Document Interrogation: Architecture, Information Extraction and Approximate Answers

Soraya Abad-Mota[1,2]

[1] University of New Mexico (soraya@cs.unm.edu)
[2] Universidad Simón Bolívar (abadmota@usb.ve)

Abstract. We present an architecture for structuring and querying the contents of a set of documents which belong to an organization. The structure is a database which is semi-automatically populated using information extraction techniques. We provide an ontology-based language to interrogate the contents of the documents. The processing of queries in this language can give approximate answers and triggers a mechanism for improving the answers by doing additional information extraction of the textual sources. Individual database items have associated quality metadata which can be used when evaluating the quality of answers. The interaction between information extraction and query processing is a pivotal aspect of this research.

1 Introduction

Many organizations produce large amounts of documents and their contents never reach the operational databases or the data warehouses of the organization. With the world-wide accessibility to the web these documents are made available to a wide audience, but browsing through them manually is cumbersome, at best. The definition of the *semantic web* [1], has led to fascinating possibilities of research in making explicit the semantics of terabytes of unstructured data available today. Our research seeks to improve the use of these unstructured sources, in particular, textual sources.

Florescu et al. in [2] have clearly defined three tasks in databases related to information management on the www. One of these tasks is *information extraction and data integration*. The data integration task deals with processing queries that require access to heterogeneous data. The queries are posed against data sources after information extraction has taken place. The extraction task is performed by a set of *wrapper programs* and the integration is addressed by *mediator systems* (see [3]).

In this research, each document is a data source from which data are extracted. Each document type has a schema of its contents and all document type schemas that belong to the same organization are integrated into a global schema. The individual document schemas are used in the task of information extraction from the documents to populate a single relational database.

As a motivating example let us think of a university where each year there is a period when faculty apply for a sabbatical leave for the following academic year. Each sabbatical application is a document which includes the dates, a plan and the institution where this plan will be carried out. The sabbatical application is a type of document which has an associated conceptual schema; each particular application instantiates this schema.

There is not a schema for each instance of a document, but for each type of document. The document type and global schemas are expressed in an extension of the ER model as defined in [4].

The *broad information extraction* phase of the architecture proposed in this work (see figure 1) uses some annotated documents of a particular type to generate extraction rules. The rules are used to extract data slots from the documents. The data extracted can then be used to populate the portion of the integrated database schema which corresponds to the document type schema. When another type of document is processed, new extraction rules are generated; all the documents of this new type are subject to the information extraction process and another portion of the global schema in the relational database is populated with extracted data.

The main idea in this ongoing work is to develop a mechanism that takes advantage of the contents of the documents of an organization. The approach proposed is to structure these contents in a database that is semi-automatically populated using information extraction techniques. The global and individual schemas of documents are produced manually. The *broad information extraction* phase allows a massive population of the database. Later, with the user queries and the answers provided to these queries, more information might be extracted from the documents in an attempt to improve previous answers. It is therefore the interaction between information extraction and query processing a crucial aspect of this research.

In the next section the architecture for document interrogation is briefly described with an emphasis on its ontology. Section 3 contains a description of the information extraction results of this work. In section 4 we introduce the need for a mechanism to provide approximate answers and present a discussion about data quality in this architecture. In section 5 we present a case study which illustrates the components of the arquitecture and their use. Finally, section 6 contains the conclusions and a plan for future work.

2 The Architecture

The Document Interrogation Architecture (*DIA*) presented in [4] and shown in figure 1, addresses the problem of providing mechanisms for structuring the data contained in textual documents and providing an ontology-based language to interrogate the documents through the database built with extracted data. Our proposed architecture deals with data integration from different data sources but within an organization.

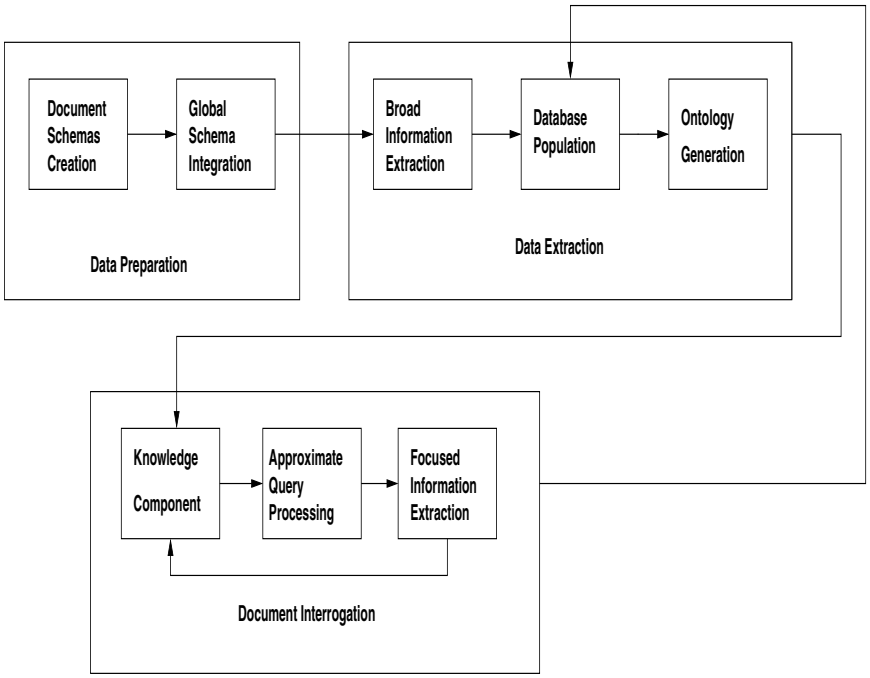

Fig. 1. Document Interrogation Architecture

The architecture has three phases. The preparation phase is where training and test documents are annotated and the information extractor is built, by generating extraction rules which are learned from the training examples. The second phase extracts data items from the documents and populates a relational database. Once the database is populated with data, the user may ask questions; the answers are provided by the third phase, the document interrogation phase of the architecture. When computing an answer, the available data is analyzed in order to determine missing data or differences in their quality with respect to the query being posed.

The database of *DIA* is represented at three levels of abstraction: the relational representation, the ER representation in which the schema integration is done, and a higher level, ontology-based representation (see [5]). This latter representation is the basis for the interrogation language. The concepts in the ontology are extended with concepts defined by the user and all these concepts are mapped to the global ER schema of documents and then to the relational database in order to answer queries.

The ontology of *DIA* is initially built with all the concepts contained in the global ER schema of the documents and their relationships. Mostly concepts and not instances are included in the ontology; the relational database contains all the instances. Suppose for example, that we have in the ER schema an entity COUNTRY with instances stored in the relational database. The initial ontology

contains the concept COUNTRY, defined as an aggregation of the attributes of the entity, but without instances.

The ontology-based document interrogation language, ODIL, is defined to interrogate and extend the concepts contained in the database. We presented the specification of the language and the mappings between the three models of the database in [6].

Following the example above, the user may extend the ontology by defining a new concept, REGION, as an aggregation of countries; this definition will be stored in the ontology with the mappings between the new concept and the corresponding concepts in the ER schema. If the user defines a particular region, for example, the "Andean Region", with the five countries that comprise it, that instance is stored in the ontology as an instance of REGION, with references to the five instances of COUNTRY that are contained in it. The user may now ask queries about REGION and the mappings between this new concept and the ER schema will let the system compose an answer using the data for the countries that compose a region. This is why we say that the ontology grows with the user queries.

3 Information Extraction in DIA

Information Extraction is a fairly recent research field which has received the attention of researchers from different areas such as: Natural Language Processing (NLP), (see [7] for empirical methods in NLP and [8] for a summary of specific information extraction methods), Machine Learning (ML) (see [9] and [10] for a description of some techniques) and Databases (DB) ([2], [11] and [12]).

Information Extraction is the process by which several pieces of relevant data are extracted from natural language documents. This process includes: locating the relevant data, extracting the phrases which contain the data, separating the facts contained in these phrases and structuring the data in a database for future use.

The approaches to IE taken by the researchers in the areas mentioned above are varied. The natural language processing and machine learning approaches to IE expect mainly natural language text as input and apply empirical methods to it. The research in IE from the database perspective has concentrated on extracting data from web sources of semi-structured data; these include two main approaches, conceptual model-based extraction as developed in the Data Extraction Group of BYU ([11]) and the "Wrapper-Mediator" Approach ([2]). The field of IE has experienced great acceleration recently in response to the increasing amount of textual documents available on the web.

The architecture proposed in [8] defines a generic architecture for an Information Extraction System, which has five phases: tokenization and tagging, sentence analysis, extraction, merging, and template generation. A very effective mechanism for implementing the extraction phase is the definition of extraction rules using patterns. The different alternatives for generating these rules are commonly called *corpus-based methods for learning IE patterns*. The methods

to produce the extraction patterns do it in three steps, sentence analysis and preprocessing, training/test case generation, and learning algorithm. Several successful tools have used this technique (see [9]), some representative examples of which are: RAPIER ([13]), WHISK ([14]), and $(LP)^2$ ([15]).

Both "wrappers" and the *corpus-based methods for learning IE patterns* work very well on structured text, a precision near 100% is not unusual. But the former assume a strict regularity in the format of the document; therefore if the format changes, the "wrapper" needs to be modified.

The database approach to IE exploits the problem domain knowledge in contrast with most of the ML approaches to IE, which focus on the natural language semantics of text. In particular, the work of Embley et al. in [11], even though motivated by web page exploration, is an attempt to build an IE system which strongly relies on a conceptual data model of the domain of the source documents. The challenge of this approach is how to represent the domain knowledge in a compact and simple form and how to find ways of acquiring such knowledge automatically.

We have developed an extractor for *DIA* as a combination of the preprocessing activities of the NLP extractors (as classified in [16]), a variation of the learning algorithm of$(LP)^2$ ([15]), and the use of the document schema to build the database tuples. The method for generating extraction rules is fully implemented. Our learning algorithm was tested in several well-known corpora and we obtained similar results to the best reported algorithms (see results of the Pascal Challenge in [17] and our results in [18]).

4 Data Quality and Approximate Answers

The canonical approach for using a database to provide answers to queries is to assume that the database has *the whole truth and nothing but the truth* ([19]). The data which populates the database of DIA is obtained through information extraction procedures from a set of documents. The extracted data might not be complete due to possible mistakes in the extraction process (mostly omissions) and due to the data items in the documents having varying degrees of accuracy or recency (see [20] for a good summary of data quality definitions).

Most of the approaches to data quality within the data integration literature consider the overall quality of a data source. We are interested in a lower granularity measure of quality. Each data item must have a description which includes: the source of the item, the date, the units if it is a measure, the object being described by it, among other metadata. The work of Rakov in [21] considers the quality of database components rather than the whole database. This is a useful measure, but we also want to produce qualitative measures that we can express in text, so that we can go back to the documents to extract more data, based on the text that describes which data are missing.

The problem is that we want to give an answer to a query but in order to build the answer we might use several data items which have different quality metadata associated with them. For example, if we want to answer what the

total population of the Andean Region is, we need the current population of the five countries which constitute this region, Bolivia, Colombia, Ecuador, Peru and Venezuela. But we might not have data for the population of Bolivia, we might have census data for Venezuela from 2000 and data for the three other countries from 2004. If we consider these differences a canonical approach would give no answer. If we add up all the population values that we have, Bolivia is not represented in that sum and the other values do not correspond to the same year. Our system should provide the sum of these values as an answer, with a measure of the quality of the answer. Additionally, it will provide a textual phrase that describes the missing data, which triggers further information extraction from the documents.

Approximate query processing in *DIA* distinguishes two cases. The first case occurs when some of the components of the answer are missing. The other case is when there is data for all the components, but their individual quality measures differ greatly from each other. We discuss the first case in the following paragraphs.

When data are missing we have two options, one is to compute an answer without the missing components, but present it as a bound for the real answer. In the case of the total population of the Andean Region, if we add up the existing data for four countries, that sum would be a lower bound for the total population, because the population of Bolivia is greater than zero.

Another option is to try to estimate the missing components. In the statistical literature there are several ways of estimating missing data; for an application of one of these methods to query processing in statistical databases see [22]. Some of the ways of computing missing data consider total uncertainty, but if there is additional information which can be used, the query processor could compute an informed guess. In particular, in the context of an answer which needs several components, it is very relevant to know the relative order of the components and the percentage of the total which can be attributed to the missing components.

The web is a good source of general facts and we can use it to bootstrap our estimating method. On a quick look at the web, we did a search of Bolivia and the other four andean countries. Since the query requires the population of the five countries and we only have these data for four countries, we searched all five countries and got the number of pages about those countries. Then we added one word to the name of each country and performed more searches.

We tried 5 searches with only the name of the country and 95 searches of one word together with the names of the countries. With the results of these trials we computed the global ranking of the countries in terms of the number of pages found by the search engine for each country. The computed order of the five countries was: Colombia, Venezuela, Peru, Ecuador and Bolivia, almost the real order of population size which is: Colombia, Peru, Venezuela, Ecuador and Bolivia.

The computed value of the population of Bolivia, used in the calculation of the total population of the region is an estimate and we include it as such in

the database; it is a statistical measure of the missing value. Together with the estimated value we store metadata which describe how it was computed.

These first experiments were performed using words selected manually, but we can use words from the ontology which are close to the concept of the "population of Bolivia" and their synonyms. Additionally, we can also select words from the conceptual schemas or we can generate words by other means.

In this case we used the web as a source of additional information, in general we can use other sources of information that are available.

Regarding the quality of the answer, at first we can say that one out of five components of the answer to the query about the population of the Andean Region is missing, that is, 20 % is missing, or equivalently, four out of five components are correct. This relation could be taken as a measurement of quality, which we could call the *uniform metric*, analogous to the *freshness-rate metric* used for data freshness in [23]. However the 20 % figure is improved to 16.12 % when we perform the procedure of searching words on the web and compute the average proportion of the number of pages about Bolivia with respect to the number of pages for the whole Andean Region. We call this estimate the *bootstrap metric*. The real data reveals that the Bolivian population is 7.43 % of the total of the region. Therefore the distance between the *bootstrap metric* and the real proportion is smaller than that between the real proportion and the *uniform metric*.

There is a qualitative difference between these two metrics. Whereas the *uniform metric* provides no information with regard to the ranking of the missing components, the computations involved when calculating the *bootstrap metric* provide additional information on the ranking of the missing components. For instance, in the case under consideration, Bolivia ranked fourth in nine out of 20 words and ranked fifth in eleven out of those 20 words. We think that this ranking observation deserves further research.

5 A Case Study: CENDA

CENDA is the Center for Document Archives of Universidad Simón Bolívar (USB). This office receives boxes of paper documents from all the organizational units of the university. The documents contained in these boxes are official and are archived for preservation and query answering purposes.

The personnel of CENDA have to manually inspect the boxes, classify the documents contained in them, verify for document duplicity and place each document physically on a drawer or a shelf. A big effort to digitize the documents is underway, but the documents are saved in an image format, which is not the best way to exploit the contents of them or to find answers to queries, because it still needs a great deal of human intervention. A typical query takes 3 days in average to be attended, if the documents needed are already in place.

The *DIA* architecture provides a solution for CENDA's needs. Instead of working with an image of the document, each document is scanned and converted into text with an OCR software. The most requested documents processed by

CENDA can be categorized into 17 major classes. Examples of these document classes are: resolutions of the councils of the university, appointments, admissions, student records, among others. In order to use *DIA* in this problem, each document class needs a document schema. All the document schemas are then integrated into the global schema for the database to be used as the primary means of query answering.

The initial ontology for CENDA is defined using the results of the database construction. The *ODIL* language is then used to pose queries and to extend the ontology.

When a user comes to CENDA with a query, the database of *DIA* is the first resource to be accessed, and the answer could be facts contained in the database or a reference to a document. In many cases, the user needs a copy of the document which contains the answer. The documents would still be classified and stored, but the answers can be processed in the database and the document could be directly accessed and retrieved using the reference to it that resides in the database. This way a huge amount of manual work can be saved for validation of the data stored which was automatically processed.

The answer to a user query in CENDA can often be found in several documents. This is due to the nature of these official documents where several instances have to approve the same fact. For example, an application for sabbatical leave is presented in several documents, one from the faculty member, one from her department and yet another one being the official communication which informs the faculty of the decision regarding the approval or the rejection of the sabbatical. For efficiency reasons and since these documents are somewhat redundant, only one of them is scanned and "loaded" into the database.

The ability of *DIA* to reason about the quality of its answers is particularly useful in the CENDA environment. When an approximate answer is produced it could be due to a missing component. In order to overcome the absence of this missing component in cases like the one just described, where several documents contain redundant information, the documents which were not scanned can be processed to try to extract additional information which might provide a more accurate answer.

Currently there is a teamwork that will select a few classes of documents to be scanned and an integrated schema is being designed to perform a test of the use of the *DIA* architecture in CENDA.

6 Conclusions and Future Work

The *DIA* architecture was defined to take advantage of the contents of textual documents, restricting human intervention to a minimum to make it feasible. One of the aspects in which we try to reduce interaction with a user is in the use of information extraction methods, which learn from a set of annotated examples. We have designed a broad information extractor for DIA, the generation of extraction rules has been implemented and tested and it provides good results; we are working on reducing the number of training examples. The implementation

of the module of the extractor that inserts the tuples in the database, using the document schemas, is under construction.

The motivating goal of this research is to provide a tool for humans to query the contents of a set of related documents, without having to manually inspect all the documents. The definition of a high-level language, based on the main concepts abstracted from the documents is a response to this goal. The language has an SQL-like syntax and its main value is in the way it is processed, with the ability to provide approximate answers, to evaluate the quality of the data used in providing an answer and with a mechanism for searching for more data in the documents, if the answer is not good enough.

We have a prototype implementation of the query processor of ODIL. The prototype includes the primitives for defining new concepts and their mapping to the ER schema. It also includes some of the procedures to compute answers. Our immediate plan is to complete the prototype with the generation of approximate answers and also with the evaluation of the metrics described. For the answers which had a missing component, a phrase describing the missing data should be produced, so that a procedure could extract more data from the documents. After concluding the implementation we will run tests to evaluate the improvement in the answers and the ability of the system to extract more data guided by the descriptive phrase.

DIA provides data integration when determining the combination of data required to provide an answer to a document query. Query processing is done over a database of facts obtained from the documents, but since the facts that populate the database do not have a homogeneous quality, a metric for the quality of the combined answer must be searched. In particular, there might be differing qualities of the stored data which are used in the computation of an answer. These qualities are relative to the query being processed. We need to find a metric that will take into account the metadata associated with each component of the answer and that will give a quality value to it. In this aspect the work of Motro et al. in [24] is relevant.

Our system also provides a description of the incomplete or inaccurate data, to perform further extraction from the documents guided by the description of the missing data. This is what we call *focused information extraction* and to the best of our knowledge this is a novel approach which has not been explored.

References

1. James Hendler Tim Berners-Lee and Ora Lassila, "The semantic web," *Scientific American*, may 2001.
2. Daniela Florescu, Alon Levy, and Alberto Mendelzon, "Database techniques for the world-wide-web: A survey," *SIGMOD Record*, vol. 27, no. 3, pp. 59–74, September 1998.
3. G. Wiederhold, "Mediators in the architecture of future information systems," *IEEE Computer*, vol. 25, no. 3, pp. 38–49, 1992.
4. Soraya Abad-Mota and Paul A. Helman, "Dia: A document interrogation architecture," in *Proceedings of the Text Mining Workshop in conjunction with the Sixth*

Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-02), 2002, pp. 35–45.

5. N. Guarino, Ed., *Formal Ontology and Information Systems*, IOS Press, 1998.
6. Soraya Abad-Mota and Paul A. Helman, "Odil: Ontology-based document interrogation language," in *Proceedings of the 2004 Information Resources Management Association International Conference*, Mehdi Khosrow-Pour, Ed. IRMA, 2004, pp. 517–520, Idea Group Publishing.
7. Eric Brill and Raymond J. Mooney, "An overview of empirical natural language processing," *AI Magazine*, , no. Winter, pp. 13–24, 1997.
8. Claire Cardie, "Empirical methods in information extraction," *AI Magazine*, vol. 18, no. 4, pp. 65–80, 1997.
9. Ion Muslea, "Extraction patterns for information extraction tasks: A survey," in *AAAI-99 Workshop on Machine Learning for Information Extraction*, July 19 1999, Orlando, Florida.
10. Un Yong Nahm and Raymond J. Mooney, "A mutually beneficial integration of data mining and information extraction," in *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, July 2000.
11. David W. Embley, Douglas M. Campbell, Y. S. Jiang, Stephen W. Liddle, Yiu-Kai Ng, Dallan Quass, and Randy D. Smith, "Conceptual-model-based data extraction from multiple-record web pages," *Data Knowledge Engineering*, vol. 31, no. 3, pp. 227–251, 1999.
12. David W. Embley, "Toward semantic understanding: an approach based on information extraction ontologies," in *CRPIT '04: Proceedings of the fifteenth conference on Australasian database*, Darlinghurst, Australia, Australia, 2004, pp. 3–12, Australian Computer Society, Inc.
13. Mary Elaine Califf, "Relational learning techniques for natural language extraction," Tech. Rep. AI98-276, University of Texas, 1 1998.
14. Stephen Soderland, "Learning information extraction rules for semi-structured and free text," *Machine Learning*, vol. 34, no. 1-3, pp. 233–272, 1999.
15. Fabio Ciarvegna, "(lp)2, an adaptative algorithm from information extraction from web-related texts," in *Proceedings of the IJCAI-2001 Workshop on Adaptative Text Extraction and Mining*. IJCAI-2001, August 2001.
16. A. Laender B., Ribeiro-Neto, A. da Silva, and J. Teixeira, "A brief survey of web data extraction tools," *SIGMOD Record*, vol. 31, no. (2), pp. 84–93, 2002.
17. Neil Ireson, Fabio Ciarvegna, Marie Elaine Califf, Dayne Freitag, Nicholas Kushmerick, and Alberto Lavelli, "Evaluating machine learning for information extraction," in *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*. IJCAI-2001, August 2005.
18. Soraya Abad-Mota and Eduardo Ruiz, "Experiments in information extraction," in *To appear in the Proceedings of the 2006 Information Resources Management Association International Conference*, Mehdi Khosrow-Pour, Ed. IRMA, 2006, Idea Group Publishing.
19. Amihai Motro, "Integrity = validity + completeness," *ACM Transactions on Database Systems*, vol. 14, no. 4, pp. 481–502, december 1989.
20. George Andrei Mihaila, *Publishing, Locating, and Querying Networked Information Sources*, Ph.D. thesis, University of Toronto, 2000.
21. Igor Rakov, "Quality of information in relational databases and its use for reconciling inconsistent answers in multidatabases. electronic document, citeseer.ist.psu.edu/133297.html," .

22. Soraya Abad-Mota, "Approximate query processing with summary tables in statistical databases," in *Proceedings of the 3rd International Conference on Extending Database Technology, Advances in Database Technology - EDBT '92. Vienna, Austria.* March 1992, number 580 in Lecture Notes in Computer Science, pp. 499–515, Springer-Verlag.
23. Mokrane Bouzeghoub and Verónika Peralta, "A framework for analysis of data freshness," in *IQIS '04: Proceedings of the 2004 international workshop on Information quality in information systems*, New York, NY, USA, 2004, pp. 59–67, ACM Press.
24. Amihai Motro and Igor Rakov, *Flexible query answering systems*, chapter Not all answers are equally good: estimating the quality of database answers, pp. 1–21, Kluwer Academic Publishers, Norwell, MA, USA, 1997.