

An Example of the ESTEST Approach to Combining Unstructured Text and Structured Data

Dean Williams, Alexandra Poulouvassilis
*School of Computer Science and Information Systems, Birkbeck College,
University of London
{dean,ap}@dcs.bbk.ac.uk*

Abstract

In this paper, we demonstrate the use of the ESTEST system, which combines the data integration approach with techniques from Information Extraction in order to allow information from ontologies and natural language sources to be integrated with other, semantically related, structured or semi-structured data.

1. Introduction

The Web and the Semantic Web require us to be able to integrate information from a variety of sources, including unstructured text from web pages, semi-structured XML data, structured databases, and metadata sources such as ontologies.

Integration of heterogeneous data sources is a problem that has been addressed by several recent data integration systems, one of which is the AutoMed system being developed at Birkbeck and Imperial Colleges (<http://www.doc.ic.ac.uk/automed>). In data integration systems, several data sources, each with an associated local schema, are integrated to form a single virtual database with an associated global schema. If the data sources conform to different data models, then these need to be transformed into a common data model as part of the integration process. The AutoMed system uses a hypergraph-based data model, the HDM, as its common data model. While current data integration systems allow a variety of heterogeneous structured or semi-structured data sources to be combined and queried by providing an integrated view over them they do not adequately provide for unstructured data.

Information extraction (IE) [1] is a branch of natural language processing that is concerned with extracting pre-defined entities from text and filling in a template with the extracted information. While information retrieval works by identifying and

retrieving documents that are of interest, IE extracts data from within documents.

Our approach combines Information Extraction technology with the AutoMed data integration system. The resulting system, called ESTEST, makes use of existing metadata such as database schemas, natural language ontologies and domain-specific ontologies, to assist the Information Extraction process. Once new data and new metadata have been extracted from the text, this is integrated with the existing data and metadata. This extraction and integration process can then be reiterated as required.

2. AutoMed

Up to now, most data integration approaches have been either **global as view (GAV)** [2,3,4] or **local as view (LAV)** [5,6,7]. In contrast AutoMed supports **both as view (BAV)** integration [8]. This is based on the use of reversible sequences of primitive schema transformations, called transformation **pathways**. From these pathways it is possible to extract a definition of the global schema as a view over the local schemas (GAV), and it is also possible to extract definitions of the local schemas as views over the global schema (LAV).

The basis of AutoMed's BAV approach is a low-level **hypergraph-based data model (HDM)** [9,10]. Facilities are provided for defining higher-level modeling languages in terms of this lower-level HDM: for example, previous work has shown how relational, ER, UML and XML data models can be so defined [11,12]. An HDM schema consists of a set of nodes, edges and constraints, and so each modeling construct of a higher-level modeling language needs to be defined as some combination of HDM nodes, edges and constraints. For any modeling language M specified in this way (via the API of AutoMed's Model Definitions Repository [13]), AutoMed automatically provides a set of primitive schema transformations that

can be applied to schema constructs expressed in M . In particular, for every construct of M there is an `add` and a `delete` primitive transformation which respectively add to, or delete from, the underlying HDM schema the corresponding set of nodes, edges and constraints. For those constructs of M which have textual names, there is also a `rename` primitive transformation.

3. The ESTEST System

ESTEST makes use of AutoMed for its data integration aspects and of an existing IE system to extract structured information from text. We have extended the data models covered by AutoMed to include RDF and RDF Schema [14] and have implemented an HDM repository for the new data ESTEST discovers [15]. We refer the reader [16,17] for a detailed description of the design of the ESTEST but an outline follows in this section and in Figure 1.

Initial Integration. The available data sources other than the text (e.g. structured, semi-structured, and domain ontologies) are first integrated into a single virtual global schema, using AutoMed schema transformation pathways. This global schema can then be queried by submitting queries expressed in AutoMed's IQL query language to its global query processor [18].

Create Data to Assist the IE Process. The global virtual resource can be used to provide data which assists the IE process. For example, lists of entities can be created by submitting queries to the global schema. These lists can then be used by the 'named entity recogniser' components of the IE system (see below). From the global schema, we can also extract information to create templates for grammars e.g. from table and column names in relational schemas.

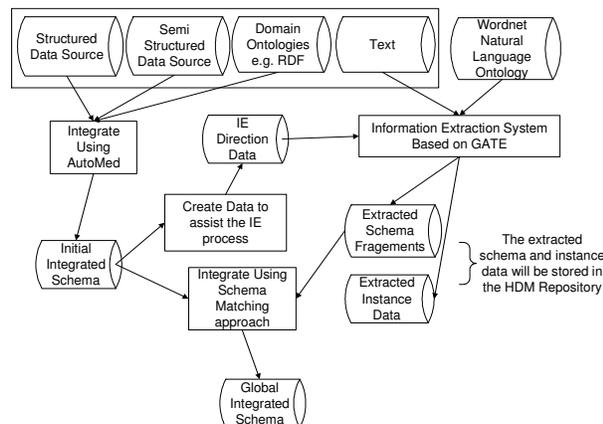


Figure 1. Overview of the ESTEST System

Information Extraction System Based on GATE.

The IE component of ESTEST is based on the Sheffield GATE system (<http://www.gate.ac.uk>) which allows for a sequence of language processing components to be assembled and marks up annotations on the input text. GATE's language processing components include standard components such as sentence splitters and named entity recognisers. Bespoke components can also be constructed and integrated with the existing standard ones. We are developing new IE components which will generate templates for the extraction, based on the assumption that the entity types in the existing schema and domain ontologies will be at least a significant subset of the entity types for which we wish to extract information from the text. We are also developing a WordNet component which will make use of its synonym and hyponym structures to allow for alternative lists for words to be found in cases where the textual descriptions of schema elements is restricted, for example to a word in a column name.

The result of the IE process is a set of named annotations over sections of the text. These annotations can be thought of as discovered fragments of schema. These fragments and the text to which they refer are stored in the Extracted Schema Fragments and Extracted Instance Data store, respectively, both of which are implemented using our HDM repository.

Integrate New and Existing Metadata Using Schema Matching. A schema-matching algorithm takes each new extracted schema fragment and finds its best match with respect to the global schema, or allows for it to be appended to the global schema. Unlike many other schema matching applications, there will not be much structural information available to assist the matching algorithm and we will rely primarily on element names. However, we will also experiment with using the new instance values extracted, looking to see if these are already present in the extents of candidate schema entities and using any presence to provide evidence of a semantic match.

The Extracted Schema Fragments are integrated with the virtual global schema by means of AutoMed schema transformation pathways, which are automatically generated by the above schema matching process. The data in the Extracted Data store can thus be treated as a new AutoMed data source, and queries posed against the virtual global schema will automatically make use of this new data.

4. An Example

Road Traffic Accident reports in the U.K. are reported using a flat-file format known as STATS-20. In STATS-20, a record exists for each accident, and following this there are multiple records for the people and vehicles involved in the accident. Suppose that the structured information relating to all accidents in a particular region over a particular period has been stored in a relational database, AccDB, while the corresponding free text portions of the accident reports are available as a separate text file. In our simplified example here, AccDB consists of two tables:

```
Accident (accNo, road, roadType)
Vehicle (accNo, vehNo, vehType)
```

In the `Accident` table, each accident is uniquely identified by an `accNo`, the `road` attribute identifies the road the accident occurred on, and `roadType` indicates the type of road. There may be zero, one or more vehicles associated with an accident, and information about each them is held in a row of the `Vehicle` table. Here `vehNo` uniquely identifies each vehicle involved in an accident and thus `accNo, vehNo` is the key of this table. (In the real data, there are also tables about the casualties involved in the accident, other people involved --- the driver, passengers, etc., and information about the road and weather conditions.) An example of the text entries collected in the separate text file might be:

```
23781FOX RUNS INTO ROAD CAUSING VEHICLE
23781TO SWERVE VIOLENTLY AND LEAVE ROAD
```

Thus, in order to answer queries such as "on which roads were there accidents caused by animals and what animals caused them?" we would need to retrieve and integrate information from the text as well. The WordNet ontology can first be used to extract information about possible animals. Suppose therefore we have an RDF representation of WordNet which contains triples of the following form describing animals:

```
concept_0, wordForm, animal
concept_1, hyponymOf, concept_0
concept_1, wordForm, carnivore
concept_2, hyponymOf, concept_1
concept_2, wordForm, canine
concept_3, hyponymOf, concept_1
concept_3, wordForm, feline
concept_4, hyponymOf, concept_2
concept_4, wordForm, dog
concept_5, hyponymOf, concept_2
concept_5, wordForm, fox
concept_6, hyponymOf, concept_3
concept_6, wordForm, cat
```

We now need to construct a (virtual) Integrated Schema which combines the information in the relational database, AccDB, with the information in this RDF data source. In our example, the Integrated Schema is a relational schema, but in practice it may be expressed in any data model supported by AutoMed. These are the tables of the Integrated Schema:

```
Accident (accNo, road, roadType)
Vehicle (accNo, vehNo, vehType)
Animals (animal)
AnimalInRoad (accNo, animal)
```

The information in `Accident` and `Vehicle` will be sourced from the relational database, the information in `Animals` from the RDF data source, and the information in `AnimalInRoad` from the Extracted Instance Data store created by the ESTEST information extraction system. We need to define two pathways, one from the RDF data source, RDFDS, to the Integrated Schema, IS, and one from AccDB to IS. The following AutoMed pathway, `RDFDS → IS`, transforms RDFDS into IS. It first creates a new table `Animals (animal)`, which contains the set of animals in RDFDS. Here, the IQL sub-query `linkedTo('hyponymOf', 'concept_0')` returns the set of concepts in the RDF source, which are recursively linked to `concept_0` (ie `animal`) by edges labeled `hyponymOf`.

We assume that it can be translated by the RDFDS wrapper into a local query on the RDFDS. All the RDF information is then dropped from the schema, and `contract` transformations are used since this information cannot be reconstructed from the remaining schema constructs.

A new table `AnimalInRoad (accNo, animal)` is then added the schema, and an `extend` transformation is used since this information cannot be derived from the RDFDS (this is a simplified presentation of how relational schemas are actually encoded in AutoMed, which suffices for the purposes of this paper and we refer the reader to [11,8] for the full details). The transformation finally extends the schema to include the `Accident` and `Vehicle` tables (whose content will be sourced from AccDB):

RDFDS → IS:

```
addTable (<<Animals, animal>>
  [a|c<-linkedTo('hyponymOf', 'concept_0');
    (c, 'wordForm', a) <- <<Triples>>]);
contractRDFEdge (<<Triples>>);
contractRDFNode (<<URI>>);
contractRDFNode (<<Literal>>);
contractRDFNode (<<Blank>>);
extendTable (
  <<AnimalInRoad, accNo, animal>>);
```

```

extendTable(
  <<Accident, accNo, road, roadType>>);
extendTable(
  <<Vehicle, accNo, vehNo, vehType>>);

```

The transformation pathway from AccDB to IS is simpler and consists of extend steps to include the missing Animals and AnimalInRoad tables:

AccDB → IS:

```

extendTable(<<Animals, animal>>);
extendTable(
  <<AnimalInRoad, accNo, animal>>);

```

ESTEST can now use the Integrated Schema IS to extract information about animals in roads from the text. The IE system will have been given a named entity recogniser for animals, and grammar rules for detecting when animals cause accidents (as opposed to when they are victims of accidents, say). An example of the kind of JAPE grammar rule used might be:

```

Macro: CAUSEACTION
// e.g. "RUNS INTO", "RUNS ONTO",
//      "WALKS IN FRONT OF"
(
  ({Token.string == "RUNS"} |
   {Token.string == "WALKS"} |
   {Token.string == "JUMPS"} )
  (SPACE)?
  ({Token.string == "INTO"} |
   {Token.string == "ONTO"} |
   {Token.string == "IN FRONT OF"} )
  (ANIMAL)
)

```

In this example rule, possible combinations of text such as "RUNS IN FRONT OF" or "WALKS INTO" are combined with the ANIMAL token which will be populated by the named entity recogniser based on a list of animal words from IS. The resulting text annotations will be extracted and, for this example, the annotations will be:

```

00 02 animal      ("FOX")
00 17 causeaction ("FOX RUNS INTO ROAD")

```

The resulting Extracted Schema Fragments are as follows (expressed as edges in the HDM data model):

```

<<_, accNo, animal>>
<<_, accNo, causeaction>>

```

which respectively express the fact that there is an (unnamed) association between accNo and animal, and between accNo and causeaction. The corresponding Extracted Instance Data consists of one

record [23781, "FOX"] within the extent of <<_, accNo, animal>>, and one record [23781, "FOX RUNS INTO ROAD"] within the extent of <<_, accNo, causeaction>>. It will now be straightforward for the schema matcher to identify that the HDM edge <<_, accNo, animal>> matches the table AnimalInRoad(accNo, animal) in the Integrated Schema IS. A transformation pathway from the Extracted Schema Fragments (ES) to IS will then be automatically generated which contracts the <<_, accNo, causeaction>> HDM edge, replaces the <<_, accNo, animal>> HDM edge by the AnimalInRoad(accNo, animal) table (by means of an add followed by a delete step), and uses a series of extend steps to add the missing Animals, Accident and Vehicle tables:

ES → IS:

```

contractHDMEdge(<<_, accNo, causeaction>>);
addTable(<<AnimalInRoad, accNo, animal>>,
  <<_, accNo, animal>>);
deleteHDMEdge(<<_, accNo, animal>>,
  (<<AnimalInRoad, accNo, animal>>));
extendTable(<<Animals, animal>>);
extendTable(
  <<Accident, accNo, road, roadType>>);
extendTable(
  <<Vehicle, accNo, vehNo, vehType>>);

```

The final result is thus three AutoMed transformation pathways from the three heterogeneous data sources to the virtual Integrated Schema IS: RDFDS → IS, AccDB → IS, ES → IS. AutoMed's global query processor is able to traverse the reverse pathways from the IS to the three sources in order to create a view definition for each schema construct in IS in terms of the schema constructs in the three sources.

When a query *q* on IS is submitted to the global query processor for evaluation, it substitutes these view definitions into *q* in order to reformulate it into a query over the data sources which, after some optimisation, can be evaluated over the local data sources [19] (more specifically, after *q* has been reformulated and optimised, sub-queries of it are submitted to the appropriate data source wrappers for translation into the data source query languages and evaluation at the data sources. The wrappers translate sub-query results back into the IQL type system and the global query processor undertakes any further necessary post-processing of the query).

We can therefore pose this IQL query on IS to find “on which roads were there accidents caused by animals and what animals caused them?”:

```

[ (r, a) | (acc, a) <-
  <<AnimalInRoad, accNo, animal>>;

```

```
(acc,r,t)←-
  <<Accident,accNo,road,roadType>>)]
```

The views generated for the global constructs

```
<<AnimalInRoad,accNo,animal>> and
<<Accident,accNo,road,roadType>> by
AutoMed's global query processor are ES:
<<\_,accNo,animal>> OR AccDB:Void
OR:RDFS:Void and ES:Void OR
AccDB:<<Accident,accNo,road,roadType>> OR
RDFS:Void the prefix ES, AccDB or RDFS indicates
the schema from which the construct is sourced.
Standard query rewriting techniques can be applied to
generated view definitions to remove instances of
Void. In the above case the two views simplify to just
ES:<<\_,accNo,animal>> and AccDB:
<<Accident,accNo,road,roadType>>.
```

We note that the query above joins information from the AccDB and the Extracted Instance Data and could not be answered using either source alone.

5. Conclusions

In this paper we have demonstrated the approach of the ESTEST system, which extends traditional data integration systems by combining the AutoMed data integration approach with IE technology in order to allow information from ontologies and natural language sources to be integrated with other, semantically related, structured or semi-structured data. ESTEST uses related schema and ontology information to assist the IE process from text.

The information extracted is integrated as a new data source with respect to a virtual global schema. We are currently finishing a first implementation of the ESTEST system and will evaluate its effectiveness in a number of application areas, including Road Traffic Accident Data, Operational Intelligence Police Reports and analysis of Financial Websites. There are a number of research directions for further work, including the use of metadata to drive IE, and schema matching where only text and metadata is available.

6. References

- [1] D. Appelt, "An Introduction to Information Extraction", 1999, Artificial Intelligence Communications
- [2] S.S. Chawathe *et al.*, "The {TSIMMIS} Project: Integration of Heterogeneous Information Sources", Proc. 10th Meeting of the Information Processing Society of Japan", 1994, pp7-18
- [3] M. Templeton, H.Henley, E.Maros and D.J. Van Buer, "InterViso: Dealing With the Complexity of Federated

Database Access", The VLDB Journal, 1995, vol 4, num 2, pp287-317

[4] M.T. Roth and P. Schwarz, "Don't Scrap It, Wrap It! A Wrapper Architecture for Data Sources", Proc. VLDB'97, Athens, Greece, 1997, pp266-275

[5] A.Y. Levy, A. Rajamaran and J.Ordille, "Querying heterogeneous information sources using source description", Proc. VLDB'96, 1996, pp252-262

[6] A.Y. Levy, "Logic-based techniques in data integration", Logic Based Artificial Intelligence, 2000, Kluwer Academic Publishers, ed. J. Minker

[7] I. Manolescu, D. Florescu and D. Kossmann, "Answering {XML} Queries on Heterogeneous Data Sources", Proc. VLDB'01, 2001, pp241-250

[8] P.J. McBrien, P.J. and A. Poulouvassilis, "Data Integration by Bi-Directional Schema Transformation Rules", Proc. ICDE'03, 2003

[9] A. Poulouvassilis and P.J. McBrien, "A General Formal Framework for Schema Transformation", 1998, "Data and Knowledge Engineering", vol 28, num 1, pp47-71

[10] P.J. McBrien and A. Poulouvassilis, "Automatic migration and wrapping of database applications a schema transformation approach", 1999, Proc. ER'99, LNCS 1728, pp96-113

[11] P.J. McBrien and A. Poulouvassilis, "A Uniform Approach to Inter-Model Transformations", Proc. CAISE'99, LNCS 1626, 1999, pp333--348

[12] P.J. McBrien and A. Poulouvassilis, "A Semantic Approach to Integrating {XML} and Structured Data Sources", Proc. CAISE'01, LNCS 2068, 2001, pp330-345

[13] M. Boyd and P.J. McBrien and N. Tong, "The AutoMed Schema Integration Repository", Proc. BNCOD02, LNCS 2405, 2002, pp42-45

[14] D. Williams and A.Poulouvassilis, "Representing RDF and RDF Schema in the HDM", Technical Report, Automated Project, 2003

[15] D. Williams, "The AutoMed HDM Data Store", institution="Automed Project", Technical Report, Automated Project, 2003

[16] D. Williams and A.Poulouvassilis, "Combining Data Integration with Natural Language Technology for the Semantic Web", Proc. Workshop on Human Language Technology for the Semantic Web and Web Services, at ISWC'03, 2003

[17] D. Williams and A.Poulouvassilis, "Combining Data Integration with Natural Language Technology for the Semantic Web", Technical Report, Automated Project, 2003

[18] Jasper, E. Poulouvassilis, A. and Zamboulis, L.", "Processing IQL Queries and Migrating Data in the AutoMed toolkit", Technical Report, Automated Project, 2003

[19] E. Jasper, N. Tong, P. McBrien and A. Poulouvassilis, "View Generation and Optimisation in the AutoMed Data Integration Framework", Technical Report, AutoMed Project, 2003