ADAPTATIVE MATCHING OF DATABASE WEB SERVICES EXPORT SCHEMAS

Daniela F. Brauner, Alexandre Gazola, Marco A. Casanova and Karin K. Breitman
Departamento de Informática – PUC-Rio
Rua Marquês de São Vicente, 225 – 22.453-900 – Rio de Janeiro, RJ, Brazil

Keywords: Schema matching, mediator, database.

Abstract: This paper proposes an approach and a mediator architecture for adaptively matching export schemas of database Web services. Differently from traditional mediator approaches, the mediated schema is constructed from the mappings adaptively elicited from user query responses. That is, query results are post-processed to identify reliable mappings and to build the mediated schema on the fly. The approach is illustrated with two case studies from rather different application domains.

1 INTRODUCTION

A *database Web service* consists of a Web service interface with operations that provide access to a backend database. When a client sends a query to a database Web service, the backend engine submits the query to the backend database, collects the results, and delivers them to the client. The *export schema* describes the subset of the backend database schema that the database Web service makes visible to the clients (Sheth & Larson, 1990). Typically, a Web service announces its interfaces, including the export schema, using the Web Service Definition language – WSDL, a W3C standard.

A *mediator* is a software component that facilitates access to a set of data sources (Wiederhold, 1992). A mediator offers a *mediated* or *global schema* that represents an integrated view of the export schemas of the data sources.

In this paper, we focus on the design of a mediator that constructs the mediated schema adaptively from evidences elicited from user query responses. It precludes the a priori definition of the mediated schema and of the mappings between the export schemas and the mediated schema. The mediator assumes that the data sources are encapsulated by Web services, that is, they are database Web services. This assumption avoids the burden of interpreting HTML pages containing query results. Indeed, the mediator communicates with the database Web services by exchanging SOAP messages conforming to their WSDL descriptions.

The schema matching approach proposed in this paper is based on the following assumptions:

1. The database Web services accept *keyword queries*, that is, lists of terms as in a Web search engine.
2. The mediator accepts only keyword queries.
3. The database Web services return query results as flat XML documents. That is, each export schema consists of a single flat relational table (encoded as an XML Schema data type in the WSDL document of the service).
4. Attributes with similar domains are semantically equivalent.

This last assumption is rather strong, but the case studies described in the paper indicate that it is warranted in a number of application domains.

The remainder of this paper is organized as follows. Section 2 summarizes related works. Section 3 describes the instance-based schema matching approach. Section 4 presents the mediator architecture. Section 5 contains two case studies that illustrate the approach. Finally, section 6 contains the conclusions and directions for future work.

2 RELATED WORK

Schema matching is a fundamental issue in many database application domains, such as query mediation and Web-oriented data integration (Casanova et al., 2007; Rahm & Bernstein, 2001). By query mediation, we mean the problem of designing a *mediator*, a software service that is able to translate

F. Brauner D., Gazola A., A. Casanova M. and K. Breitman K. (2008).
ADAPTATIVE MATCHING OF DATABASE WEB SERVICES EXPORT SCHEMAS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - DISI*, pages 49-56
DOI: 10.5220/0001706900490056

user queries, formulated in terms of a mediated schema, into queries that can be handled by local databases. The mediator must therefore match each export schema with the mediated schema. The problem of query mediation becomes a challenge in the context of the Web, where the number of local databases may be enormous and, moreover, the mediator does not have much control over the local databases, which may join or leave the mediated environment at will.

In general, the *match* operation takes two schemas as input and produces a mapping between elements of the two schemas that correspond to each other. Many techniques for schema and ontology matching have been proposed to automate the match operation. For a survey of several schema matching approaches, we refer the reader to (Rahm & Bernstein, 2001).

Schema matching approaches may be classified as *syntactic vs. semantic* and, orthogonally, as *a priori* vs. *a posteriori* (Casanova et al., 2007). The *syntactic approach* consists of matching two schemas based on syntactical hints, such as attribute data types and naming similarities. The *semantic approach* uses semantic clues to generate hypotheses about schema matching. It generally tries to detect how the real world objects are represented in different databases and leverages on the information obtained to match the schemas. Both the syntactic and the semantic approaches work *a posteriori*, in the sense that they start with pre-existing databases and try to match their schemas. The *a priori approach* emphasizes that, whenever specifying databases that will interact with each other, the designer should start by selecting an appropriate standard (a common schema), if one exists, to guide the design of the export schemas.

An implementation of a mediator for heterogeneous gazetteers is presented in (Gazola et al., 2007). Gazetteers are catalogues of geographic objects, typically classified using terms taken from a thesaurus. Mediated access to several gazetteers requires the use of a technique to deal with the heterogeneity of different thesauri. The mediator incorporates an instance-based technique to align thesauri that uses the results of user queries as evidences (Brauner et al., 2006).

A semantic approach for matching export schemas of geographical database Web services is described in (Brauner et al., 2007). The approach is based on the use of a small set of global instances and on an ISO-compliant predefined global schema.

An instance-based schema matching technique, based on domain-specific query probing, applied to Web databases, is proposed in (Wang et al., 2004). A Web database is a backend database available on the Web and accessible through a Web site query interface. In particular, the interface exports query results as HTML pages. In particular, a Web database has two different schemas, the *interface schema* (IS) and the *result schema* (RS). The interface schema of an individual Web database consists of data attributes over which users can query, while the result schema consists of data attributes that describe the query results that users receive.

The instance-based schema matching technique described in (Wang et al., 2004) is based on three observations about Web databases:

1. Improper queries often cause search failure, that is, return no results. For the authors, improperness means that the query keywords submitted to a particular interface schema element are not applicable values of the database attribute to which the element is associated. For instance, if you submit a string to query search element that is originally defined as an integer, you get an error. As an example, consider submitting a *title* value to the search element *pages number*.
2. The keywords of proper queries that return results very likely reappear in the returned result pages.
3. There is a global schema (GS) for Web databases of the same domain (He & Chang, 2003). The global schema consists of the representative attributes of the data objects in a specific domain.

The query probing technique consists of exhaustively sending keyword queries to the query interface of different Web databases, and collecting their results for further analysis. Based on the third observation, they assume, for a specific domain, the existence of a pre-defined global schema and a number of sample data objects under the global schema, called *global instances*. For Web databases, they deal with two kinds of schema matching: *intra-site schema matching* (that is, matching global with interface schemas, global with result schemas, and interface with result schemas) and *inter-site schema matching* (that is, matching two interface schemas or two result schemas).

The data analysis is based on the second observation. Given a proper query, the results will probably contain the reoccurrence of the submitted value (referring to the values of the attributes of the global instances). The results will be collected in the HTML sent to the Web browser. Thus, the reoccurrence of the query keywords in the returned results can be used as an indicator of which query submission is appropriate (i.e., to discover associated elements in the interface schema). In addition, the position of the submitted query keywords in the

result pages can be used to identify the associated attributes in the result schema.

Note that, differently from (Wang et al., 2004), we work with Web services that encapsulate databases. In particular, we assume that the service interface is specified by a WSDL document that describes the input attributes (interface schema) and the output attributes (export schema). This means that both the query definitions and query answers are encoded as SOAP messages. Therefore, we avoid the interpretation of query results encoded in HTML, which introduces a complication that distracts from the central problem of mediating access to databases.

Also, differently from (Brauner et al., 2007) and (Wang et al., 2004), we avoid the use of a global schema and a set of global instances. Defining a global schema and collecting a good set of global instances are hard tasks. The technique presented here instead uses an instance-based approach to align the export schemas using the results of user queries as evidences for the mappings, as in (Brauner et al., 2006).

3 THE SCHEMA MATCHING APPROACH

The schema matching approach proposed in this paper is based on the matching process illustrated in Figure 1. The matching process is the basis of a mediator that facilitates access to a collection of database Web services, assumed to cover the same application domain. In section 4, we discuss the complete mediator architecture.

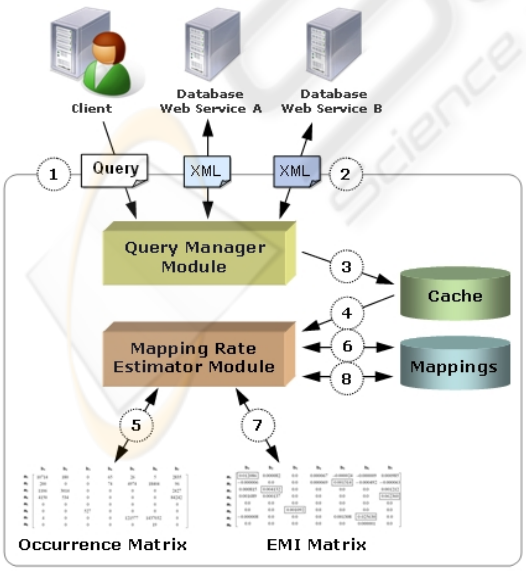

Figure 1: The matching process.

The matching process starts with a client query submitted to the mediator and the WSDL descriptions of the database Web services accessed through the mediator engine (step 1 on Figure 1). Note that, through the WSDL documents, the mediator obtains the necessary information to encode queries to be sent to the database Web services, as well to interpret the results sent back.

Our current schema matching approach works under the following assumptions:

1. The database Web services accept *keyword queries*, that is, lists of terms as in a Web search engine.
2. The mediator accepts only *keyword queries*.
3. The database Web services return query results as flat XML documents. That is, each export schema consists of a single flat relational table (encoded as an XML Schema data type in the WSDL document of the service).
4. Attributes with similar domains are semantically equivalent.

This last assumption is rather strong, but the case studies described in the paper indicate that it is warranted in a number of application domains.

When a client submits a query, the mediator forwards it to the database Web services registered through the Query Manager Module (QMM) (step 2). The result sets are delivered to the user, and simultaneously, they are cached in a local cache database (step 3). Then, the mediator analyzes the cached instances by counting instance values that reoccurs in both result sets (step 4). This task is performed by the Mapping Rate Estimator Module (MREM).

The MREM analyzes the result sets by a *probing technique*. The probing technique consists of counting the reoccurrence of instances values from one set in the other.

After this analysis, the MREM generates the occurrence matrix (step 5). To enable the mediator to accumulate evidences, the MREM stores the occurrence matrix in the Mappings local database (step 6). If there is a previously stored occurrence matrix, the MREM must sum the existing matrix with the new matrix, generating an accumulated occurrence matrix. If new attributes are matched, the MREM must add rows and columns to the accumulated occurrence matrix. Finally, given an accumulated occurrence matrix, the MREM also generates the Estimated Mutual Information (EMI) matrix (step 7). The generation of the EMI is explained in detail at the end of this section.

For instance, suppose that the mediator provides access to databases D_A and D_B with export schemas S_A and S_B, respectively. Suppose that the mediator receives a keyword query Q from the client application, which it resends to the database Web

services. Let R_A and R_B be the results received from D_A and D_B, respectively. These results are analyzed to detect if there are XML elements a_n in R_A and b_m in R_B that have the same value v. If this is indeed the case, then v establishes some evidence that a_n and b_m map to each other. This analysis generates an occurrence matrix M between attributes from the export schemas of both databases.

As in (Wang et al., 2004), we assume that the attributes of S_A and S_B induce a partition of the result sets R_A and R_B returned by the database Web services. Suppose the attributes of S_A partition R_A into sets $A_1,A_2,\ldots A_m$ and the attributes of S_B partition R_B into sets $B_1,B_2,\ldots B_n$. The element M_{ij} in the occurrence matrix M actually indicates the content overlap between partitions A_i and B_j with respect to the reoccurrences of values in the two partitions. The schema matching problem now becomes that of finding pairs of partitions from different schemas with the best match. In what follows, we formalize what we mean by best match.

To address this point, we introduce the concept of mutual information (Wang et al., 2004), which interprets the overlap between two partitions X and Y of a random event set as the "information about X contained in Y" or the "information about Y contained in X" (Papoulis, 1984). In other words, the concept of mutual information aims at detecting the attributes that have similar domains, i.e., similar domain value sets.

Given an occurrence matrix M of two export schemas S_A and S_B, the estimated mutual information (EMI) between the r^{th} attribute of S_A (say a_r) and the s^{th} attribute of S_B (say b_s) is:

$$\mathrm{EMI}(a_r, b_s) = \frac{m_{rs}}{\sum_{i,j} m_{ij}} \log \frac{\frac{m_{rs}}{\sum_{i,j} m_{ij}}}{\frac{\sum_{j} m_{rj}}{\sum_{i,j} m_{ij}} * \frac{\sum_{i} m_{is}}{\sum_{i,j} m_{ij}}} \quad (1)$$

Note that, if m_{rs} is equal to 0, EMI is assumed to be 0 as well.

So, given an EMI matrix, the MREM derives the mappings between the export schemas elements (step 8 in Figure 1). Given an EMI matrix $[e_{ij}]$, the r^{th} attribute of S_A matches with the s^{th} attribute of S_B iff $e_{rs} \geq e_{rj}$, for all $j \in [1,n]$ with $j \neq s$, and $e_{rs} \geq e_{is}$, for all $i \in [1,m]$ with $i \neq r$. Note that, by this definition, there might be no best match for an attribute of S_A.

4 THE MEDIATOR ARCHITECTURE

Figure 2 illustrates the architecture for a mediator implementing the schema matching approach.

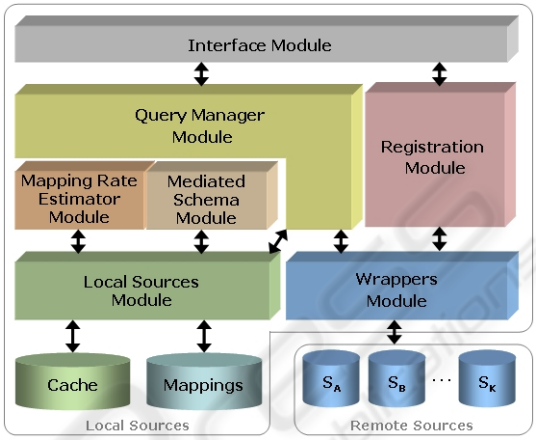

Figure 2: The mediator architecture.

The User Interface Module (UIM) is responsible for the communication between the user (clients) and the mediator. The UIM accepts two kinds of user interactions: query and registration. Through the UIM, a user could register a new data source providing the WSDL description of the database Web service. The Registration Module (RM) is responsible for accessing the WSDL and registering the new service, and creating a new wrapper to access the remote source. The UIM also accepts keyword queries. Every time the UIM receives a new query, it forwards it to the Query Manager Module (QMM). The QMM is responsible for submitting user queries to database Web services. The QMM communicates with the Local Sources Module (LSM) and Wrappers Module (WM) to access local and remote sources respectively. Moreover, the QMM communicates with the LSM to store query responses in the local cache database. The Mapping Rate Estimator Module (MREM) is an autonomous module that is responsible for accessing the local cache database to compute the occurrence matrix and the estimated mutual information matrix between export schemas and generate the mappings. The MREM communicates with the LSM to store mappings into the local mappings database. The Mediated Schema Module (MSM) is responsible for the creation of the mediated schema. To achieve this, the MSM uses the mappings discovered during the matching schema process. The stored mappings are retrieved from the mappings local database and used to induce the elements of the mediated schema (see Section 5 examples).

5 CASE STUDIES

5.1 Bookstore Databases Experiment

The first experiment we describe uses two bookstore databases, Amazon.com (D_A) and Barnes and Noble (D_B).

D_A is available as a database Web service. Figure 3 shows a fragment of the Amazon.com WSDL. We selected the operation *ItemSearch*. Referring to Figure 3, this operation accepts an *ItemSearchRequestMsg* message as input and returns an *ItemSearchResponseMsg* message as output. We suppressed some elements from Figure 3 to preserve readability. Table 1 reproduces the parameter list returned by the *ItemSearch* operation, representing the export schema S_A of D_A.

The second data source, Barnes and Noble (D_B), does not provide a Web service interface, which forced us to create one to run the experiment. Table 2 shows D_B export schema S_B.

For the keyword query "Age", D_A returned the result set R_A, with 23,338 entries, and D_B returned the result set R_A, with 6,168 entries. With both result sets in cache, the mediator applied the technique described in Section 3, generating the occurrence matrix between attributes of D_A and D_B. Figure 4 shows the occurrence matrix and Figure 5 the estimated mutual information matrix.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:tns="http://webservices.amazon.com/AWSECommerceService/2007-10-29"
    targetNamespace="http://webservices.amazon.com/AWSECommerceService/2007-10-29">
  - <types>
    - <xs:schema
        targetNamespace="http://webservices.amazon.com/AWSECommerceService/2007-10-29"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:tns="http://webservices.amazon.com/AWSECommerceService/2007-10-29" elementFormDefault="qualified">
      + <xs:element name="ItemSearch">
      + <xs:complexType name="ItemSearchRequest">
      + <xs:element name="ItemSearchResponse">
        ...
        </xs:schema>
    </types>
  - <message name="ItemSearchRequestMsg">
      <part name="body" element="tns:ItemSearch" />
    </message>
  - <message name="ItemSearchResponseMsg">
      <part name="body" element="tns:ItemSearchResponse" />
    </message>
    ...
  - <portType name="AWSECommerceServicePortType">
    + <operation name="Help">
    - <operation name="ItemSearch">
        <input message="tns:ItemSearchRequestMsg" />
        <output message="tns:ItemSearchResponseMsg" />
      </operation>
      ...
    </portType>
  + <binding name="AWSECommerceServiceBinding"
    type="tns:AWSECommerceServicePortType">
  - <service name="AWSECommerceService">
      - <port name="AWSECommerceServicePort"
          binding="tns:AWSECommerceServiceBinding">
          <soap:address location="http://soap.amazon.com/onca/soap?Service=AWSECommerceService" />
        </port>
    </service>
</definitions>
```

Figure 3: Amazon.com WSDL fragment.

Table 1: Amazon.com (D_A) Export Schema (S_A).

Attribute name	*Description*	*Data type*
title (a_1)	title	String
edition (a_2)	edition	String
author (a_3)	author name	String
publisher (a_4)	publisher	String
isbn (a_5)	International Standard Book Number (10 digit)	String
ean (a_6)	European Article Number - a barcoding standard – book id	String
pages (a_7)	number of pages	Number
date (a_8)	publication date	String

Table 2: Barnes and Noble (D_B) Export Schema (S_B).

Attribute name	*Description*	*Data Type*
name (b_1)	title	String
by (b_2)	author name	String
isbn (b_3)	International Standard Book Number – book id	String
pub_date (b_4)	publication date	String
sales_rank (b_5)	number of times that other titles sold more than this book title	Number
number_of_ pages (b_6)	number of pages	Number
publ (b_7)	publisher	String

	b_1	b_2	b_3	b_4	b_5	b_6	b_7
a_1	10714	180	0	65	26	5	2835
a_2	200	0	0	74	4974	18404	96
a_3	1106	3014	0	0	0	0	2427
a_4	4150	534	0	0	0	0	84242
a_5	0	0	0	0	0	0	0
a_6	0	0	527	0	0	0	0
a_7	4	0	0	0	121577	1437032	0
a_8	0	0	0	0	0	19	0

Figure 4: Occurrence matrix between S_A and S_B.

	b_1	b_2	b_3	b_4	b_5	b_6	b_7
a_1	0.012086	0.000082	0.0	0.000067	−0.000024	−0.000009	0.000985
a_2	−0.000006	0.0	0.0	0.000069	0.001314	−0.000492	−0.000063
a_3	0.000815	0.004132	0.0	0.0	0.0	0.0	0.001212
a_4	0.001689	0.000137	0.0	0.0	0.0	0.0	0.062360
a_5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
a_6	0.0	0.0	0.001092	0.0	0.0	0.0	0.0
a_7	−0.000008	0.0	0.0	0.0	0.001308	0.025630	0.0
a_8	0.0	0.0	0.0	0.0	0.0	0.000001	0.0

Figure 5: Estimated Mutual Information matrix between S_A and S_B.

In this first experiment, we used simple comparison operators to identify reoccurred values. For textual attributes, we used the SQL statement "LIKE" and for numerical attributes, the "=" operator.

Table 3 shows the alignments found between the attributes of S_A and S_B. Note that the only wrong alignment was between *edition* from S_A and *sales_rank* from S_B. These attributes are not semantically similar. However, a false matching was found due to the reoccurrence of 4974 values from D_A in D_B. For instance, in the Amazon.com database, several first book editions had the *edition* value equal to "1" and, in the Barnes and Noble database, several books had *sales_rank* also equal to "1".

An interesting observation can be made regarding book identifiers. Starting in 2007, the 13-digit ISBN began to replace the 10-digit ISBN. D_A stores both numbers, with the attribute ISBN holding the old 10-digit ISBN and the attribute EAN, the new 13-digit ISBN. D_B stores only the new 13-digit ISBN (the attribute ISBN). Differently from a syntactical approach, which would wrongly match attribute ISBN from S_A with attribute ISBN from S_B, our instance-based technique correctly matched attribute EAN from S_A with attribute ISBN from S_B.

The date attributes would never reoccur due to format differences. For instance, Amazon.com database stores dates in the format "YYYY-MM-DD", while the Barnes and Noble database stores the publication date as "Month, YEAR". To solve this problem, the algorithm would have to be enhanced with a type-based filter. In our experiments, attribute *date* was modelled as a string in both sources.

Assume that *attributes with similar domains are semantically equivalent*. Based on this assumption, by analyzing the values of the estimated mutual information matrix, the mediator has some evidence of which attributes should be part of a mediated schema. For instance, by observing Table 3, we note that *title* from the Amazon.com database aligns with *name* from the Barnes and Noble database, and so on. Table 4 shows the complete mediated schema, where attribute names were chosen from the first export schema, in our case, that of Amazon.com.

5.2 Gazetteers Experiment

Our second experiment used two geographical gazetteers available as database Web services, the Alexandria Digital Library (ADL) Gazetteer (D_C), and Geonames (D_D). In our experiments, we accessed both gazetteers through their *search-by-place-name* operations.

The ADL Gazetteer contains worldwide geographic place names. The ADL Gazetteer can be accessed through XML- and HTTP-based requests (Janée & Hill, 2004). Table 5 contains the ADL export schema (S_C) and Figure 6 shows a fragment of the XML response of this service.

Table 3: The aligned attributes (S_A - S_B).

S_A *attributes*	S_B *attributes*
title (a_1)	name (b_1)
edition (a_2)	*sales_rank* (b_5)
author (a_3)	by (b_2)
publisher (a_4)	publ (b_7)
ean (a_6)	Isbn (b_3)
pages (a_7)	number_of_pages (b_6)

Table 4: The mediated schema (S_A - S_B).

Attribute name	*Description*	*Data type*
title	title	String
author	author name	String
publisher	The publisher of the book	String
ean	book identifier	String
pages	number of pages	Number
date	publication date	String

Table 5: ADL Gazetteer (D_C) Database Web Service Export Schema (S_C).

Attribute name	*Description*	*Data type*
identifier (c_1)	entry local id	String
gnis_identifier (c_2)	entry id on GNIS	String
placeStatus (c_3)	entry place-status (current or former)	String
displayName (c_4)	display name	String
names (c_5)	alternative names	names
bounding-box_X (c_6)	entry longitude	Number
bounding-box_Y (c_7)	entry latitude	Number
ftt_class (c_8)	entry class of FTT	String
gnis_class(c_9)	entry class of GNIS	String

Geonames is a gazetteer that contains over six million features categorized into one of nine classes and further subcategorized into one out of 645 feature codes. Geonames was created using data from the National Geospatial Intelligence Agency (NGA) and the U.S Geological Survey Geographic Names Information System (GNIS). Geonames services are available through Web services. Table 6 presents the Geonames export schema (S_D) and Figure 7 shows a fragment of the XML response of this service.

Table 6: Geonames (D_D) Database Web Service Export Schema (S_D).

Attribute name	*Description*	*Data type*
name (d1)	primary name	String
lat (d2)	latitude	Number
lng (d3)	longitude	Number
geonameId (d4)	identifier	String
countryCode (d5)	country code (ISO-3166 2-letter code)	String
countryName (d6)	country name	String
fcl(d7)	Feature type super class code	String
fcode (d8)	Feature type classification code	String
fclName (d9)	Feature type super class name	String
fcodeName (d_{10})	Feature type classification name	String
population (d_{11})	population	Number
alternateNames (d_{12})	alternative names	String
elevation (d_{13})	elevation, in meters	Number
adminCode1 (d_{14})	Code for 1st adm. division	String
adminName1 (d_{15})	Name for 1st adm. division	String
adminCode2 (d_{16})	Code for 2nd adm. division	String
adminName2 (d_{17})	Name for 2nd adm. division	String
timezone (d_{18})	Timezone description	String

```
<?xml version="1.0" encoding="UTF-8" ?>
- <gazetteer-service xmlns="http://www.alexandria.ucsb.edu/gazetteer"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.alexandria.ucsb.edu/gazetteer
    http://www.alexandria.ucsb.edu/gazetteer/protocol/gazetteer-
    service.xsd" version="1.2">
  - <query-response>
    - <standard-reports>
      - <gazetteer-standard-report>
          <identifier>adlgaz-1-1410143-3a</identifier>
          <place-status>current</place-status>
          <display-name>Amazon River - Brazil</display-name>
        - <names>
            <name primary="true" status="current">Amazon
            River</name>
            ...
          </names>
        + <bounding-box>
        - <footprints>
          - <footprint primary="true">
              - <gml:Point>
                - <gml:coord>
                    <gml:X>-49.0</gml:X>
                    <gml:Y>-0.1667</gml:Y>
                  </gml:coord>
                </gml:Point>
            </footprint>
          </footprints>
        - <classes>
            <class thesaurus="ADL Feature Type Thesaurus"
            primary="true">streams</class>
            <class thesaurus="NIMA Feature Designation"
            primary="false">STM (stream)</class>
          </classes>
        - <relationships>
            <relationship relation="part of" target-name="UTM grid GE28"
            />
            <relationship relation="part of" target-name="JOG Sheet
            Number SA22-0" />
            <relationship relation="part of" target-name="Brazil" target-
            identifier="adlgaz-1-19-19" />
          </relationships>
        </gazetteer-standard-report>
      </standard-reports>
    </query-response>
</gazetteer-service>
```

Figure 6: ADL XML response fragment.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
- <geonames style="FULL">
    <totalResultsCount>2</totalResultsCount>
  - <geoname>
      <name>Amazon River</name>
      <lat>-0.1666667</lat>
      <lng>-49.0</lng>
      <geonameId>3407729</geonameId>
      <countryCode>BR</countryCode>
      <countryName>Brazil</countryName>
      <fcl>H</fcl>
      <fcode>STM</fcode>
      <fclName>stream, lake, ...</fclName>
      <fcodeName>stream</fcodeName>
      <population />
      <alternateNames>Amazone,Orellana,Rio Amazonas,Rio Maranon,Rio
        Solimoes,Rio Solimões,Rio el Amazonas,Río Amazonas,Río
        Marañón,Río el Amazonas,Salimoes
        River,Solimoens</alternateNames>
      <elevation />
      <adminCode1>00</adminCode1>
      <adminName1 />
      <adminCode2 />
      <adminName2 />
      <timezone dstOffset="-3.0
        gmtOffset="3.0">America/Belem</timezone>
    </geoname>
</geonames>
```

Figure 7: Geonames XML response fragment.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
d_1	0	0	0	0	77	0	0	0	0
d_2	0	0	0	0	0	0	61	0	0
d_3	0	0	0	0	0	67	0	0	0
d_4	0	0	0	0	0	0	0	0	0
d_5	0	0	0	0	0	0	0	0	0
d_6	0	0	0	0	0	0	0	0	0
d_7	0	0	0	0	0	0	0	0	0
d_8	0	0	0	0	0	0	0	0	55
d_9	0	0	0	0	0	0	0	3	0
d_{10}	0	0	0	0	0	0	0	1645	0
d_{11}	0	0	0	0	0	0	0	0	0
d_{12}	0	0	0	0	33	0	0	0	0
d_{13}	0	0	0	0	0	0	0	0	0
d_{14}	0	0	0	0	0	0	0	0	0
d_{15}	0	0	0	0	0	0	0	0	0
d_{16}	0	0	0	0	0	0	0	0	0
d_{17}	0	0	0	0	0	0	0	0	0
d_{18}	0	0	0	0	0	0	0	0	0

Figure 8: Occurrence matrix between S_C and S_D.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
d_1	0.0	0.0	0.0	0.0	0.049454	0.0	0.0	0.0	0.0
d_2	0.0	0.0	0.0	0.0	0.0	0.0	0.047225	0.0	0.0
d_3	0.0	0.0	0.0	0.0	0.0	0.050464	0.0	0.0	0.0
d_4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.043854
d_9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000110	0.0
d_{10}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.060230	0.0
d_{11}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_{12}	0.0	0.0	0.0	0.0	0.021195	0.0	0.0	0.0	0.0
d_{13}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_{14}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_{15}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_{16}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_{17}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
d_{18}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 9: EMI matrix between S_C and S_D.

In this experiment, we submitted the keyword query "alps". D_C returned 71 entries (R_C) and D_D returned 77 entries (R_D). With both result sets in cache, the mediator generated the occurrence matrix between attributes of the schemas of D_C and D_D. Figure 8 shows the occurrence matrix and Figure 9

the estimated mutual information matrix. Based on the estimated mutual information matrix, Table 7 shows the alignments between attributes of S_C and S_D.

By analyzing the values of the estimated mutual information matrix, the mediator has some evidence of which attributes could be part of a mediated schema. For instance, by observing Table 7, we note that attribute *names* from ADL aligns with *name* from Geonames, and so on for the other attributes in Table 7. Based on these observations, the mediator suggested the mediated schema in Table 8.

Table 7: The aligned attributes (S_C - S_D).

S_C *attributes*	S_D *attributes*
names (c_5)	name (d_1)
bounding-box_X (c_6)	lng (d_3)
bounding-box_Y (c_7)	lat (d_2)
ftt_class (c_8)	fcodeName (d_{10})
gnis_class(c_9)	fcode (d_8)

Table 8: The mediated schema (S_C - S_D).

Attribute name	*Description*	*Data type*
names	entry name	String
bounding-box_X (c_6)	entry longitude	Number
bounding-box_Y (c_7)	entry latitude	Number
ftt_class (c_8)	entry class of FTT	String
gnis_class(c_9)	entry class of GNIS	String

6 CONCLUSIONS AND FUTURE WORK

In this paper, we proposed an instance-based approach for matching export schemas of databases available through Web services. We also described a technique to construct the mediated schema and to discover schema mappings on the fly, based on matching query results. To validate the approach, we discussed an experiment using bookstore databases and gazetteers.

As future work, we intend to extend the case studies to include several data sources. In this context, we plan to investigate the alignment of the included export schemas with the mediated schema and its implications, as the association of the mediated schema with a global instance set derived from existent sources.

ACKNOWLEDGEMENTS

This work was partly supported by CNPq under grants 550250/05-0, 301497/06-0 and 140417/05-2, and FAPERJ under grant E-26/100.128/2007.

REFERENCES

Brauner, D. F., Casanova, M. A., Milidiú, R. L., 2006. Mediation as Recommendation: An Approach to Design Mediators for Object Catalogs. In *Proc. 5th Int. Conf. on Ontologies, DataBases, and Applications of Semantics (ODBASE 2006).* Montpellier, France.

Brauner, D. F., Intrator, C., Freitas, J. C., Casanova, M. A., 2007. An Instance-based Approach for Matching Export Schemas of Geographical Database Web Services. In *IX Brazilian Symposium on GeoInformatics (GeoInfo 2007),* Brazil.

Casanova, M. A., Breitman, K. K., Brauner, D. F., Marins, A. L., 2007. Database Conceptual Schema Matching. In *Computer*, IEEE Computer Society, p. 102 - 104.

Gazola, A., Brauner, D. F., Casanova, M. A., 2007. A Mediator for Heterogeneous Gazetteers. In *XXII Simpósio Brasileiro de Banco de Dados*, João Pessoa, PB, Brazil.

He, B., Chang, C. C., 2003. Statistical schema matching across Web query interfaces. In *Proc. ACM SIGMOD Int'l Conference on Management of Data*. New York, NY, USA. pp.217 - 228.

Janée, G., Hill, L. L., 2004. *ADL Gazetteer Protocol.* Alexandria Digital Library Project. Available at http://www.alexandria.ucsb.edu/gazetteer/protocol.

Papoulis, A., 1984. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill.

Rahm, E., Bernstein, P. A., 2001. A Survey of Approaches to Automatic Schema Matching, In *The VDLB Journal*, vol. 10, pp. 334–350.

Sheth A., Larson, J., 1990. Federated database systems for managing distributed, heterogeneous and autonomous databases. In *ACM Computing Surveys*, v.22 (3), set. 1990. pp.183-236.

Wang, J., Wen,J.-R., Lochovsky, F., Ma, W.-Y., 2004. Instance-based schema matching for web databases by domain-specific query probing. In *Proceedings of the 30th VLDB Conference*, Toronto, Canada.

Wiederhold, G., 1992. Mediators in the Architecture of Future Information Systems. In *IEEE Computer Society Press*, Vol.25, mar. 1992, p. 38-49.