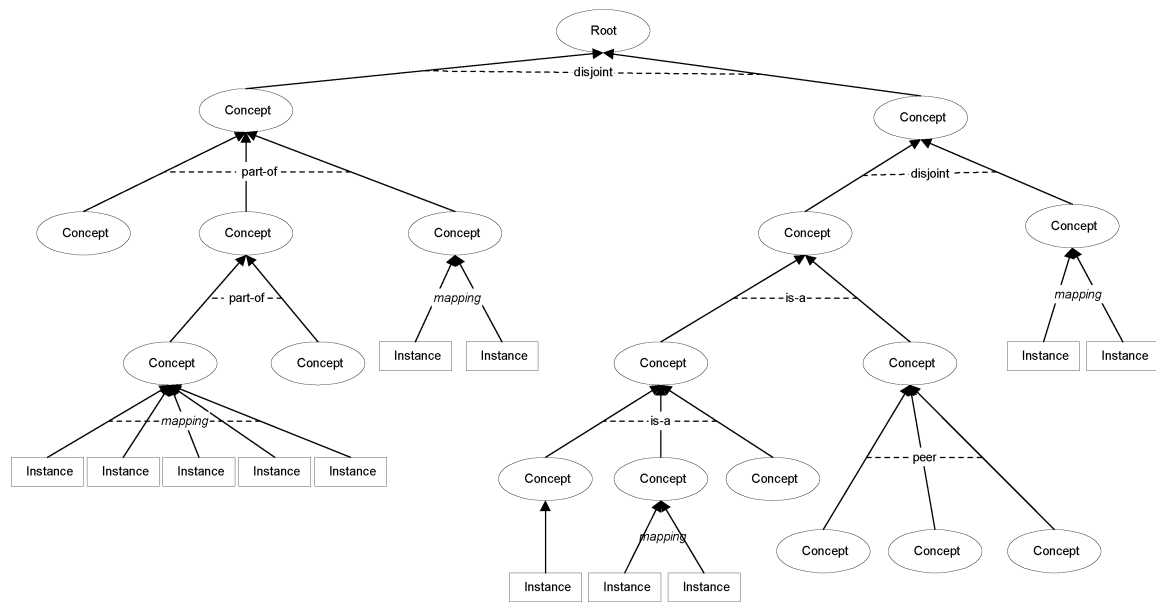


Fig. 1. Structure of SCROL.

sharing and interoperability among applications based on a shared conceptualization:

1. clarity (i.e., objective definitions),
2. coherence (i.e., logically consistent definitions),
3. extensibility (i.e., ability to define new terms based on the existing vocabulary without revisions of the existing definitions),
4. minimal encoding bias (i.e., specification of the conceptualization at the knowledge level without depending on a particular symbol-level encoding), and
5. minimal ontological commitment (i.e., specifying the weakest theory and defining minimal terms that are essential to the communication of knowledge consistent with that theory).

Ouksel, however, argues that it is not practically nor theoretically possible to develop and maintain an ontology that strictly adheres to these design criteria in an environment of autonomous, dynamic, and heterogeneous databases [23]. We believe that most currently existing ontologies have been developed mainly for the purpose of representing domain specific or commonsense knowledge, and they do not identify nor accurately classify semantic conflicts [23]. However, SCROL directly addresses this issue.

We define SCROL to provide a systematic method for automatically detecting and assisting in resolving various semantic conflicts in heterogeneous databases. Unlike other traditional ontology frameworks designed to capture domain specific [17], [19], [39] or commonsense knowledge [3], [16], [37], SCROL is developed to encode extensible knowledge on commonly found semantic conflicts that have been identified in our classification framework. It then provides an automatic way of comparing and manipulating contextual knowledge of each information source. Before we describe how SCROL can be structured to support semantic interoperability among heterogeneous databases, we first introduce and define the basic constructs of SCROL.

The structure of SCROL is a tree. We slightly extend and modify the basic definition of a tree to define *horizontal* relations (i.e., *sibling* relations and *domain value mapping* relations, which are discussed later).

Definition 1. A Semantic Conflict Resolution Ontology (SCROL) is a tuple $\Lambda = (OC, OI, RS, RM, u)$, where OC , OI , RS , RM , and u are as defined below. Its structure is graphically illustrated in Fig. 1, and the graphical notation of each SCROL construct is illustrated in Fig. 2.

Definition 2. OC is a distinct set of concepts. The oval shapes depicted in Fig. 1 represent concepts. A concept is related to instances in that a concept is a generalized abstract term that may have several concrete instances. For example, the term *Temperature* is a concept and *Fahrenheit* and *Celsius* are instances of *Temperature*. *Fahrenheit* and *Celsius* are two different specific expressions of *Temperature*. A concept may have zero or more children, and each child may be another concept or an instance. A concept that does not have any child concept is a leaf concept, i.e., a leaf concept may have one or more instances as children, but cannot have any concept as its child. Concepts have properties. Such properties are defined as follows:

- Name is a term that represents a concept.
- Definition is an agreed upon description of a concept written in plain, human-readable text.
- Subconcept is a property that is present in all concepts. This property contains a list of subconcepts

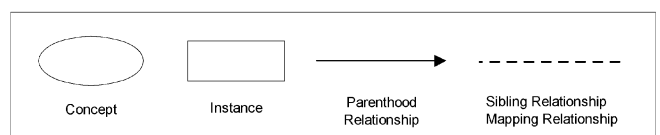


Fig. 2. Graphical notation of SCROL constructs.

(children) belonging to the concept. All leaf concepts have null values in this property because they do not have any subconcepts (but may have instances as stated above).

- Subconcept-of is a mandatory property of all concepts except the root. It contains its parent concept (called superconcept). A concept is allowed to have at most one parent concept.
- Instance is a property for only leaf concepts that have instances as their children. This property contains a list of instances belonging to the concept.
- Referenced-by is a property that is used to store mapping information about the underlying schema components (i.e., entity classes, relationships, and attributes) that are mapped to the concepts. This property is expressed as an ordered tuple $\langle s, o, t, l \rangle$, where s is a referring schema, o is a referring object id (oid), t is an object type (i.e., entityclass, relationship, or attribute), and l is a level of reference (i.e., by-type or by-value). The by-type means the object is referenced by its type and domain, while by-value refers to the data value itself. The default value of l is by-type.

Definition 3. *OI* in Λ is a distinct set of instances. The rectangles depicted in Fig. 1 represent instances. Every instance has exactly one concept as its parent. Instances do not have children. Instances have properties. Such properties are defined as follows:

- Name is a term that represents an instance.
- Definition is an agreed upon description of an instance written in plain, human-readable text.
- Instance-of is a mandatory property of all instances. It contains its parent concept. An instance has exactly one parent concept.
- Referenced-by is a property that is present in every instance. It is used to store mapping information about the underlying schema components (i.e., entity classes, relationships, and attributes) that are mapped to the instances. This property is expressed as an ordered tuple $\langle s, o, t, l \rangle$, where s is a referring schema, o is a referring object id (oid), t is an object type (i.e., entityclass, relationship, or attribute), and l is a level of reference which is only by-type. Since the mapping by-value is only used to establish mappings for schematic discrepancies, a type of schema-level conflicts, instances are never referenced by the mapping by-value. An example related to schematic discrepancies is demonstrated in Section 5.2.

Definition 4. *RS* refers to a sibling relationship and is a relation on *OC*. *RS* can occur only between two concepts, but not between two instances. *RS* consists of a disjoint relationship, a peer relationship, a part-of relationship, and an is-a relationship and has the following form: $\langle x, y, F \rangle$, where $x, y \in OC$, and $F = \{\text{disjoint}, \text{peer}, \text{is-a}, \text{part-of}\}$. Note that disjoint and peer are symmetric, but is-a and part-of are asymmetric. They are all transitive. The dotted vertical lines between concepts depicted in Fig. 1 represent

sibling relationships with proper labels indicating a disjoint, peer, part-of, or is-a relationship.

- The disjoint relationship between two concepts indicates that they are not semantically equivalent. For example, the concepts Distance and Temperature have a disjoint relationship because they are not semantically equivalent.
- The peer relationship is used when two concepts are semantically equivalent, that is, two concepts represent the same real world object; thus, it is possible for the given two concepts to define one-to-one mapping between all the instances of these two concepts. Therefore, instances belonging to the two concepts can be transformed into each other through semantic transformation rules. In this case, the domain value mapping relationships (shown in the next definition) between all instances of such concepts are always one-to-one. For instance, as illustrated in Fig. 3, the subconcepts of a concept Localized Time have peer relationships. They are peers because in every instance, they can have one-to-one semantic mappings; thus, they can be transformed into each other in a given context. For instance, the transaction time of stock trades recorded in Seoul can be converted to the local time in Bombay.
- The part-of relationship is similar to an aggregation in semantic data models and object-oriented data models. For example, the concept City is a part-of Urban Area which is a part-of County. The concept County is again a part-of the concept State/Providence, etc.
- The is-a relationship is the same as generalization/specialization in semantic data models and object-oriented data models. For instance, the concept Water can have several specialized concepts, such as Ground Water and Surface Water.

Note that we do not have overlapping relationships because they can be integrated and depicted through generalization, i.e., is-a relationship. The disjoint and peer relationships are used by mediators to determine whether semantic conflicts exist (particularly data-level conflicts) and, if semantic conflicts exist, whether they are resolvable. The purpose of using part-of and is-a relationships is to allow mediators to detect schema-level conflicts between schemas.

Definition 5. *RM* in Λ is a relation on *OI* and called a domain value mapping relationship (or, briefly, a mapping relationship). The dotted vertical lines across parenthood relationships between instances depicted in Fig. 1 represent mapping relationships. By definition, *RM* can occur only between instances; it cannot occur between concepts. Another property of *RM* is that all instances belonging to a concept may be regarded as synonyms of their parent concept. This is the case where instances have different names for the same concept. Similarly, if two instances have the same name but belong to different concepts, they are homonyms. Note that all mapping relationships described below are derived from functional mappings in set theory. The mapping relationships are used by mediators to determine whether the actual data values that are mapped to instances can be transformed from

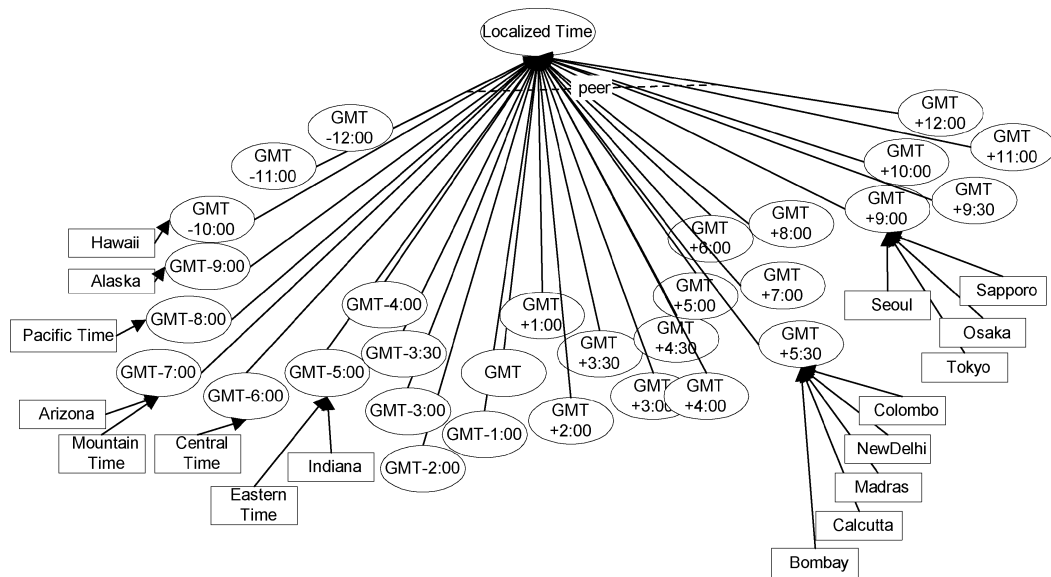


Fig. 3. Peer relationship example.

one value to another and vice versa. RM consists of one-one, one-many, many-many, or none. In addition, total and partial mappings are used in combination of one, one-many, and many-many relationships to indicate whether every value in one instance has a corresponding value in other instances. The corresponding notation and description of each mapping relationship is presented in Table 1.

Definition 6. u is the root of Λ . The root u has no parent. By definition, there is exactly one u in Λ .

4 A METHODOLOGY FOR SEMANTIC CONFLICT DETECTION AND RESOLUTION

We have implemented a software system called CREAM (Conflict Resolution Environment for Autonomous Mediation). CREAM can be used to model and access heterogeneous databases via a Web-based graphical user interface [26]. CREAM provides various collaborative components, such as Schema Designer, Schema Mapper, Ontology Designer, Ontology Mapper, and Semantic Mediators, which enable the automation of conflict detection and resolution through their interaction with SCROL. Details about the integrated environment, CREAM, and its components can be found in Park and Ram [26]. SCROL constitutes a central part of CREAM. It has been implemented in Java using Oracle 9i as a repository. In the following sections, the process of instantiating SCROL using our CREAM system is described.

4.1 Defining SCROL Using Ontology Designer

The Ontology Designer, a major component of CREAM, helps the database administrator define various concepts, instances, and their relationships by describing and classifying different types of semantic conflicts in SCROL. Such concepts, instances, and their relationships are stored in a set of tables owned by SCROL. A built-in integrity checker prevents errors during the ontology design. Remember that SCROL is domain independent. Hence,

the definition of SCROL is a *one time effort* and the resulting ontology can essentially be used repeatedly in many application domains. SCROL needs to be extended only if the database administrator discovers that there is a new type of a semantic conflict not previously defined in SCROL.

Since the discovery and reconciliation of semantic conflicts necessitate the ability to classify these conflicts, we construct SCROL based on a comprehensive classification framework of semantic conflicts [30]. This classification framework is based on extensive field study and investigation of various real geographic data sets (i.e., US Geological Survey data, vegetation data, land use data, etc.) containing several million records. We have defined additional types of semantic conflicts that are regularly encountered in geographic databases. We also tested each type of semantic conflict using these data sets to examine the extent to which CREAM automates the semantic conflict resolution process. The classification framework characterizes semantic conflicts at two different levels, i.e., data and schema level conflicts, each having six types of semantic conflicts. In most instances, data-level conflicts are differences in data domains caused by the multiple representations and interpretations of similar data. Schema-level conflicts are, on the other hand, characterized by differences in logical structures and/or inconsistencies in metadata (i.e., schemas) of the same

TABLE 1
Notation of Mapping Relationships

Mapping Relationship	Notation
total one-one	$x \leftrightarrow y$
total one-many	$x \leftrightarrow \rightarrow y$
total many-one	$x \leftarrow \rightarrow y$
total many-many	$x \leftrightarrow \leftrightarrow y$
partial one-one	$x \leftrightarrow \dots y$
partial one-many	$x \leftrightarrow \dots \rightarrow y$
partial many-one	$x \leftarrow \dots \rightarrow y$
partial many-many	$x \leftrightarrow \dots \leftrightarrow y$
none	$x \not\leftrightarrow y$

TABLE 2
Data Level Conflicts

Conflict	Description
Data value conflict	Different interpretations of the meaning of data instance values
Data representation conflict	Similar objects are described by different data types or data format representations
Data unit conflict	Use of different measurement units
Data precision conflict	Implementation of different scales, different domain precision, or different data granularities and resolutions
Known data value reliability conflicts	Data present in different databases may be subject to data reliability (i.e., measurement of error, measuring instruments, precision of measurements, topological properties, and treatment of time dimension)
Spatial domain conflict	Specifications of geographic regions or objects are "differently" but "legally" defined by different people

TABLE 3
Schema Level Conflicts

Conflict	Description
Naming conflict	Labels of schema elements (i.e., entity classes, relationships, and attributes) are somewhat arbitrarily assigned by different database designers (homonyms and synonyms)
Entity identifier conflicts	Assignment of different identifiers (primary keys) to the same concept in different databases
Schema isomorphism conflicts	Same concept (entity class) is described by a dissimilar set of attributes (i.e., the same concept is represented by a number of different attributes) or is not set operation compatible
Generalization conflicts	Different design choices for modeling related entity classes
Aggregation conflicts	When an aggregation is used in one database to identify a set of entities in another database
Schematic discrepancies	When the logical structure of a set of attributes and their values belonging to an entity class in one database are organized to form a different structure in another database

application domain. A more detailed description of the classification framework is found in Ram et al. [30]. Using the Ontology Designer, we defined SCROL to capture all of the types of conflicts represented in Tables 2 and 3.

The various data level and schema level conflicts are encoded as the first-level children concepts within SCROL's tree. For example, Fig. 4 shows the various data level conflicts encoded within SCROL. Due to space limitations, only the data precision conflicts within the data level conflicts are expanded. All other types of conflicts are also encoded as children concepts and instances of these concepts to represent the various semantic conflicts that may arise in the domain of the heterogeneous databases. This common ontology is later mapped to the individual local schemas and also to the federated schema so that each type of semantic conflict may be detected as needed.

4.2 Defining Conflict Controller, Original Context, and Target Context

In order to detect a certain semantic conflict, three pieces of information are needed from SCROL: a conflict controller, an original context, and a target context. The *conflict controller* is a concept that has child concepts or instances. Each of these represents multiple interpretations of semantically related objects. Each conflict controller has a corresponding semantic resolver that handles semantic reconciliation between the conflict controller's child concepts or instances (see Fig. 5). The *original context* is a

concept or an instance that has been mapped to the local schema component to represent the specific context of the local schema component. The *target context* represents the resulting context in which the user wants to transform the meanings of data values from local databases.

From the previous example, a concept Area has two instances, Square Meter and Acre, and the mapping relationship between the two instances is *total one-one*. Thus, any data value mapped to Square Meter can be converted to the corresponding value in Acre and vice versa. As illustrated in Fig. 5, the instance Square Meter is called *original context* of COUNTY_AREA.area because the attribute area is mapped to the instance Square Meter and actual data values stored in the area are represented in square meters (m²). Similarly, the instance Acre is also called *original context* of the attribute gross_size in CITY_SIZE. The parent concept of Square Meter and Acre (i.e., Area) is referred to as a *conflict controller*, which is mapped to REGION.area in the federated schema. When a user retrieves data sources mapped to Square Meter and wants to convert his output to Acre, the instance Acre becomes a *target context*. In this particular case, the conflict controller, Area, has a corresponding semantic resolver that contains two simple conversion rules, one converting values from square meter to acre and the other converting values from acre to square meter. The semantic resolver then captures conflict resolution knowledge about how to transform data values from COUNTY_AREA.area, which is originally in square meter, to values in acre (i.e.,

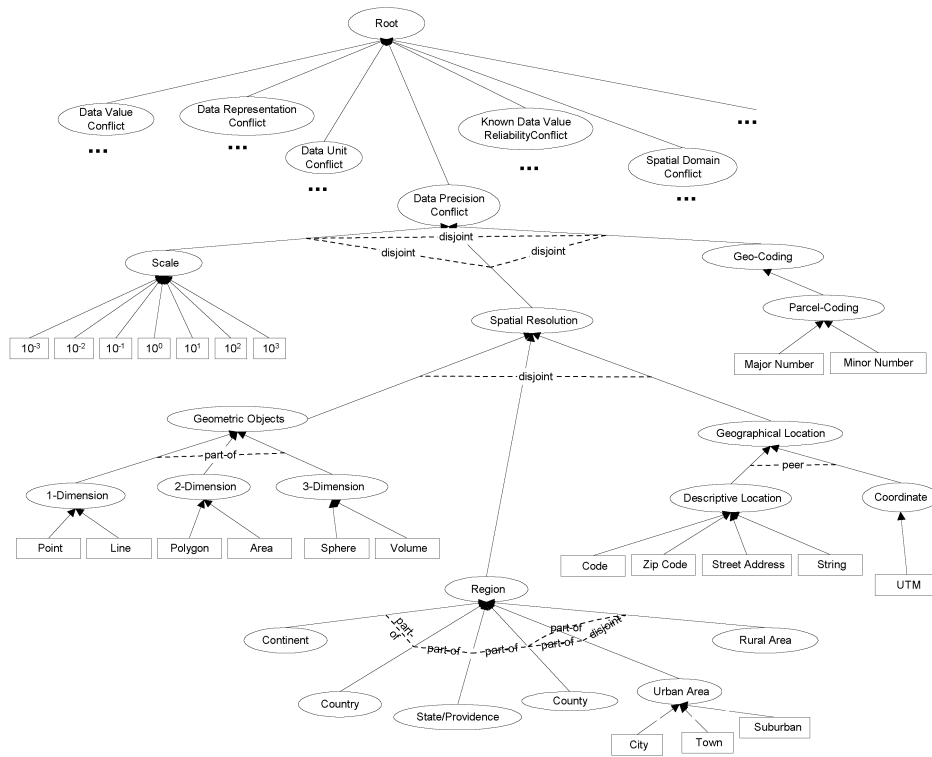


Fig. 4. Data precision conflicts in SCROL.

1 acre = 4,047 square meters). If a new instance (e.g., Square Feet) is added to the concept (e.g., Area) later, a domain expert creates associated new semantic transformation rules and encodes such rules into the appropriate semantic resolver.

4.3 Defining the Ontology Relationship and Mapping Knowledge

In order to determine whether two terms are semantically related and, if they are related, to what extent one can be translated to another, it is necessary to explicitly describe the relationship between SCROL components. Thus, the next step requires the construction of the *ontology relationship knowledge* defined as follows.

Definition 7. Let *ORK* be the ontology relationship knowledge. *ORK* is defined as a relation on $\sigma \times \sigma \times \lambda$, where $\sigma \in OC \cup OI$ in Λ and $\lambda \in RS \cup RM$ in Λ (see Definitions 1, 2, 3, 4, and 5). It is given by $ORK \subseteq \sigma \times \sigma \times \lambda$. Note that *ORK* is derived from *RS* and *RM*.

According to Fig. 6, Acre and Square Meter are specific instances of concept Area in SCROL, and these two instances have a *total one-one* mapping because every value in Acre is associated with exactly one value on in Square Meter and vice versa. In this example, the ontology relationship knowledge can be expressed as:

$$ORK = \{(Square\ Meter, Acre, \longleftrightarrow), (City, County, part-of)\}.$$

After the construction of *ORK*, the next step is to establish mapping knowledge between SCROL and database schemas. Contextual knowledge in a database

component is captured via mappings between SCROL and the schema of the component. Therefore, the *ontology mapping knowledge* is defined as follows.

Definition 8. Let *OMK* be the ontology mapping knowledge. *OMK* is a set of mappings between schema (both federated and local schemas) and SCROL. *OMK* is, therefore, defined as a relation on $SCROL \times (FS \cup LS)$ and given by $OMK \subseteq SCROL \times (FS \cup LS)$, where *FS* is a federated schema and *LS* is a set of participating local schemas.

For example, as Fig. 6 illustrates, CITY_SIZE.gross_size is mapped to an instance Acre, which specifies that the measurement unit of CITY_SIZE.gross_size is acre. On the

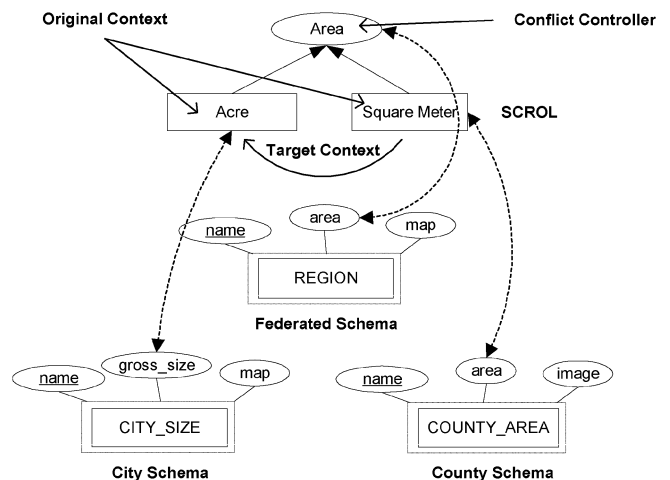


Fig. 5. Conflict controller, original context, and target context.

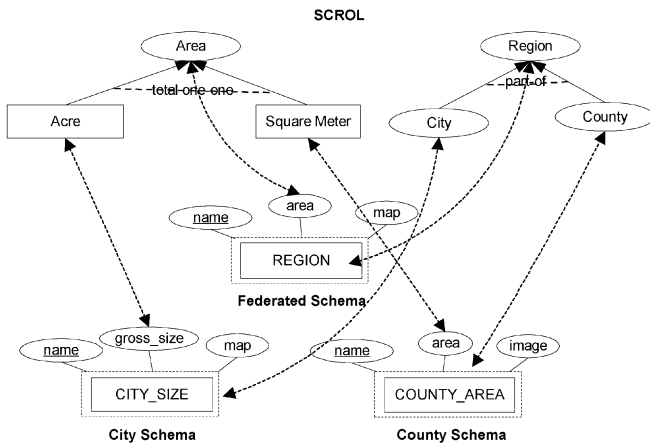


Fig. 6. An example of ontology mappings.

other hand, COUNTY_AREA.area is mapped to Square Meter, which indicates that the measurement unit of COUNTY_AREA.area is square meter. By looking at the given ontology mapping knowledge, the system is able to detect the fact that the two attributes are represented in different measurement units. Also, note that REGION.area in the federated schema is mapped to concept Area. This particular mapping knowledge indicates that REGION.area is mapped to a conflict controller and, thus, its measurement unit could be either acre or square meter. It allows the user to select his or her target context (either acre or square meter in this case), which is the resulting context in which the user wants to view the retrieved data. The ontology mapping knowledge can be established at any level (e.g., attribute, entity, or relationship level). At each level, either a concept or instance in SCROL is mapped to a schema component. For example, the entity CITY_SIZE is mapped to concept City and the entity COUNTY_AREA is mapped to concept County, where the City is *part-of* Country. The established mappings are stored in the ontology-mapping

repository. The ontology mapping knowledge can be explicitly represented in the following form:

OMK =
 {(Area, REGION.area), (Acre, CITY_SIZE.gross_size),
 (Square Meter, COUNTY_AREA.area), (Region, REGION),
 (City, CITY_SIZE), (County, COUNTY_AREA)}.

5 ILLUSTRATIVE EXAMPLES

In this section, we describe how SCROL is used to detect and resolve a variety of data and schema conflicts. Due to the space limitations, only one example of a data-level conflict and one example of a schema-level conflict are described in detail.

5.1 Data Level Conflict

Two heterogeneous databases from the land-use heterogeneous databases case are introduced in Fig. 7. They contain information on land parcels and buildings. Note that Fig. 7 displays only the attributes and entities that are necessary to explain data precision conflicts. However, schema mappings that are already established between the federated schema and two local schemas are not shown in Fig. 7 due to limited space. LAND_PARCEL.temperature in local schema 1 and LAND_PARCEL.avg_temperature in local schema 2 are semantically equivalent, and both are mapped to LAND_PARCEL.avg_temperature in the federated schema. Similarly, CI_BUILDING.gross_area and RES_BUILDING.area_in_square_feet in local schema 1 and CI_BUILDING.gross_area_ci and RES_BUILDING.total_housing_unit_area_rb in local schema 2 are mapped to CI_BUILDING.gross_area and RES_BUILDING.area_in_square_feet in the federated schema, respectively.

There are three data unit conflicts in this example:

- The measurement unit of LAND_PARCEL.temperature in local schema 1 is Celsius, whereas the unit of

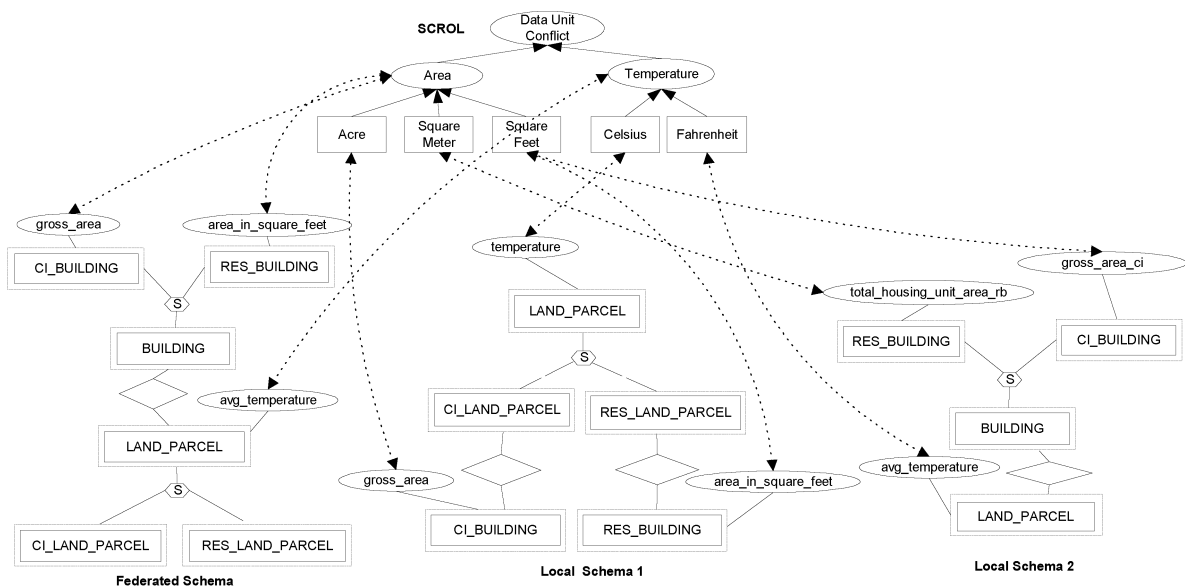


Fig. 7. An example of data unit conflicts.

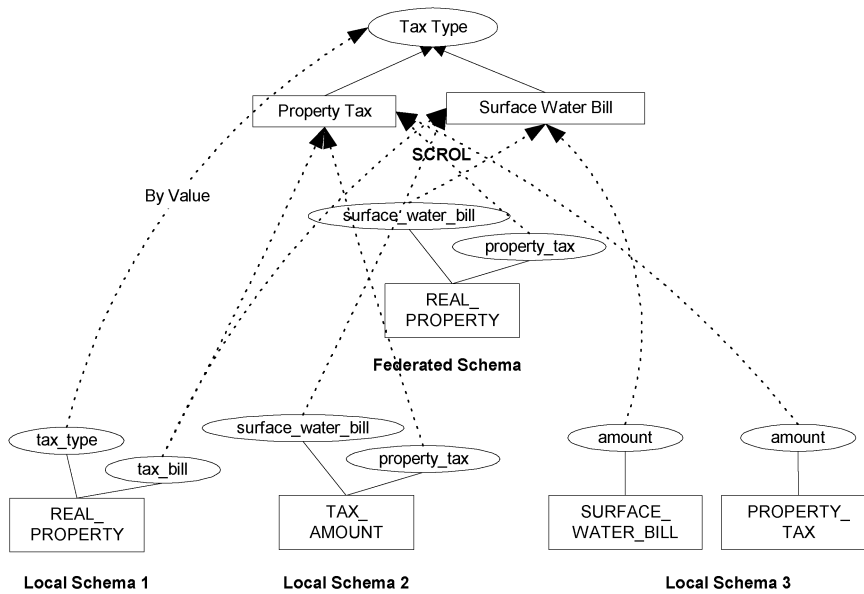


Fig. 8. An example of schematic discrepancies.

LAND_PARCEL.avg_temperature in local schema 2 is Fahrenheit.

- CI_BUILDING.gross_area in local schema 1 is in acres, while CI_BUILDING.gross_area_ci in local schema 2 is in square feet.
- RES_BUILDING.area_in_square_feet in local schema 1 is in square feet, whereas RES_BUILDING.total_housing_unit_area_rb in local schema 2 is in square meters.

As depicted in Fig. 7, these conflicts are identified by ontology mappings. In SCROL, Area and Temperature are conflict controllers, and they have a *disjoint* relationship between each other. Recall that the *disjoint* relationship between two concepts indicates that they are not semantically equivalent. Acre, Square Feet, and Square Meter have *total one-one* mapping relationships because their data values can be transformed from one to another without losing their meanings. Similarly, Celsius and Fahrenheit have a *total one-one* mapping relationship.

Based on the ontology mapping of the local schemas to SCROL, we may readily recognize data unit conflicts since each of the local schemas have attributes distinctly mapped to different instances of a concept. For instance, LAND_PARCEL.temperature in local schema 1 is mapped to the Celsius instance of the Temperature concept, whereas LAND_PARCEL.avg_temperature in local schema 2 is mapped to the Fahrenheit instance of the same concept. After the initial detection of the semantic conflict, the appropriate transformation rule may be used to resolve the conflict and convert the data values to any one of the data units captured in SCROL that represents the same concept. For example, if the user queries LAND_PARCEL.avg_temperature from the federated schema, the two conflicting contexts (i.e., Celsius and Fahrenheit) are recognized from the conflict controller, Temperature, and the user is asked to specify his or her target context. If the user wants to view the data in Celsius from the two underlying local databases, the user can select Celsius as a target context. For the local

schema 1, no semantic reconciliation process is required because the original and target contexts are identical. On the other hand, the original and target contexts of LAND_PARCEL.avg_temperature in schema 2 are different. Therefore, each data value retrieved from local schema 2 is converted to data value in Celsius according to the transformation rules and then the resulting data retrieved from both local schemas are presented to the user. The other two conflicts can also be detected easily and resolved in a similar way.

5.2 Schema Level Conflict

Consider the three different databases representing information about taxes shown in Fig. 8. All describe the sum of collected tax amounts (in thousands) of different taxes each year for a particular county. In local schema 1, REAL_PROPERTY stores one tuple per year per tax type (REAL_PROPERTY.tax_type) with amount of tax paid. TAX_AMOUNT in local schema 2 contains one tuple per year, and one attribute for each tax type (TAX_AMOUNT.surface_water_bill and TAX_AMOUNT.property_tax), where the value of the attribute is the tax amount paid. Local schema 3, in contrast, has one relation for each tax (SURFACE_WATER_BILL and PROPERTY_TAX) and each relation has one tuple per year with its tax amount paid (SURFACE_WATER_BILL.amount and PROPERTY_TAX.amount). Hence, this situation is a typical schematic discrepancy conflict.

As shown in Fig. 8, these conflicts are identified by ontology mappings. Note that the ontology mapping for the attribute, tax_type, in local schema 1 is defined as mapping-by-value because each tuple value itself is a tax type; that is, either property tax or surface water bill. During the semantic reconciliation, the Conflict Resolver determines the tax type in the local schema 1 by looking at each data value stored in the REAL_PROPERTY.tax_type and then assigns each data value captured in the REAL_PROPERTY.tax_amount to the corresponding attribute in the

federated schema. In the case of local schema 3, the name of each tax type is the name of an entity class. SURFACE_WATER_BILL.amount is mapped to Surface Water Bill and PROPERTY_TAX.amount is mapped to Property Tax in SCROL. Thus, if the user queries tax amounts of surface water bills through REAL_PROPERTY.surface_water_bill from the federated schema, then only data from SURFACE_WATER_BILL.amount in local schema 3 is retrieved.

Similarly, there are other types of more complex semantic conflicts such as the one on property tax assessment mentioned in the Introduction. This type of complex conflict consists of both schema and data level conflicts. Our methodology can automatically identify that it contains both types of conflicts, further process it to see which specific types of schema and data conflicts exist, and then resolve all of them automatically by employing SCROL.

6 EVALUATION OF SCROL

As mentioned earlier, due to the space limit, we show only two examples in detail: one from a data-level conflict and one from a schema-level conflict. However, SCROL can handle more complex data-level and schema-level conflicts, and the detailed analyses of the test results can be found in Park [25]. We tested SCROL using three different cases, each of which represents a different application domain. The first case, called land-use heterogeneous databases, involved three heterogeneous databases used in land-use applications. These databases were developed to store information about buildings constructed in residential or commercial and industrial areas. We adopted the heterogeneous database schemas originally described in Nyerges [22] and modified and expanded them to include as many semantic conflicts as possible. The second case, ecology heterogeneous databases, involved a set of databases used by ecologists for ecological system analysis at a large research organization. It consisted of three databases designed for different purposes. The first captured soil-related information, the second was designed for fire simulation, while the third was used to capture vegetation and different water types. The third case, called publication databases, involved two databases for publications, each of which was independently developed by researchers and librarians. The two database schemas were originally used by Batini and Lenzerini [1] to demonstrate schema integration methodology. We modified and extended the two schemas to evaluate and demonstrate the performance of our system.

The first two cases are categorized as being in the geographic database domain, while the third deals with nongeographic databases. The two geographic database cases were selected because temporal and spatial dimensions of geographic data objects are frequent and significant sources of semantic conflicts that lead to multiple interpretations [23]. Finally, the publication case was also selected in order to validate the effectiveness of SCROL in a nongeographic domain. We selected the specific cases from a variety of databases because the three cases were representative of commonly used geographic and nongeographic databases and the three cases combined cover all of

TABLE 4
Summary of Semantic Conflicts Resulting from the Three Cases

Conflict Type		Number of Conflicts Found	Number of Conflict Resolved
Data-Level Conflicts	Data Value	2	2
	Data Representation	5	5
	Data Unit	6	6
	Data Precision	4	4
	Known Data Value Reliability	6	0*
	Spatial Domain	1	0*
Schema-Level Conflicts	Naming	20	20
	Entity Identifier	2	2
	Schema Isomorphism	7	7
	Generalization	3	3
	Aggregation	6	6
	Schematic Discrepancies	6	6

* only partially supported

the different types of semantic conflicts in the classification framework described in Section 4.1. We found several semantic conflicts that were common across these cases, allowing for reuse of knowledge among different domains. The initial construction of SCROL consists of about 120 concepts and 160 instances.

Three different cases consisting of 123 relations and 434 attributes were tested to evaluate SCROL. Among the total of the 557 database components (i.e., the sum of 123 relations and 434 attributes) used during the evaluation phase, 201 components involved semantic conflicts. The 201 heterogeneous components caused 68 conflicts: 24 data-level conflicts, and 44 schema-level conflicts. Table 4 shows a summary of the results obtained from the three cases.

All of the conflicts found at the data level except for known data value reliability and spatial domain conflicts were resolved by the semantic mediators using SCROL and ontology mapping knowledge. Known data value reliability conflicts cannot be resolved semantically because they are not the cause of multiple interpretations of the same data, i.e., semantic heterogeneity. The differences in data source reliability are generally due to the use of different measuring instruments, which affect the precision and accuracy of data being collected. Spatial domain conflicts are also not easily resolved by the semantic mediators since differences in spatial domain specifications are legally defined by different people who have different needs in different application domains. However, as long as source database schemas are properly mapped to the matching ontology components and these conflicts can be classified in SCROL, the semantic mediators can automatically detect differences between them and provide a partial solution by notifying the user which information sources have conflicts. Unlike data-level semantic conflicts, semantic reconciliation of some schema-level conflicts requires both SCROL and schema mapping knowledge. Schema mapping knowledge contains all of the details about local schema structures [26]. The result also shows that both schema mapping knowledge and ontology mapping knowledge play important roles as knowledge sources for semantic mediators during global query processing. In conclusion, SCROL plays a pivotal role in our proposed approach.

7 CONTRIBUTIONS AND COMPARISON WITH OTHER APPROACHES

The major contribution of SCROL is that it is a domain independent ontology with a simple structure. However, it is powerful enough to identify and resolve a large number of different types of semantic conflicts.

The SCROL approach is different from domain ontology, which attempts to resolve semantic conflicts by using domain specific knowledge. It is very difficult to construct and manage domain knowledge because it is much more complex than a conventionally shared database schema. A domain model requires capturing tacit knowledge within its particular domain in great detail in order to clarify the meaning of each data object. To reduce this overhead, many researchers in the AI community have explored knowledge sharing and reusability [27]. Along with these endeavors, the need of ontology standardization has been addressed. To date, however, there is no formal agreement on these efforts. Furthermore, this approach requires a significant amount of reasoning with a domain model whenever it attempts to resolve even a simple semantic conflict because the model itself is inherently complex. We believe, however, that a domain model does not need to be complex if its purpose is only to detect and resolve semantic conflicts. This can be accomplished by ignoring all tacit knowledge relevant to a particular domain application.

Context Interchange (COIN) uses a domain model [5]. The domain model in COIN is a collection of primitive types and *semantic types* (similar to *type* in the object-oriented paradigm), which define the application domain corresponding to the data sources that are to be integrated. Each type in a domain model binds a specific context (semantic value) that is provided by local data sources when a query occurs. The data value is exchanged from one system to another by converting the semantic value from its source context to its receiving context through a context mediator. Conversion functions are used to convert semantic values from one context to another. A domain model (i.e., shared-ontology) specifies terminology mappings. These mappings describe naming equivalencies among the component information systems, so that references to attributes, meta-attributes, and their values in one information system can be translated to the equivalent names in another. This approach, however, focuses solely on the semantics of individual data items, i.e., data-level conflicts. Since the context information is represented at the level of data values, the context mediation proposed in this approach is not able to resolve semantic conflicts at the schema level. Thus, COIN will not be able to resolve the property tax assessment conflict described in the introduction of this paper.

OBSERVER is another ontology-based system that uses domain-specific metadata and contextual coordinates to map contextual descriptions to the database schema [14]. The construction of ontologies is based on domain specific terminologies that can be expressed in description logic. The difference with our work is that, while OBSERVER exploits interontology interoperation and, thus, focuses mainly on the translation of queries over different ontologies, SCROL can handle data-level semantic translation of underlying multiple heterogeneous and

distributed information sources. OBSERVER will be able to resolve naming differences between databases but not the other types of conflicts mentioned in Table 4.

In the federated approach, two methodologies have been proposed: the tightly coupled approach and the loosely coupled approach [35]. The distinction is based on who manages the federation and how the component databases are integrated. In the loosely coupled approach, a federation is created and managed by the user and there is no centralized control over the system. Users can directly interact with local databases instead of being restricted to querying federated schemas. One of the main drawbacks of this approach is that it places too heavy a burden on users by requiring them to understand the underlying local databases and provides little or no support for identifying semantic conflicts [6]. This approach typically requires users to engage in the detection and resolution of semantic conflicts. A federated system is tightly coupled if the federation is created and maintained by administrators, not by users. One or more federated schemas are constructed from the schemas of the participating local databases. This approach provides uniform and integrated access to the underlying local databases because the federated schema serves as a front-end system that supports a canonical data model and a single global query language on top of these local database systems. In this approach, however, semantic conflicts must be identified and resolved a priori by the administrators. Consequently, many researchers have identified a scalability problem in these approaches [5], [33].

Cupid [18] is an approach for schema matching. Cupid takes two schemas as input and returns a mapping that discovers equivalent elements in the two schemas. This approach uses a thesaurus to identify the similarity of schema element names (linguistic matching) and a structure matching algorithm to measure the structural similarity (structure matching). Thus, Cupid will be able to identify that the property tax assessment fields in different databases are all similar; however, it may not be able to convert from one to the other. The Cupid approach is schema-based and provides semiautomatic identification of schema-level conflicts between heterogeneous database schemas. In contrast, our approach tries to establish mappings at both data and schema levels, and to resolve semantic conflicts at both levels. We believe, however, that Cupid complements SCROL because, by automating the schema matching process, Cupid can automatically (or semiautomatically) generate the schema mapping knowledge between a federated schema and a local schema, which can reduce the time required to construct the schema mapping knowledge. Similarly, Clio [21] provides a set of tools and techniques to manage and transform heterogeneous databases. However, we believe it does not cover the whole range of schema and data conflicts shown in Table 4.

It is worth noting that there is an interesting approach that produces interoperable queries over heterogeneous object and relational databases. Chang and Raschid [2] use mapping knowledge to resolve semantic conflicts in heterogeneous schemas expressed in a canonical representation. F-logic [15] is used to represent the canonical representations for the queries. A set of transformation algorithms is defined to resolve semantic conflicts between local and remote schemas by accessing the relevant mapping information. While both the approach used by

Chang and Raschid [2] and our own approach utilize mapping knowledge and deal with semantic translation discussed in this paper, Chang and Raschid rely on "parameterized" templates as the canonical representations to encode the mapping knowledge, and focus on the interoperability between object and relational queries. Our approach, on the other hand, uses a shared ontology and mapping knowledge to facilitate semantic interoperability.

Our paper presents a hybrid approach by addressing the limitations of these approaches. By automating, the process to detect and resolve a wide variety of schema and data conflicts. We use a federated schema, but our approach requires only a reasonable amount of schema integration effort because semantic conflicts do not have to be identified and resolved a priori by an administrator. The specific context of each data object in the schema is identified through mappings to the extendable, domain independent ontology, and the local database administrators and domain experts are responsible for encoding semantic conflict classification knowledge.

8 CONCLUSION AND FUTURE RESEARCH

We have presented details of an ontology, SCROL, which can be used to identify and resolve semantic conflicts among multiple heterogeneous databases. We have also presented the results of an evaluation of SCROL to show how it comprehensively identifies and resolves many different types of conflicts. Important areas for further study include exploring the use of UML to represent SCROL as well as a full-fledged comparison of our CREAM system with other major operational systems such as Cupid and Clio.

There are several other interesting extensions of this research, which we are currently exploring. The research presented in this paper assumes that the underlying information sources are structured data which may reside in structurally organized text files or database systems. However, the unprecedented growth of Internet technologies has made vast amounts of resources instantly accessible to various users via the World Wide Web. The majority of data available on the Internet are semistructured or unstructured, consisting of multimedia data such as audio, video, and image files as well as textual documents, using a variety of markup languages such as HTML and variants of XML among others. However, even with the use of XML, capturing context knowledge on the meaning of each tag for semantic interoperability among heterogeneous XML formats is still required. We are currently investigating XML as a standard method for exchanging the knowledge captured in SCROL with other systems on the Internet.

We are also extending our work to semantic interoperability in digital libraries. A digital library is a networked system environment where individual components interact to allow users to submit and access digital content. Among the many issues and research directions in digital libraries identified by Ram et al. [31], ensuring interoperability and knowledge-based information sharing is one of the key aspects of successful implementation. Mediator and agent-based business transaction on the Internet (i.e., electronic commerce or EC) is another interesting extension of the current research. In the multiagent systems (MAS) environment, it is natural to deal with heterogeneous agents that

attempt to communicate with one another in different agent communication languages (ACL). Since each ACL is differentiated from others in terms of its syntax as well as semantics (e.g., FIPA-ACL and KQML), the interoperability issue in this area becomes interesting. We propose to extend these issues as an important direction for extending the research described in this paper.

ACKNOWLEDGMENTS

The authors would like to thank the associate editor and the anonymous reviewers. Their valuable comments and suggestions greatly improved the quality of this paper. Jinsoo Park was partially supported by a grant from the SK Fund to the College of Business Administration, Korea University.

REFERENCES

- [1] C. Batini and M. Lenzerini, "A Methodology for Data Schema Integration in the Entity Relationship Model," *IEEE Trans. Software Eng.*, vol. 10, no. 8, pp. 650-664, Nov. 1984.
- [2] Y.-H. Chang and L. Raschid, "Producing Interoperable Queries for Relational and Object-Oriented Databases," *J. Intelligent Information Systems*, vol. 14, no. 1, pp. 51-75, Mar. 2000.
- [3] C. Collet, M.N. Huhns, and W.-M. Shen, "Resource Integration Using a Large Knowledge Base in Carnot," *Computer*, vol. 24, no. 12, pp. 55-62, Dec. 1991.
- [4] S. Dao and B. Perry, "Applying a Data Miner to Heterogeneous Schema Integration," *Proc. First Int'l Conf. Knowledge Discovery and Data Mining*, pp. 63-68, Aug. 1995.
- [5] C.H. Goh, S. Bressan, S.E. Madnick, and M.D. Siegel, "Context Interchange: New Features and Formalisms for the Intelligent Integration of Information," *ACM Trans. Information Systems*, vol. 17, no. 3, pp. 270-293, July 1999.
- [6] C.H. Goh, S.E. Madnick, and M.D. Siegel, "Context Interchange: Overcoming the Challenges of Large-Scale Interoperable Database Systems in a Dynamic Environment," *Proc. Third Int'l Conf. Information and Knowledge Management*, pp. 337-346, 1994.
- [7] T.R. Gruber, "The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases," *Proc. Second Int'l Conf. Principles of Knowledge Representation and Reasoning*, pp. 601-602, 1991.
- [8] T.R. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing," Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford Univ., 1993.
- [9] T.R. Gruber, "A Translation Approach to Portable Ontology Specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199-220, June 1993.
- [10] M.N. Huhns and M.P. Singh, "Managing Heterogeneous Transaction Workflows with Co-Operating Agents," *Agent Technology: Foundations, Applications, and Markets*, N.R. Jennings and M.J. Wooldridge, eds., pp. 219-239, Berlin: Springer, 1998.
- [11] J. Kahng and D. McLeod, "Dynamic Classificational Ontologies: Mediation of Information Sharing in Cooperative Federated Database Systems," *Cooperative Information Systems: Trends and Directions*, M.P. Papazoglou and G. Schlageter, eds., pp. 179-203, San Diego, Calif.: Academic Press, 1998.
- [12] V. Kashyap and A.P. Sheth, "Semantic and Schematic Similarities Between Objects in Databases: A Context-Based Approach," Technical Report TR-CS-95-001, Dept. of Computer Science, Univ. of Georgia, 1995.
- [13] V. Kashyap and A.P. Sheth, "Semantic and Schematic Similarities Between Database Objects: A Context-Based Approach," *The VLDB J.*, vol. 5, no. 4, pp. 276-304, Dec. 1996.
- [14] V. Kashyap and A.P. Sheth, "Semantic Heterogeneity in Global Information Systems: The Role of Metadata, Context and Ontologies," *Cooperative Information Systems: Trends and Directions*, M.P. Papazoglou and G. Schlageter, eds., pp. 139-178, San Diego, Calif.: Academic Press, 1998.
- [15] M. Kifer and G. Lausen, "F-Logic: A Higher-Order Language for Reasoning about Objects, Inheritance, and Scheme," *Proc. ACM SIGMOD Int'l Conf.*, pp. 134-146, 1989.

- [16] D.B. Lenat, R.V. Guha, K. Pittman, D. Pratt, and M. Shepherd, "CYC: Toward Programs with Common Sense," *Comm. ACM*, vol. 33, no. 8, pp. 30-49, Aug. 1990.
- [17] R. MacGregor, "The Evolving Technology of Classification-Based Knowledge Representation Systems," *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, J.F. Sowa, ed., pp. 385-400, San Mateo, Calif.: Morgan Kaufmann, 1991.
- [18] J. Madhavan, P.A. Bernstein, and E. Rahm, "Generic Schema Matching with Cupid," *Proc. VLDB Conf.*, pp. 49-58, Sept. 2001.
- [19] K. Mahalingam and M.N. Huhns, "An Ontology Tool for Query Formulation in an Agent-Based Context," *Proc. Second IFCIS Int'l Conf. Cooperative Information Systems*, June 1997.
- [20] K. Mahesh and S. Nirenburg, "A Situated Ontology for Practical NLP," *Proc. IJCAI-95 Workshop Basic Ontological Issues in Knowledge Sharing*, Aug. 1995.
- [21] R.J. Miller, M.A. Hernandez, L.M. Hass, L. Yan, C.T.H. Ho, R. Fagin, and L. Popa, "The Clio Project: Managing Heterogeneity," *SIGMOD Record*, vol. 30, no. 1, pp. 78-83, Mar. 2001.
- [22] T.L. Nyerges, "Schema Integration Analysis for the Development of GIS Databases," *Int'l J. Geographical Information System*, vol. 3, no. 2, pp. 153-183, 1989.
- [23] A.M. Ouksel, "A Framework for a Scalable Agent Architecture of Cooperating Heterogeneous Knowledge Sources," *Intelligent Information Agents: Agent-Based Information Discovery and Management on the Internet*, M. Klusch, ed., pp. 100-124, Berlin: Springer, 1999.
- [24] A.M. Ouksel and C.F. Naiman, "Coordinating Context Building in Heterogeneous Information Systems," *J. Intelligent Information Systems*, vol. 3, no. 2, pp. 151-183, Apr. 1994.
- [25] J. Park, "Facilitating Interoperability among Heterogeneous Geographic Database Systems: A Theoretical Framework, A Prototype System, and Evaluation, Dissertation," PhD thesis, Dept. of Management Information Systems, Univ. of Arizona, 1999.
- [26] J. Park and S. Ram, "A Conflict Resolution Environment for Autonomous Mediation Among Heterogeneous Databases," Technical Report 01-05, Univ. of Minnesota, Apr. 2001.
- [27] R.S. Patil, R.E. Fikes, P.F. Patel-Schneider, D. McKay, T. Finin, T. Gruber, and R. Neches, "The DARPA Knowledge Sharing Effort: Progress Report," *Proc. Third Int'l Conf. Principles of Knowledge Representation and Reasoning*, pp. 777-788, Oct. 1992.
- [28] J.-C.R. Pazzaglia and S.M. Embury, "Bottom-Up Integration of Ontologies in a Database Context," *Proc. Fifth KRDB Workshop*, May 1998.
- [29] S. Ram, J. Park, and G. Ball, "Semantic Model Support for Geographic Information Systems," *Computer*, vol. 32, no. 5, pp. 74-81, May 1999.
- [30] S. Ram, J. Park, K. Kim, and Y. Hwang, "A Comprehensive Framework for Classifying Data- and Schema-Level Semantic Conflicts in Geographic and Non-Geographic Databases," *Proc. Ninth Workshop Information Technologies and Systems*, pp. 185-190, Dec. 1999.
- [31] S. Ram, J. Park, and D. Lee, "Digital Libraries for the Next Millennium: Challenges and Research Directions," *Information Systems Frontiers*, vol. 1, no. 1, pp. 75-94, July 1999.
- [32] S. Ram and V. Ramesh, "Schema Integration: Past, Current and Future," *Management of Heterogeneous and Autonomous Database Systems*, A. Elmagarmid, M. Rusinkeiwicz, and A.P. Sheth, eds., pp. 119-155, San Francisco: Morgan Kaufmann, 1999.
- [33] V. Ramesh, K. Canfield, S. Quirologico, and M. Silva, "An Intelligent Agent-Based Architecture for Interoperability among Heterogeneous Medical Databases," *Proc. Second Am. Conf. Information Systems*, pp. 549-551, Aug. 1996.
- [34] A.P. Sheth, "Semantic Issues in Multidatabase Systems," *SIGMOD Record*, vol. 40, no. 4, pp. 5-9, Dec. 1991.
- [35] A.P. Sheth and J.A. Larson, "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," *ACM Computing Surveys*, vol. 22, no. 3, pp. 184-236, Sept. 1990.
- [36] M. Siegel, S. Madnick, and E. Sciore, "Context Interchange in a Client-Server Architecture," *J. Systems Software*, vol. 27, no. 3, pp. 223-232, Dec. 1994.
- [37] V.C. Storey, R.H.L. Chiang, D. Dey, R.C. Goldstein, and S. Sundaresan, "Database Design with Common Sense Business Reasoning and Learning," *ACM Trans. Database Systems*, vol. 22, no. 4, pp. 471-512, Dec. 1997.
- [38] H. Takeda, K. Iwata, M. Takaai, A. Sawada, and T. Nishida, "An Ontology-Based Cooperative Environment for Real World Agents," *Proc. Int'l Conf. Multiagent Systems*, pp. 353-360, Dec. 1996.
- [39] P.E. van der Vet and N.J.I. Mars, "Bottom-Up Construction of Ontologies," *IEEE Trans. Knowledge and Data Eng.*, vol. 10, no. 4, pp. 513-526, 1998.



Sudha Ram received the BS degree in mathematics, physics and chemistry from the University of Madras in 1979, PGDM from the Indian Institute of Management, Calcutta, in 1981, and the PhD degree from the University of Illinois at Urbana-Champaign, in 1985. She is Eller Professor of management information systems in the Eller School of Business and Public Administration at the University of Arizona. Dr. Ram has published articles in such journals as *Communications of the ACM*, *IEEE Expert*, *IEEE Transactions on Knowledge and Data Engineering*, *Information Systems Research*, *Information Science*, and *Management Science*. Her research deals with modeling and analysis of database and knowledge-based systems for manufacturing, scientific, and business applications. Her research has been funded by IBM, NCR, US ARMY, NIST, US National Science Foundation, NASA, NIH, and ORD (CIA). Specifically, her research deals with interoperability among heterogeneous database systems, semantic modeling, data allocation, schema and view integration, intelligent agents for data management, and tools for database design. She is the director of the Advanced Database Research Group based at the University of Arizona. She is a member of the IEEE and the IEEE Computer Society.



Jinsoo Park received the PhD degree in management information systems from the University of Arizona in 1999. He is an assistant professor of information systems in the College of Business Administration at Korea University. He was formerly on the faculty of the Department of Information and Decision Sciences in the Carlson School of Management at the University of Minnesota. His research interests are in the areas of semantic interoperability and metadata management in interorganizational information systems, heterogeneous database management and integration, knowledge sharing and coordination, data modeling, spatial database systems, geographic information systems, and intelligent agents for information resource management. His research has been published in *Computer*, *Information Systems Frontiers*, and several other journals and conference proceedings. Dr. Park currently serves on the editorial board of *Journal of Database Management (JDM)*. He is a member of IEEE, the IEEE Computer Society, ACM, AIS, and INFORMS.

► For more information on this or any computing topic, please visit our Digital Library at <http://computer.org/publications/dlib>.