Data Model for SIPAMER Prototype

Adela BÂRA, Anca ANDREESCU
Academy of Economic Studies, Bucharest, Romania,
bara.adela@ie.ase.ro, anca.andreescu@ie.ase.ro

The article presents the data model of the decision support systems in energy field and the future work plan for design development of cloud service information system for integration and knowledge management based in renewable energy. The research is part of SIPAMER project, financed by NASR agency.

Keywords: *decision support systems, XML, data integration, data model, renewable energy sources*

1 Introduction

Decision-making process, on the organization level, generally is very complex, as is revealed by the proposed definition in [1], where the process is seen as all phases, processes which determine their objectives and embedded subsystems using a complex methods and techniques in order to achieve more efficient the reasons which led the establishment of such organization.

Decision-making involves making decisions that are classified according to [1] in terms of: functional content (organizational, managerial and planning decisions), decision-making level (strategic, tactical and operational decisions), objectives achievement certainty (certain, uncertain and risk decisions).

As is described in the book [2], making decision involves the following elements: establishing targets to be achieved, finding more alternative action plans, including in economic constraining factors decisional map (resources, time, work), substantiating the decision on scientific level and its formulation in terms that can be understood and applied. Organization management involves different activities types, and therefore, requires different types of information and to efficiently manage and process this information and give them for analyses into a synthetic form as it requires a decision support systems type (DSS) developed on entire organization level.

Nationally, in the renewable power plants, at the moment, renewable resource management isn't supported by a decision support system that could enable efficient monitoring and analysis of energy resources from these sources. In Europe, there are several countries that developed decision support systems used for more effective management of renewable resources (e.g. Germany, Spain), but the building cost of these systems are high and specific national energy potential makes it impossible application of these methods in Romania. In terms of energy production forecasting systems, there are a series of software, e.g. production monitoring system based on the parameters of wind produced by GreenByte Sweden or forecasting services provided by the Fraunhofer Institute, Wind Energy and Energy System Technology. Problems related with the wide application of these solutions in Romania are related to the high costs related to forecasting services, but especially the errors recorded by these systems are caused primarily by the peculiarities of wind farms operation in areas with potential in Romania.

2. Heterogeneous systems and data sources

In installed renewable power plants there are installed different monitoring systems (e.g. SCADA/EMS) that use a variety of algorithms for the allocation and

management of equipment. From these plants originate information, forecasts and predictions with different errors from various types of applications and local devices. Based on the online transmission data system using EMS-SCADA, the TSO and NDC receive a series of notifications regarding the state of the elements and functioning of the generating groups. For obtaining the required information for the tactical and strategic decision making process, integration techniques and data processing, solutions for advanced analysis and data presentation in a friendly user interfaces are needed.

3. Data Transformation and Integration

Data integration consists in combining data from different sources, in order to give the user a unified view of these data [3]. This process is relevant to a variety of applications, including data warehousing, enterprise application integration and e-commerce applications. Also, data integration is a major challenge for situations like: a) advanced research in fields such as biology, ecosystems, environment and water management, where groups of researchers independently collect data and seek to work together; b) governments intention that different national agencies to be better coordinated; c) in a broader context, the retrieval and visualization of heterogeneous and disparate information on the Web.

Essentially, the purpose of IT solutions for data integration is to provide uniform access to a set of autonomous and heterogeneous data. In more detail, this means that data integration solutions must pursue at least the following [4]:

a) **Data query** - data integration solutions focus on integrating disparate data sources and updating them.

b) **Multiple data sources** - data integration is a challenge even when it involves a small number of data sources (e.g., less than ten), but these challenges are exacerbated when the number of data sources increases.

c) **Data heterogeneity** - in most cases, data integration involves data sources that have been independently created. Therefore, these data sources run on different systems, some of which being databases, other spreadsheet or even text files. Data sources may use different schemes and references to objects, even when modeling the same domain. Moreover, some data sources are completely structured (relational databases), while others may be unstructured or semi-structured (such as text or XML files). The heterogeneity of distributed data sources can be classified as: syntactic, semantic and schematic [5]. Syntactic heterogeneity is caused by the use of different models or languages (e.g. relational and XML). Schematic heterogeneity arises from different forms of organizing data (for example, application of different classification schema, different aggregation and generalization hierarchies). Semantic heterogeneity is caused by different interpretations of the data meaning. To achieve data integration and interoperability of systems, it is desirable that all these types of heterogeneity to be mediated and subsequently resolved.

d) **Data autonomy** - data sources do not necessarily belong to a single administrative unit or may be present at different organizational divisions. Therefore, certain aspects will be treated with particular care, such as: obtaining data access, data security and the management of any changes in data format.

Data transformation involves removing the existing errors in data, translating and integrating data / schema, as well as data filtering and aggregation. Usually, the most complex and difficult part of data integration is to transform data into a common format. Understanding the data to be combined and the structure of the results of integration requires both knowledge of data technical details and structure and their meaning within the

organization or organizations where data is used [6]. In figure 1 multiple data sources in different formats are converted into a set of integrated data. Many data transformations are performed simply by changing the technical format of data. But frequently, additional information is needed in order to show how data sources must be transformed from a set of values to another.

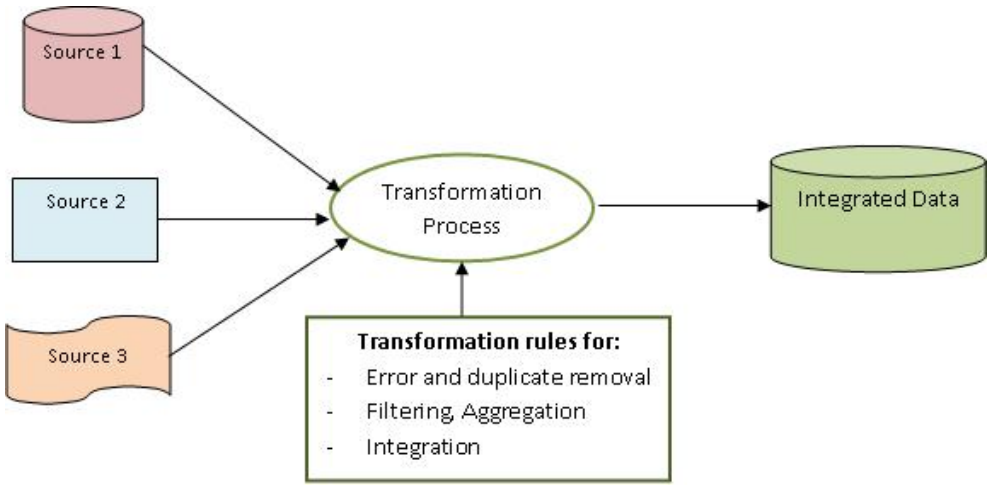

Fig.1 - Transforming data for integration (adapted after [6])

In data integration the way of transforming the local schemas (data sources) in the global schema (integrated data) determines the type of approach used for data integration. Four types of such approaches have been identified in [7], namely: a) global-as-view (GAV), where the transformation defines each component of the global schema in the form of queries on local schemas; b) local-as-view (LAV), where the transformation defines each component of each local schema as a set of queries of global schema components; c) global-local-as-view (GLAV) extends the LAV approach and represents a combination of the first two; d) both-as-view (BAV) applies a sequence of primitive transformations on global schema or local schemas.

When multiple data sources have to be integrated into a data warehouse, federated databases or other data processing systems, the need for data cleaning increases significantly. This is caused by the fact that sources often contain redundant data in different representations. Data quality problems are present within individual collections of data, such as files and databases, caused, for example, by misspellings during data entry, lack of information or other invalid data. Data cleansing deals with detecting and removing errors and incompatibilities from data in order to improve their quality [8].

4. Data model

The data model consists in a set of formal data description elements that can be used to generate and implement the database's structures and define the system's metadata. These elements have to offer a high level description of data in order to make possible the implementation of the business requirements within any type of database: relational, object-oriented or hierarchical. To ensure such flexibility and also an open standard for data description an eXtensible Markup Language (XML) extension can be used. The extension is known as XML Schema language, also known as XSD (XML Schema Definition) and it can be used to express a set of rules to which an XML document must conform in order to be considered 'valid' according to that schema. It provides a formal description that can contribute to create a

description that is precise and follows a prescribed set of rules in order to generate the set of entities and associated features of the data model. According to [9], the XSD specification acknowledges the influence of DTDs and other early XML schemas (DDML, SOX, XML-Data, and XDR) and it has adopted features from each of these proposals. The features offered in XSD that are not available in XML's native Document Type Definitions (DTDs) are namespace awareness, and datatypes, that is, the ability to define element and attribute content as containing values such as integers and dates rather than arbitrary text.

The growing need to have a common platform that can provide uniformity and improve interoperability between applications and systems of different organizations, led to the widespread acceptance of XML as a standard for data exchange. As a consequence, currently, most of the data exchanges are achieved through XML- based data representations.

Unlike HTML documents, whose purpose is to publish text content via a Web browser, XML documents are used to store data in a structured manner through the Web environment. Their content is structured in the form of user defined nested tags. Furthermore, it is possible to attach an additional document to the XML document in order to describe the labels and their structure. Thus, the labels are designed not to help visualize information, but to define its content and meaning.

XML advantages derived from its defining characteristics, offering a simple, flexible and self-describing data representation. The language flexibility is due to the fact that several alternative schemes can be efficiently combined using disjunction. Also, the self-description of XML lies in the fact that the instances have associated, along with the actual data, the data structure in the form of easily understandable labels. Therefore, XML data may be changed without the need for associated schemas. The simplicity and flexibility of XML has led to its widespread use in areas where ease of data exchange is a prerequisite, such as peer-to-peer applications (P2P), bioinformatics or semantic web [10]. On the other hand, these very properties, and in particular the flexibility of data structure and the possibility that each user to have its own schema (via DTD), are the ones which raise problems in data integration.

Through XML it is also provided a consistent set of guidelines, standards and approaches that can be used in the representation and management of the data in XML format [11], such as: query languages (XPath and XQuery) to query collections of XML documents and obtain adequate results, transformation facilities (XSLT) or facilities for the presentation of a document content in different formats (e.g. HTML, pdf or doc), description of data schema (XML schema and DTD) with the role of imposing integrity constraints, SQL extensions needed to manage, in the same time, (object oriented-) relational data and XML data (SQL / XML facilities) and index structures for efficient evaluation of queries.

By mid 90s, the growing need to have a common platform that can provide uniformity and improve interoperability between applications and systems of different organizations, led to the widespread acceptance of eXtensible Markup Language (XML) as a standard for data exchange. As a consequence, currently, most of the data exchanges are achieved through XML- based data representations.

XML data integration can bring a number of challenges highlighted in [11] and described briefly below.

Identifying and extracting data requires a special attention because not all data in XML format are accompanied by associated schemas. In fact, the main difference between XML and its

predecessor, Standard Generalized Markup Language (SGML), is precisely this relaxation as regards the requirement that each document to have an associated DTD that would establish rules for data structuring. Even if the transfer of data does not explicitly require knowledge of their structure, data integration requires this.

Data correspondence and mapping comes from the necessity, found in almost all data integration projects, to accomplish a correspondence between data components from different data sources.

Merging XML data / metadata occurs after finding mapping solutions and requires integration of data or metadata, depending on how the system operates (in terms of data or schemas).

Query processing requires that the resulting data to allow the performance of queries in an efficient manner, while conflict resolution occurs when XML data merging does not produce a valid result or scheme.

The main benefits of using the XML schema definition or XSD according to [12] are:

- offers an open platform to data modeling that support a wide range of programming languages and platforms;
- is object oriented and provides a well-specified framework for object oriented development;
- provides a wide range of pre-defined, built-in simple types (attributes and elements with text only content) that constrains the data model to conform to multiple platform requirements;
- offers a formal definition of the model that allows developers to take advantage of validation functionality abstracted into schema-aware XML parsers. Most information processing frameworks in use today now have schema-aware XML parsers built in and readily available for access by application developers.

To define a **schema** we can say that is an abstract collection of metadata, consisting of a set of **schema components**: element and attribute declarations and complex and simple type definitions. These components are usually created by processing a collection of **schema documents**, which contain the source language definitions of these components which are organized by namespace: all the named schema components belong to a target namespace, and the target namespace is a property of the schema document as a whole.

According to the definition in [12], a s**chema component** is the generic term for the building blocks that comprise the abstract data model of the schema. The *xs:schema* symbol defines the root element of a schema that contains representations for a collection of components (type definitions and element declarations), which have a common target namespace. There are 13 kinds of component in all as follows:

- Simple type definitions - a set of constraints of an element with no element children. It is defined as *xs:simpleType*;
- Complex type definitions - a set of constraints of an element that contains a sequence of items which conforms to a particular model group. It is defined as *xs:complexType*;
- Attribute declarations - an association between a name and a simple type definition, together with occurrence information and a default value. It is defined as *xs:attribute*;
- Element declarations - an association of a name with a type definition, either simple or complex, an default value and a set of identity-constraint definitions. It is defined as *xs:element*;
- Attribute group definitions - an association between a name and a set of attribute declarations, enabling re-use of the same set in several complex type definitions. It is defined as *xs:attributeGroup*;

- Identity-constraint definitions - an association between a name and one of several varieties of identity-constraint related to uniqueness and reference ;
- Model group definitions - an association between a name and a model group, enabling re-use of the same model group in several complex type definitions ;
- Notation declarations - an association between a name and an identifier for a notation ;
- Annotations - information for human and/or mechanical consumers ;
- Model groups - a constraint in the form of a grammar fragment that applies to lists of element information items. It consists of a list of particles, i.e. element declarations, wildcards and model groups ;
- Particles - a term in the grammar for element content, consisting of either an element declaration, a wildcard or a model group, together with occurrence constraints ;
- Wildcards - a special kind of particle which matches element and attribute information items dependent on their namespace name, independently of their local names ;
- Attribute Uses - an attribute declaration within a complex type definition is embedded within an attribute use, which specifies whether the declaration requires or merely allows its attribute, and whether it has a default or fixed value.

For the data types description XSD provides a set of 19 primitive data types (boolean, string, decimal, double, float, anyURI, QName, hexBinary, base64Binary, duration, date, time, dateTime, gYear, gYearMonth, gMonth, gMonthDay, gDay, and NOTATION). Also, new data types can be derived from these primitives by the following methods:

- restriction by reducing the set of permitted values ;
- list by allowing a sequence of values of the attributes ;
- union by allowing a choice of values from several types;

For a relational data representation the XSD schema components can be used as it is described in the following table:

Table 1 – Relational and XSD components

No	Relational element	XSD component
1	Table	Element, complex type
2	Attribute	Element, simple type
3	Constraint	Restriction, reference

For example, in order to specify the structure of the table *CENTRALA (centrala_id number primary key, denumire varchar(29), locatie varchar(30))*, the XSD schema can be written as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- CENTRALE -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="CENTRALE" targetNamespace="CENTRALE">
<xs:element name="CENTRALE">
<xs:complexType>
<xs:sequence>
```

```
<xs:element name="DENUMIRE">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:maxLength value="100"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="TIP">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:maxLength value="100"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="DATA_PUNERE_FUNCTIUNE">
<xs:simpleType>
<xs:restriction base="xs:date"> </xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="PUTERE_INSTALATA">
<xs:simpleType>
<xs:restriction base="xs:double">
<xs:minInclusive value="0"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="DATE_METEO_EOL">
<xs:complexType>
<xs:attribute name="ID_CENTRALA" type="xs:IDREF" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="DATE_METEO_PV">
<xs:complexType>
<xs:attribute name="ID_CENTRALA" type="xs:IDREF" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="TURBINE">
<xs:complexType>
<xs:attribute name="ID_CENTRALA" type="xs:IDREF" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="PV">
<xs:complexType>
<xs:attribute name="ID_CENTRALA" type="xs:IDREF" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="ID_CENTRALA" type="xs:ID" use="required"/>
</xs:complexType>
```

```
</xs:element>
</xs:schema>
```

In order to implement the model into a relational database, such as Oracle Database or Microsoft SQL Server, the XSD documents can be used to generate code for Data Definition Language (DDL). This is capability is referred to as *XML Data Binding*. This facility adds another advantage over the XML schema definition which can be considered as the best solution for data modeling (figure 2).

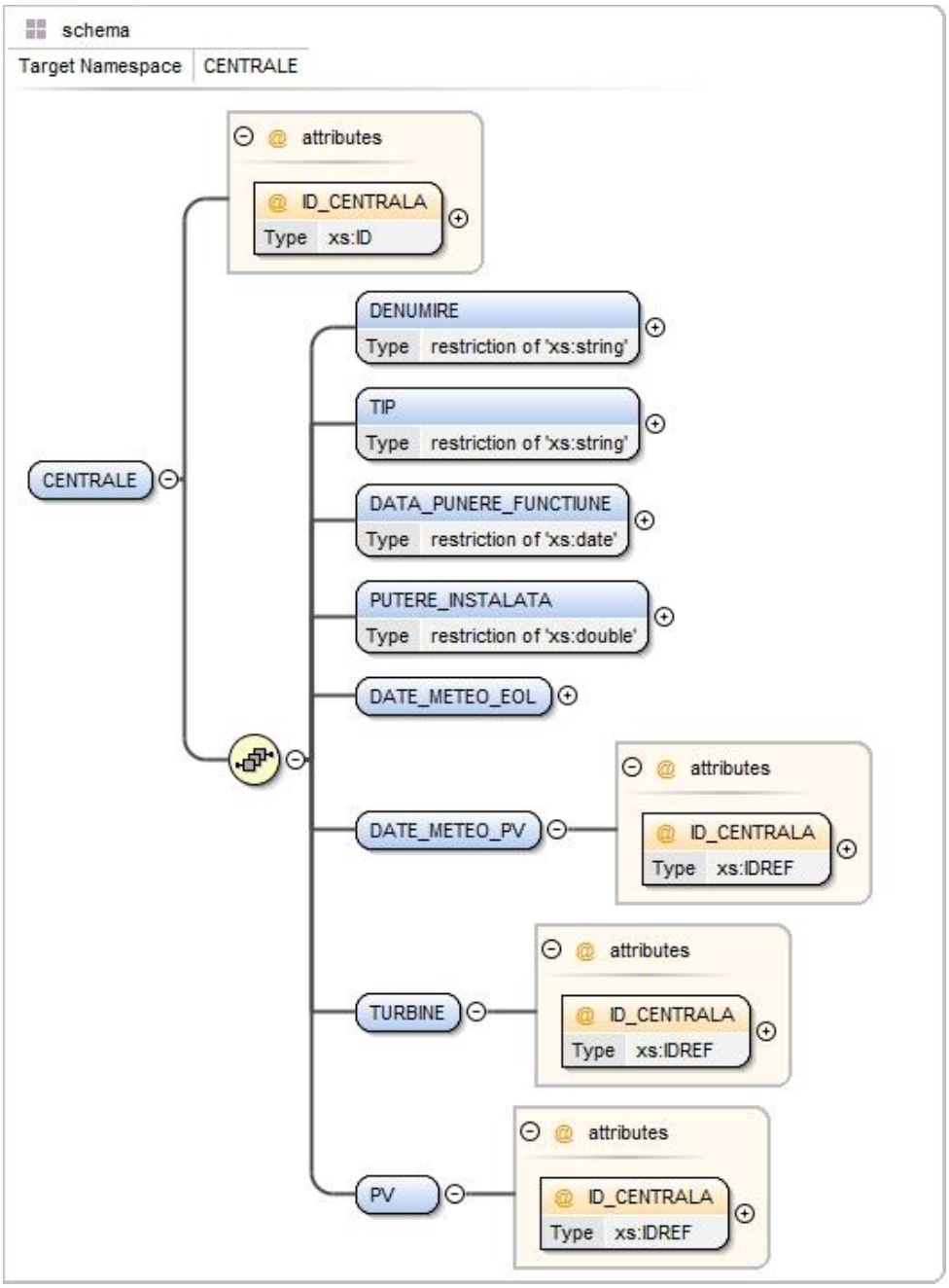

Fig.2 – The XSD schema for CENTRALE

After the data model design, the metadata is generated and table structures are created within the database (figure 3).

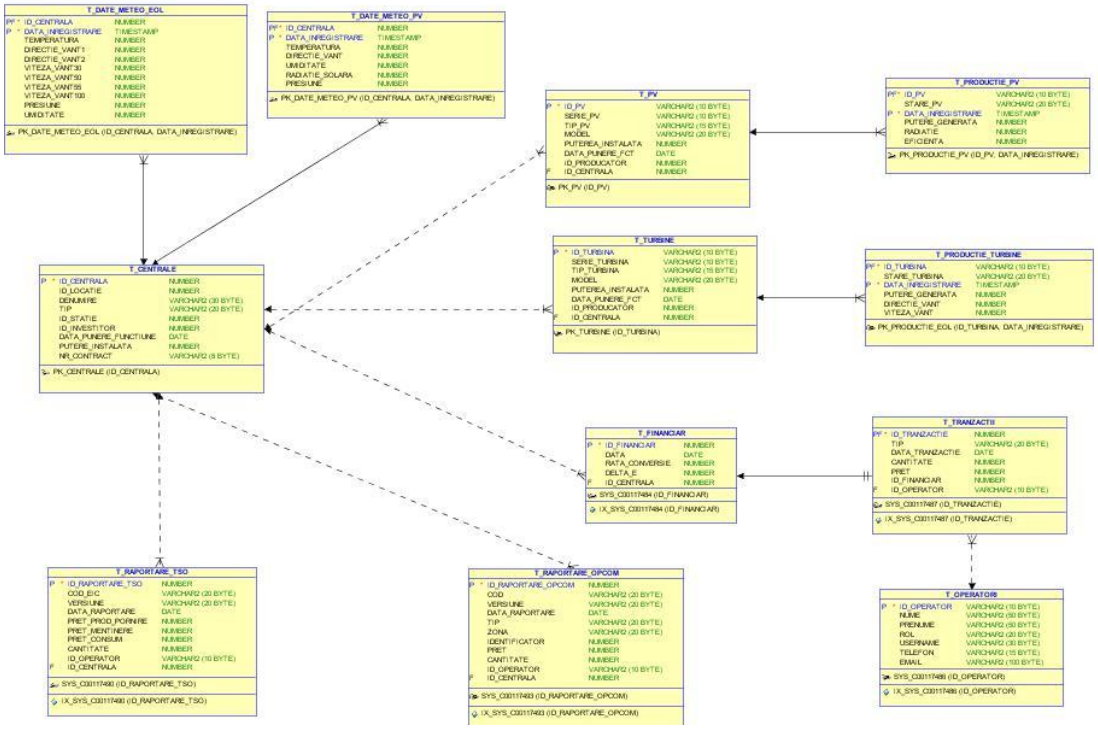

Fig. 3 – The database schema for RES producers

Conclusions

In this paper we presented a brief overview about the data model. The main objective is to identify the best suitable solution for data representation and organization in order to develop a flexible and scalable model. To ensure such flexibility and also an open standard for data description an XML extension can be used.

Acknowledgments. This paper presents results of the research project: *Intelligent System for prediction, analysis and monitoring of performance indicators of technological and business processes in the field of renewable energies (SIPAMER),* research project, PNII – PCCA 2013, code 0996, no. 49/2014 funded by NASR.

References

[1] O. Nicolaescu, I. Verboncu, *Fundamentele managementului organizatiei*, Tribuna Economică Publishing House, 2001, ISBN 973-934-890-4

[2] I. Lungu, A. Bara, *Sisteme informatice executive*, ASE Publishing House, 2007;

[3] Lenzerini, M. - *Data Integration: A Theoretical Perspective*. In Proceedings of the 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2002), Madison, Wisconsin, Iunie 2002, ACM;

[4] Doan, A., Halevy, A., Ives, Z. G. - *Principles of Data Integration*, Elsevier, 2012

[5] Bish, Y. A. - *Overcoming the semantic and other barriers to GIS interoperability*, International Journal of Geographical Information Science, 12(4), 1998

[6] Reeve, A. - *Managing Data in Motion: Data Integration Best Practice Techniques and Technologies*, Elsevier, 2013

[7] Zamboulis, L. - *XML Data Transformation and Integration — A Schema Transformation Approach*, PhD thesis, School of Computer Science & Information Systems,

Birkbeck College, University of London, 2009
[8] Rahm, E., Hong-Hai, D. - *Data Cleaning: Problems and Current Approaches*, IEEE Bulletin of the Technical Committee on Data Engineering, Volume 23, Number. 4, December 2000
[9] http://en.wikipedia.org/wiki/XML_Schema_%28W3C%29, July, 2010
[10] Huiping Cao, H., Qi, Y., Candan, S., Sapino, M. L.- *XML Data Integration: Schema Extraction and Mapping. Advanced Applications and Structures in XML Processing: Label Streams, Semantics Utilization and Data Query Technologies*, IGI Glogal, 2010
[11] Mesiti, M., Jiménez-Ruiz, M., Sanz, I., Berlanga-Llavori, R., Perlasca, P., Manset, G. - *XML-Based Approaches for the Integration of Heterogeneous Bio-Molecular Data*, Recent Advances in Biochemistry, Apple Academic Press, 2011
[12] Altova GmbH, *Whitepaper - Enterprise Data Modeling Using XML Schema*, Investigating an emerging paradigm using components of Altova's MissionKit™ for Software Architects, 2007
[13] The World Wide Web Consortium's XML Schema, Part 1: Structures Second Edition, 28 October 2004, available at http://www.w3.org/TR/xmlschema-1/#d0e504, July, 2010
[14] T. Ackerman, *Wind Power*, John Wiley & Sons, 2005
[15] T. Burton, D. Sharpe, *Wind Energy Handbook*, John Wiley & Sons, 2001

Adela BÂRA is Associate Professor at the Economic Informatics Department at the Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies of Bucharest. She has graduated the Faculty of Economic Cybernetics in 2002, holds a PhD diploma in Economics from 2007. She is the author of 7 books in the domain of economic informatics, over 40 published scientific papers and articles (among which over 20 articles are indexed in international databases, ISI proceedings, SCOPUS and 10 of them are ISI indexed). She participated as team member in 3 research projects and has gained as project manager two research contract, financed from national research programs. She is a member of INFOREC professional association. From May 2009, she is the director of the Oracle Excellence Centre in the university, responsible for the implementation of the Oracle Academy Initiative program. Domains of competence: Database systems, Data warehouses, OLAP and Business Intelligence, Executive Information Systems, Decision Support Systems, Data Mining.

Anca Ioana ANDREESCU is Associate Professor in Economic Informatics Department, Academy of Economic Studies of Bucharest. She published over 20 articles in journals and magazines in computer science, informatics and business management fields, over 30 papers presented at national and international conferences, symposiums and workshops and she was member in over twelve research projects. In January 2009, she finished the doctoral stage, the title of her PhD thesis being: The Development of Software Systems for Business Management. She is the author of one book and she is co-author of five books. She is a member of INFOREC professional association. Her scientific fields of competence and interest include: business analytics software, languages for data analysis, modelling languages, business rules, software systems methodologies and decision support systems.