On semi-automated matching and integration of database schemas

Ünal Karakaş, Ö.

Publication date
2010

Chapter 2

Interlinking and integrating schemas - background

Focusing on interlinking and/or integrating heterogeneous data from distributed nodes, database management research has introduced a number of approaches, architectures, and systems to enable their data sharing and data exchange, and in this process, it has also introduced a large variety of terms and concepts. This chapter addresses these approaches and definitions of these introduced terms and concepts that are closely related to schema matching and schema integration. Section 2.1 addresses these variety and it specifically represents our classification of the main concepts related to distributed information management, which are introduced in previous research. Section 2.2 depicts the main related categories of approaches from multidatabases research, based on schema coupling. Section 2.3 addresses the notions of schema integration and schema matching. Finally, Section 2.4 summarizes this chapter and emphasizes the importance of the automation of schema matching and schema integration processes.

The research results presented in this chapter were partially published in the Journal of Software (Unal & Afsarmanesh, 2009).

2.1 Related Concepts

High-speed networks have made it possible for the distributed information to be made available to everybody connected to the Internet. This has facilitated distributed information management systems, enabling access by authorized users to distributed data. The main requirements that need to be met with such systems have been summarized by (Kamel & Kamel, 1992) as follows: authorized users must be able to transparently access distributed and heterogeneous databases, there must be no changes needed in existing databases and applications, new databases should be easily added to the system, databases should be accessible for retrievals and updates, and finally performance of the system should be comparable to homogeneous systems.

Extensive research to enable data sharing in a distributed environment has given rise to variety of terms referring to different types of distributed information management systems. For instance, *distributed databases*, *multidatabases*, and *federated databases* are the most frequently used terms and concepts in the database research for several decades. However,

there are not yet commonly agreed definitions for these terms and concepts and quite often different researchers use the same term with different meanings. Therefore, the aim of this section is to provide enough background for this research, in order to differentiate among various definitions.

Distributed database (DDB) and distributed database management system (DDBMS) correspond to two base terms in distributed information management research. DDB and DDBMS are defined by (Ozsu & Valduriez, 1999) as follows:

> "A DDB is a collection of multiple, logically interrelated databases distributed over a computer network. A DDBMS is the software system that permits the management of the distributed database and makes the distribution transparent to the users."

In a typical distributed database management system, several databases over a network are managed by one management system. In other words, only one implementation of the database software is used in each network node.

According to (Ozsu & Valduriez, 1999), if the distributed database systems at various sites are also autonomous and possibly heterogeneous, they are referred to as *multidatabase* systems. Multidatabase systems allow integrated access to distributed, autonomous, and heterogeneous databases as (Bukhres & Elmagarmid, 1996) defines.

Multidatabase systems can be *homogeneous* or *heterogeneous*. Homogeneous database systems have the same database management systems and use the same data model and database manipulation language (Heimbigner & Mcleod, 1985) (Ozsu & Valduriez, 1999). Heterogeneous multidatabase systems on the other hand have different database management systems and use different data model and database manipulation language.

Another classification in *multidatabase* systems used by (Sheth & Larson, 1990) divides them into two types based on the autonomy of the participating database systems (components): *non-federated* and *federated*. Components in a *non-federated database system* are not autonomous. On the other hand, the components in a *federated database system* preserve their autonomy while also sharing their data in a partial and controlled manner. They share a part of their data by defining export schemas and making them available only to specific components. Every component is able to import schemas from other components according to the defined access permissions. As a consequence of this general interaction, this approach allows the cooperation between the nodes in the federation to accomplish a common or global task (Afsarmanesh et al., 2004).

(Sheth & Larson, 1990) categorizes federated databases further as *loosely coupled* and *tightly coupled*. A federated database system is loosely coupled if there is not a single authority to create and maintain the system; but this is the responsibility of users from each component system. There is no single global schema in loosely coupled systems. On the other hand, in tightly coupled systems, there is a central authority to administer the federation. If a tightly coupled federated database system only supports a single federated schema it is said to have a single federation, but if it supports multiple federated schemas it is said to have multiple federations. Users can submit queries applied to the federated schema, and the central authority is in charge of the distribution of sub-queries between the component databases and the processing of the individual results to satisfy the global request.

Besides terms referring to different types of distributed information management systems, another widely used term in database research domain is the *data integration.* Information systems mentioned above apply data integration techniques. *Data integration* aims at combining data residing at different sources and providing the user with a unified view of these data (Lenzerini, 2002). Data integration systems can be defined as a triple $\langle G, S, M \rangle$,

where G is the global schema, S is the heterogeneous set of source schemas, and M is the mapping between G and S. Two approaches are mentioned in the literature for defining M: *Global as View (GAV) (Chawathe et al., 1994)* and *Local as View (LAV) (Levy et al., 1996)*. In the GAV approach, there is a global schema expressed in terms of source schemas (S). Mappings M between the global schema G and source schemas S are well defined. However, when there is a new component database entering the system, a large amount of effort is required to update G. On the other hand, in the LAV approach, global schema is defined independently from source schemas and the relationships between the global schema and the sources are established by defining every source as a view over the global schema. Relationships between source schemas and the global schema may not be well defined here, which requires more complex query re-writing and thus puts more burdens on the query processor. Nevertheless, unlike GAV, addition of a new component database to the system does not require much effort.

Similar to variety of definitions related to distributed information management, there exist many definitions for database *interoperability*. For example, Brodie and Ceri (Brodie & Ceri, 1992) referred to interoperability as the ability of different systems to operate with each other. On the other hand, Silberschatz et al. (Silberschatz et al., 1990) defined interoperability as the problem of making heterogeneous and distributed databases behave as if they form part of a single database. Litwin and Abdellatif (Litwin & Abdellatif, 1986) and Zisman (Zisman, 1995) used the term interoperability to refer to the management and co-operation of multidatabase systems without using a global schema. Although there is no consensus on these definitions, database interoperability is a broader term than the terms related to distributed information management.

As it is clear from the definitions given above, there are many related terms concerning management of data provided by distributed and possibly heterogeneous and autonomous databases, whereas there is no consensus of terminology in the database community. In order to provide our understanding of the terms related to an integrated information management system, we have organized these definitions as shown in Figure 2.1.

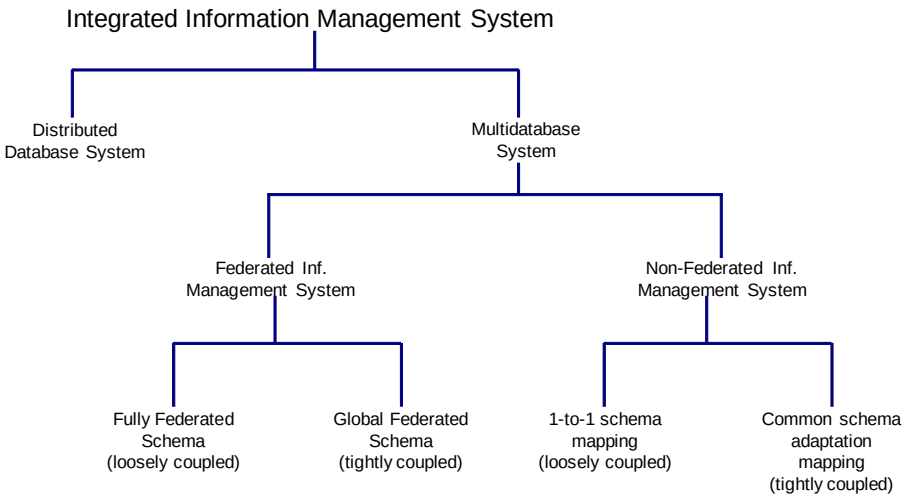

Fig. 2.1. Integrated Information Management System

Following the definition of (Ozsu & Valduriez, 1999), we mention two types of integrated information management systems: distributed database systems and multidatabase systems. Based on the classification of (Sheth & Larson, 1990), we divide the multidatabase systems as federated information management systems and non-federated information management systems.

Federated information management systems consist of autonomous nodes that can follow a fully federated schema or a global federated schema approach. As illustrated in Figure 2.2-a, a fully federated schema approach (Afsarmanesh et al., 1998) constructs an integrated schema at each node by merging the local schema of that node with the schemas imported from other nodes. Import schemas represent the information that other nodes make available to this node. A global federated schema approach on the other hand, generates a global schema by integrating the export schemas (representing the shared part of the information) from different nodes into a single schema, as shown in Figure 2.2-b.

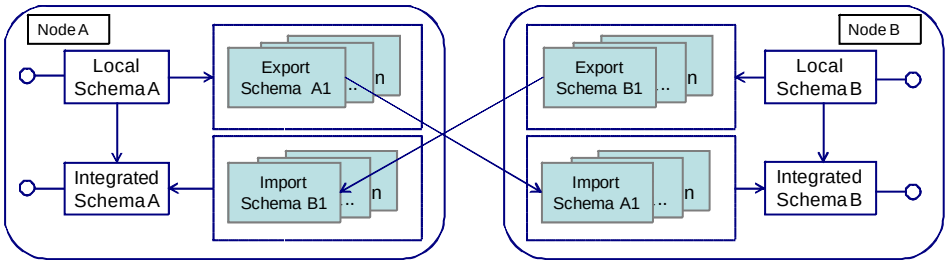

Fig. 2.2-a. Fully Federated Schema

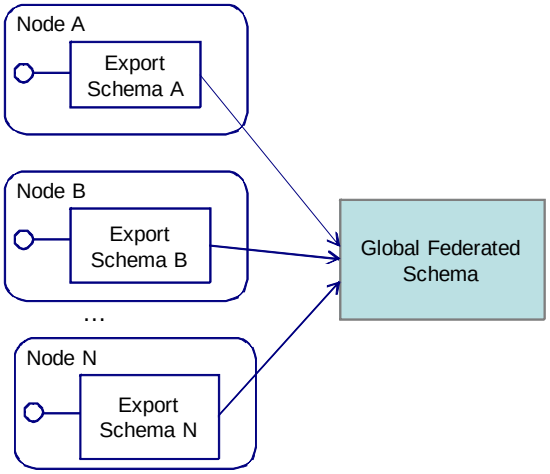

Fig. 2.2-b. Global Federated Schema

Nodes of non-federated information management systems are not autonomous. Two approaches can be mentioned here: 1-to-1 schema mapping and common schema adaptation mapping. In 1-to-1 schema mapping approach, mappings between the schemas of nodes are identified in a pair-wise manner. For instance, as represented in Figure 2.3-a, mappings between the schema of Node A and schemas of each other nodes are independently defined. Whereas in common schema adaptation mapping approach, mappings are specified between the common schema and the local schema of each node, as depicted in Figure 2.3-b.

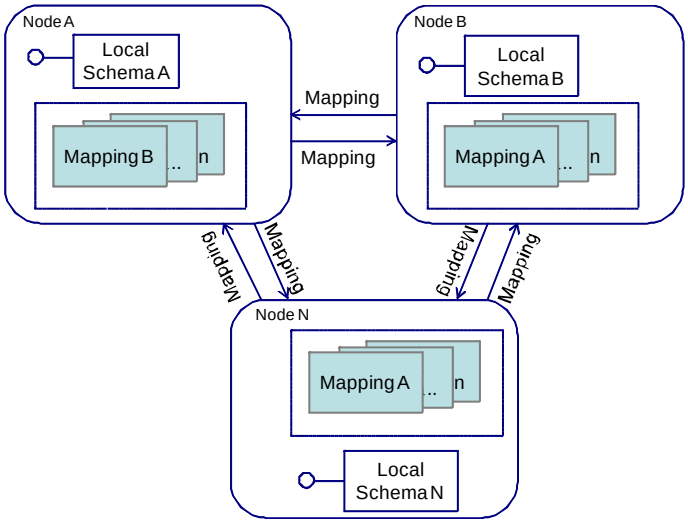

Fig. 2.3-a. 1-to-1 Schema Mapping

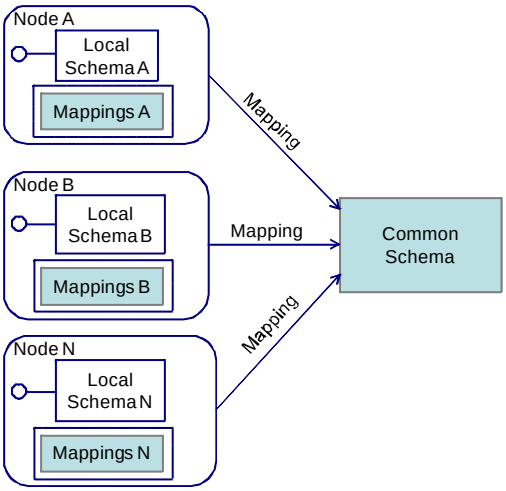

Fig. 2.3-b. Common Schema Adaptation Mapping

2.2 Multidatabase Classification Based on Schema Coupling

In this section, we focus on different types of multidatabase architectures based on schema coupling. We refer to multidatabase system as the one consisting of distributed and heterogeneous databases. By following definitions of (Zisman, 1995), we present a general overview of multidatabase architectures based on schema coupling in Figure 2.4.

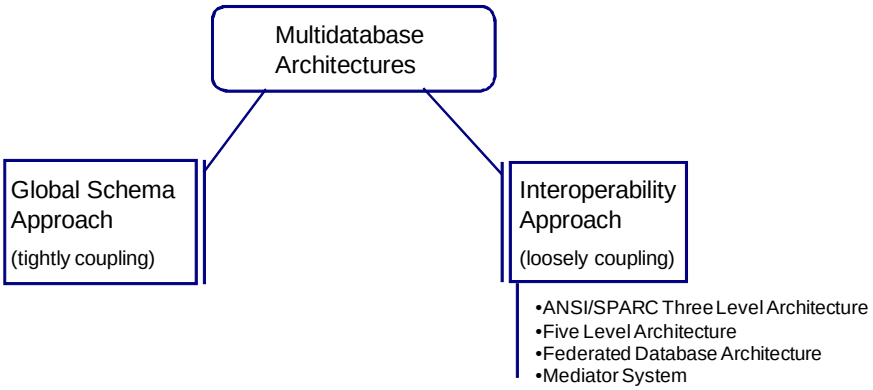

Fig. 2.4. A general overview of Schema Matching Approaches Based on Schema Coupling

In one of the multidatabase architectures, Global Schema Approach (also called as tightly coupling), there exists a single global schema representing all information across the databases. Global schema is generated by resolving the conflicts among local schemas and then integrating them into a single schema. A global schema is usually difficult to create as in order to create it one needs to fully understand the local database structures of participants. Generation of a global schema is achieved in several steps (Zisman, 1995). First, schemas are represented in a canonical data model, in case they are defined using different data models. Secondly, conflicts are resolved and the integrated schema is generated. In this approach, queries are created in terms of global schema and when such query arrives, it is decomposed into sub-queries to be sent to local databases. After this step, sub-queries are translated into the data language of the local database. When results of each query are received, they are merged for the final result to be sent to users. Global schema approach is suitable whenever the schemas are not subject to frequent changes. Advantages of this approach are; it is easy for querying and information loss is reduced. However, when the number of local schemas to be integrated is large or the environment is dynamic, this approach becomes complex.

In another multidatabase architecture, instead of creating a global schema, the aim is to make heterogeneous databases interoperable. In this architecture, either partial or no integration is required. Two types of interoperability approaches can be mentioned (Zisman, 1995):

1) *Direct Interoperability*, which consists of direct mappings (translations) among the components. Direct mapping is difficult when one schema is semantically more expressive than the other.

2) *Indirect Interoperability*, where an intermediate (canonical) data model and data manipulation language is used to manipulate heterogeneous databases. The canonical

data model is used to represent other models, bridge the gap between local models, detect inter-database semantic relationships, and achieve interoperability.

During the 80s, a variety of interoperability architectures were proposed in the literature, including ANSI/SPARC Three Level Architecture (Tsichritzis, 1981), Five Level Architecture (Sheth & Larson, 1990), federated database architecture (Hammer & Mcleod, 1979; Heimbigner & Mcleod, 1985), and mediator systems (Wiederhold, 1992).

Interoperability architecture overcomes the drawbacks of global integrated schema approach. This approach is appropriate when there are a large number of information sources and/or the environment is dynamic. However, the query processing costs are high in interoperability architectures.

An Example of Federated Database Architecture: PEER Federated Database System

The PEER system (Afsarmanesh et al., 1996; Tuijnman & Afsarmanesh, 1993) is a fully federated system designed and implemented at the University of Amsterdam. PEER is a generic object-oriented federated information management system enabling information sharing among autonomous and heterogeneous nodes. In the PEER architecture (see Figure 2.2-a), there is no need to create a global schema, as information stored in different nodes are interlinked through federated schemas. There are four types of schemas at each node: a local schema, a number of export schemas, a number of import schemas, and an integrated schema. The local schema models the local data at the node. Export schemas model the information that this node wants to share with other nodes of the network. Import schemas model the information that this node can access from other nodes. In other words, an import schema at each node is the export schema of another node that shares its data through this export schema. The integrated schema models the information that the node can access.

2.3 Schema Matching and Schema Integration

Organizations model their data using a variety of schema constructs. However, there is no single way to represent the same or similar data, which results in diversities in schema definitions even in the same organization. Distributed information management systems, introduced in the previous sections, need to tackle conflicts or heterogeneities and identify correspondences among schemas. As a result, schema matching and schema integration have become two main facilitating processes of distributed information management systems, which are mainly performed manually at present.

Schema specification is the main element of the schema matching and schema integration processes. A schema specifies how data is stored, accessed, and managed in the database management system (DBMS) and is described in a formal language supported by the DBMS. Examples of schema related languages include the SQL's DDL from relational data modeling domain, the Object Definition Language (ODL) from the object-oriented data modeling domain, the XML Metadata Interchange (XMI) of the Unified Modeling Language (UML), the XML Schema Definition (XSD) for XML documents, and the Resource Description Framework Schema (RDFS) as well as the OWL for ontologies.

The main inputs for the schema matching and schema integration processes are therefore the schemas. Following sub-sections make the role of schemas in these two processes clear. Also, more detailed information about schema matching and schema integration is given in these sub-sections.

2.3.1 Schema Integration

The problem of schema integration in the context of distributed information management systems is a relatively old challenge. In different approaches to enabling access to distributed and heterogeneous data, a different level of integration is achieved. Considering the classification of integrated information management systems, shown in Figure 2.1, schema integration is necessary in both fully-federated schema and global federated schema approaches. However, in the case of fully-federated schema approach, each node needs to integrate its local schema with the import schemas of other nodes to generate a representation of information that this node can access. On the other hand, in the case of global federated schema approach, schemas of all nodes, representing the information that these nodes make available to the network, need to be integrated to generate a single common schema that defines the information available at the network of nodes.

In database research, schema integration is typically used to refer to both the view integration and database integration (Batini et al., 1986). View integration aims at producing an integrated schema of users' views and is performed during the database design process, whereas database integration derives a new schema from existing specification. As identified in (Spaccapietra et al., 1992), view integration methodologies work with views based on the same data model, but database integration technologies work with schemas that are usually defined using heterogeneous data models. Considering the goals of the research work explained in this thesis, the focus is on the database integration. Therefore, when we use the phrase 'schema integration', we actually refer to integration of 'databases'. Furthermore, while we devise ways of semi-automatically integrating schemas, we target schemas which are based on the relational data model; i.e. both source and target schemas that we try to match and integrate are relational schemas.

There has been an extensive research work on the schema integration subject. A comprehensive survey of schema integration methodologies were done by (Batini et al., 1986). In (Batini et al., 1986), an analysis of twelve related methodologies were carried out, and they were compared based on different criteria, including the used data model, inputs, outputs, and strategies followed.

Considering all methodologies and approaches for schema integration and adding our own approach to it, three main integration steps can be identified, namely: 1) the Pre-integration step, 2) the Matching step, and 3) the Integration step.

1. The *Pre-integration* step consists of a number of preparation steps before the integration, such as identifying schemas to be integrated, preferences to be considered in the integration process, and amount of user input, as (Batini et al., 1986) mentioned. The type of the integration strategy followed affects the identification of schemas to be integrated. Two types of strategies are mentioned in (Batini et al., 1986) for schema integration: binary and n-ary strategies. Binary strategies allow the integration of two schemas at a time, while n-ary strategies can integrate *n* schemas at a time. Because of the complexities of integrating *n* schemas at a time, most approaches in the literature prefer a binary strategy.

2. The *Matching* step, also called the Investigation step by (Spaccapietra et al., 1992), identifies correspondences among different schemas by resolving their conflicts. Instead of the Matching step, the (Batini et al., 1986) categorizes two other steps called: comparison of the schemas and conforming the schemas, which together constitute the Matching step.

3. The *Integration* step is responsible for integrating the schemas, based on the correspondences identified in the matching step.

In this direction, the previously suggested schema integration approaches and methodologies are either fully manual or with some limited degree of automation focused only on the third step, and not including the matching step. Furthermore, for any automation on the integration step, it is typically assumed that the full semantics and structural knowledge of the two schemas are available.

Schema integration is a challenging and complex task: The integration step cannot be fully automated, since automatic resolution of some types of conflicts is not possible and user input is required to determine the appropriate meanings and decide on mappings for the integrated schema. Nevertheless, carrying out this process as automatically as possible and helping the users with this complicated task are needed in order to cope with the increasing demand for integrated information management systems.

The research work explained in this thesis addresses the full cycle of semi-automatic schema integration in three main steps, including: Configuration, Schema Matching, and Schema Integration, as shown in Figure 2.5. Some limited user input is required at these steps, as addressed below. The configuration step is responsible for assigning desired weights to the algorithms used in the linguistic and structure matching components, as well as for identifying the desired selection strategy for the results of schema matching. The schema matching step starts with a preparation activity that automatically turns the two source schemas (donor and recipient schemas) into a common format. Then, this process takes the schemas represented in the common format, as well as some other required inputs, as described in Section 2.3.2, and identifies all possible matches between the two schemas. After receiving the user input on the match results, at the third step, the schema integration takes as input the accepted correspondences between the two schemas and using a number of predefined integration rules, it automatically generates both an integrated schema as well as the needed mappings between the integrated schema and the two source schemas being integrated. The mappings are expressed in terms of a derivation language introduced in Chapter 4. Finally, the user input is required for the final validation of the integration results.

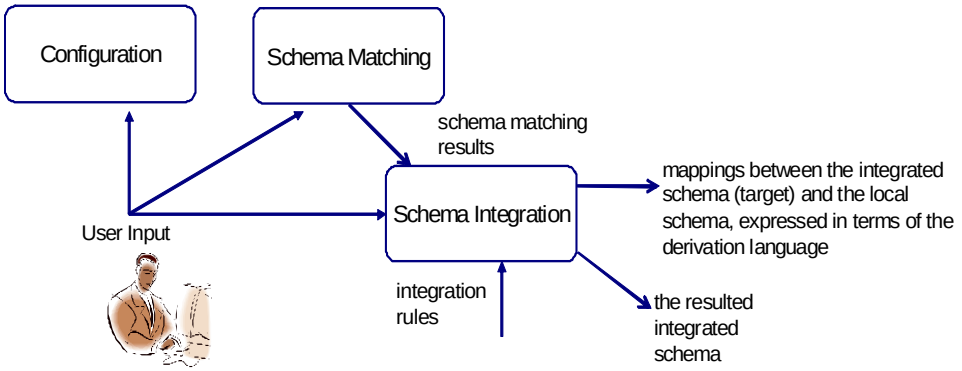

Fig. 2.5. Main Steps of the Schema Integration Process

2.3.2 Schema Matching

To achieve any of the integrated information management systems introduced in Section 2.1, there is a need to compare their schemas (e.g. two schemas at a time) and identify correspondences between them. As addressed in Section 2.3.1 on schema integration, federated information management approaches generate integrated schemas, where schema matching is one main step for the schema integration. In non-federated information systems however, the aim is to generate mappings either between the global schema for the network of databases and each of the local schemas - thus resulting a set of schema adaptation mappings, or between each pair of the local schemas - thus resulting a set of 1-to-1 schema mappings. Identifying the correspondences and generating the needed mappings also require support from the schema matching process.

Schema matching can be defined as the process for finding the correspondences between different elements of two schemas. The simplest type of matching is the 1-to-1 matching. For two schemas, e.g. A and B, schema matching process can identify for each element of schema A, the most similar element of schema B. In addition to 1-to-1 matches, some complex matches also frequently occur among schema elements. Complex matching identifies correspondences between each element or a group of elements of the schema A and a group of elements of schema B. Groups of elements are combined and inter-related with a formula. For example, suppose that there is a match identified between the 'name' element of Schema A and the 'fname' and 'lname' elements of Schema B. In this case, 'fname' and 'lname' can be combined through concatenation and a mapping can be defined between 'name' and this combination of 'fname' and 'lname'. Most schema matching approaches focus only on the 1-to-1 matches, considering that it is much easier to identify 1-to-1 matches than complex ones.

As shown in Figure 2.6, which represents a simplified version of the classification provided in (Rahm & Bernstein, 2001), individual and combined matchers are two top-level classes of matchers in this classification of schema matching approaches. Combined matchers represent a set of individual matchers. Individual matchers are further divided into two: instance-based and schema-based matchers. Instance-based matchers exploit the instance information; schema-based matchers on the other hand consider the definition of schema itself. Furthermore, schema-based matchers can be applied to individual schema elements (at element level) or for combination of elements (at structure level). Element level matchers use the linguistic characteristics of the element names. They apply techniques such as tokenization and word separation, removal of stops words and hyphens, expansion of abbreviations, and lemmatization, the details of which are all given in Chapter 4. Element level matchers consider both the syntactic as well as the semantic features of names in the schema. Furthermore, these types of matchers benefit also from the constraint-based techniques**,** which deal with the internal constraints being applied to the definitions of entities, such as types, cardinality of attributes, and keys.

On the other hand, the structure level matchers exploit the graph-based techniques. Graph matching techniques and the relationships among the graph elements together form the base of structure level matchers.

In (Shvaiko & Euzenat, 2005), a different classification is introduced, where three main dimensions are mentioned for the classification of schema matching algorithms, including:

1. *Input dimension*: This dimension is related to both the kinds of data or conceptual model to express schemas that the matching algorithms shall use, such as the relational or object-oriented, as well as the kinds of elements that algorithms shall exploit, such as the schema level and/or instance level data.

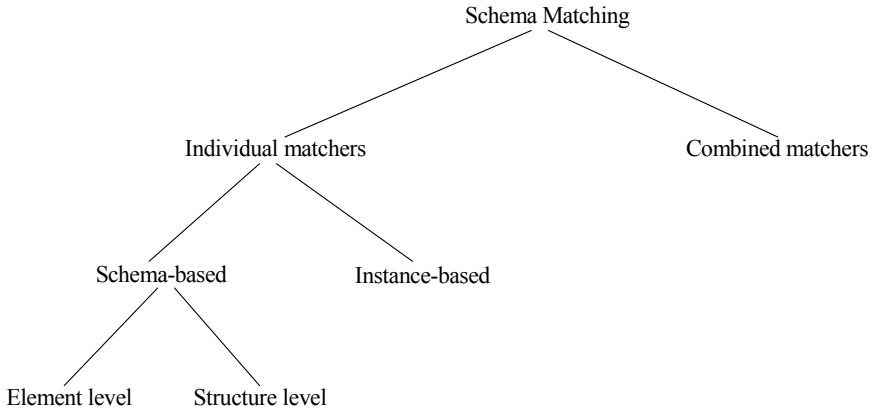

Fig. 2.6. A general overview of Schema Matching Approaches (simplified version of (Rahm & Bernstein, 2001))

2. *Process dimension*: This dimension considers the nature of the computation in the matching algorithms, which can be exact or approximate. For the exact algorithms, the completeness of the solution is considered, whereas for the approximate algorithms the performance aspects are preferred over exactness.
3. *Output dimension*: This dimension considers different possible forms of outputs generated by matching algorithms. For example, one algorithm can determine only 1-to-1 matches, while another one can also identify 1-to-many matches. Another example is that the results of some types of algorithms are values in the range [0,1] for element pairs being compared, whereas some other types of algorithms identify the match results using some relationship operations, such as 'equivalent'.

Schema matching process may take a variety of inputs and may produce some outputs depending on the matching approach that it applies. Figure 2.7 shows briefly the inputs and outputs for the matching process introduced in this research work, as later explained in this thesis. The variety of inputs consist of the schema specification, a linguistic dictionary, a number of linguistic and structural similarity algorithms, and the user input for validating the results. Output of the matching process is a set of similarity scores for each match identified for schema elements as well as the relationship operations for complex matches, such as string concatenation.

Extensive research has been done in the past in relation to the schema matching field. A number of approaches have been proposed, requiring different amounts of manual intervention from user. More detailed information about these schema matching approaches is given in Chapter 4.

A number of other terms and concepts related to schema matching process have been introduced in the research literature, such as: the ontology matching and mapping discovery. Especially, the ontology matching has drawn considerable attention in recent years with the increasing popularity of the Semantic Web. Although ontology and database schemas have different purposes, the spectrum of "ontology specification" is very broad and a database schema can be considered as a simple descriptive ontology of an environment. In general, "Ontology" is assumed with different meanings and details depending on where it is used. For

example, (Gruber, 1993) defines ontology as a specification of a conceptualization. In this direction, it can be related to a database schema, which in general presents the meta-data defined for a database containing information about the structure and content of that database.

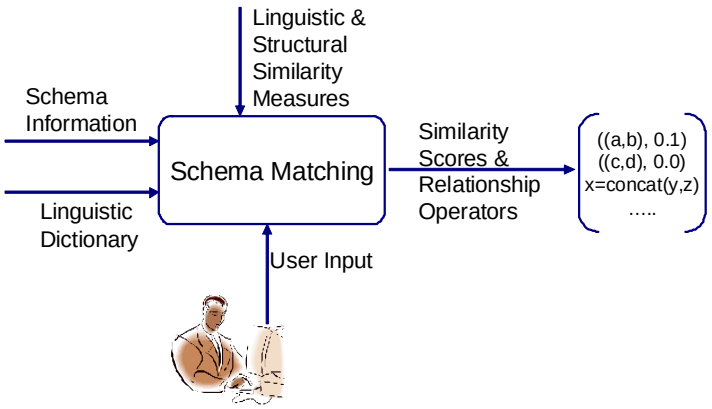

Fig. 2.7. Inputs and Outputs of Schema Matching

Applications of Schema Matching

In addition to its role in the semi-automatic schema integration, the matching process plays an important role in several other application domains, such as in data warehouses, query processing, Semantic Web, and e-business (Rahm & Bernstein, 2001) (Do et al., 2002). Each of these applications needs to deal with some heterogeneous schemas and identify the matches and mappings either manually or semi-automatically. The following paragraphs briefly address these application domains and their relation to matching process.

Data Warehouses

The number of data warehousing applications has increased rapidly in the last decade. Data warehousing has become popular with the need for analyzing large amount of data using different techniques and algorithms to extract information related to a variety of domains, such as sales. Data warehouses aim to most optimally support the analysis and reporting of collected data. In order to form a data warehouse, data from different sources need to be transformed into a common warehouse format. Schema matching process can help in creating an appropriate interlinking and transformation.

E-business and E-commerce

Another application area for schema matching is related to the heavy use of the e-commerce and e-business for transactions among companies. In recent years, these have become popular among both national and international trading partners, using the opportunities that the Internet provides. Using these technologies, partners exchange messages, receive product information,

place orders, sign contracts, etc. Each organization may use different tools and thus different formats for exchanging messages and conducting their transactions, such as the XML, the Electronic Data Interchange (EDI), etc. In order to exchange messages, they need to be translated from one format into another, for which the schema matching process can be applied.

Query Processing on the Web

With the increasing number of data sources available, query processing on the Web has become important. Users pose queries applying their own terminology and the query processing systems need to re-write these queries. For this purpose, the query processing system needs to identify correspondences and mappings between the terms in these queries and the actual terms introduced in the underlying schemas. This is therefore another potential application where schema matching can be utilized.

Semantic Web

The Semantic Web mechanisms contribute to semantically enriching the contents of the Web pages. Semantic enrichment is mainly achieved by associating the concepts on Web pages to ontologies. In other words, the contents of the Web pages are annotated by definitions within these ontologies. However, it is not necessary (and not practiced) that all Web pages use the same ontology. Therefore, before integrating information from different sites, Semantic Web needs to first identify correspondences among different related ontologies that these sites use to annotate their concepts. This is therefore another example of the need for schema matching process, where a semi-automatic schema matching approach can play an important role in matching different ontology elements for Semantic Web applications.

2.4 Conclusion

There are different definitions introduced in the literature for the terms and concepts related to data sharing among distributed nodes. For instance, the concept of ‘Federated Database Architecture’ is interpreted differently by different researchers and authors. This chapter provides some background on the concepts and definitions used by the past research, as related to the subject of this thesis.

Furthermore, for the purposes of setting the context for the problem space addressed in the thesis, and achieving common understanding of the terminology pertaining to the problem area that we try to tackle, this chapter provides our approach on integrated information management system taxonomy. Schema integration and schema matching are two important processes required by the integrated information management systems. As explained in this chapter, these processes need to be automated to the extent possible in order to facilitate easy construction of such information management systems.