INT J COMPUT COMMUN, ISSN 1841-9836
8(3):366-374, June, 2013.

Efficient Data Organisation in Distributed Computer Systems using Data Warehouse

S. Cosma, M. Văleanu, D. Cosma, G. Moldovan, D. Vasilescu

Smaranda Cosma
Faculty of Business, Babeş-Bolyai University,
Cluj-Napoca, Romania, Tel/fax: 0040-264-599170
E-mail: smaranda.cosma@tbs.ubbcluj.ro

Mădălina Văleanu, Dan Cosma, Dana Vasilescu
Faculty of Medicine, "Iuliu Haţieganu" University of Medicine and Pharmacy,
Cluj-Napoca, Romania, Tel/fax: 0040-264-207043
Email: drvasilescu@yahoo.com

Grigor Moldovan
Faculty of Mathematics and Computer Science,
Babeş-Bolyai University, Cluj-Napoca, Romania
Tel/fax: 0040-264-405300
E-mail: moldovan@cs.ubbcluj.ro

Abstract:
Databases represent a highly developed form of data organisation. The efficient use of databases by their beneficiaries is a permanent and stringent concern. The first part of the article presents a short development of the way data are organised and of the ways distributed databases can be optimised in computer networks. Then, two means for the efficient operation of distributed databases are shown. The last part of the paper evaluations of the architectures of some data warehouses (DW) and of their building are made. Using the data warehouse, a beneficiary can prepare in advance the required support to get reports and then make the right decisions in specific situations.
Keywords: Database, Distributed Databases, Relational Databases, Optimization Algorithms, Data Warehouse, Data Marts.

1 Introduction

Computer science has had a large contribution to the solution of issues from technical, scientific and economic fields, mainly with regard to the processing and managing of large volumes of data, such as those from economy, health and other fields. After the Internet has appeared, more and more distributed data processing systems have been designed and built. The computer-based data processing performance increased and large data volumes processing have expanded.

The *data* concept represents a basic concept always in the attention of the computer science experts. Data, whatever their nature, are identified by their *name*. The data as a notion has changed in meaning together with the development of the computer-related instruments and tools, especially due to the memorisation and access of the data found in various information supports. An efficient processing of data has always required some kind of data organisation, mainly when the information support used by computers is discussed. At the beginning, *isolated data* and grouped data stored in the internal memory of the computers were considered. Later on, economy-related applications have extended together with the opportunity to memorise data on external information supports; data could be organised under the form of *files*.

The processing of isolated or grouped data and that of files is achieved with the help of some software programs written in various automated programming languages, usually oriented to the application type (technical, scientific, economic etc.). The volume of processed data increased, together with the requirements of the users from various fields to receive reports and statements. Under such circumstances, there occurred the need to distinguish data from their subsequent processing, data being still stored on certain data supports. It is under this circumstance that the notion of *database* issued. Every database has a name, used whenever one wants to access data from the database in question. Data can be isolated or grouped or file or a database itself, as each and everyone has a name to be accessed with. The software applications have also diversified. Operational software appeared to manage certain data permanently; they are today called *operational data* (for example: in banking and medical systems) and are distinguished from *persistent data* which can also be stored in certain databases. To organise databases, various models have been used and consequently, various classifications of databases exist. The most known model of databases is the relational model. A relational database is a set of tables (that is relationships in the mathematical meaning of the word) with a relatively simple structure, where every table is defined with a set of attributes.

The Internet allowed the design and definition of *distributed databases.* Until today, we speak about more kinds of databases: relational databases, distributed databases, temporal databases, logic-based databases, object-based databases, object - relational databases or Web-based or XML-based databases. XML databases are defined by attributes to which documents interpreted as a sequence of characters are associated (that is, given as value). To create, manage and query databases, specific instruments have been achieved, and they are easy to use, powerful, and highly performing as they need to give answers to extremely complex situations and questions. One of these instruments to create and query relational databases is represented by the *SQL* language with its different extensions. The new generations of databases, mainly those based on the Web or XML, have the tendency of being NoSQL [14].

At this moment, databases constitute a high level form of data organisation. Relational databases are widely spread as they have a simple representation module of the component data under the shape of tables (that is relationships in the mathematical meaning of the word). If for $\forall$ i $\in$ *{1, ... ,m}* , A_i represent attributes of an instance of an r database, and D_i is the domain of attribute A_i , then r $\subseteq D_1 \times D_2 \times ... \times D_m$. We agree to note with R$(A_1, A_2, ..., A_m)$ or R=$(A_1, A_2, ..., A_m)$, the scheme of r database. When the Internet was developed, the move to distributed databases was a natural one. Various means to extend and particularise databases were developed, and the majority of them started from relational databases. The hardware infrastructure also became more diverse when more and more high performance computer networks were installed. Different systems such as *Database Management Systems (DBMS)* are known today, built mainly on the relational model, for instance: DB2, Ingres, Informix, SQL Server, Oracle. Now the concern has moved to the architecture of databases, so that the use of databases can lead to more performance, mainly in the operation of systems of databases (with main components: data, hardware, software and users) to *support decision-making processes.* The theoretical and practical research is oriented more to the internal structures of organising databases. The ANSI/SPARC architecture is reasonable for databases systems regarding three levels an internal level, a conceptual level and an external level (views of individual users).

2 Distributed databases

Distributed databases are part of a distributed system with a well-defined purpose. As a principle, a distributed database should be perceived by a user similarly to a non-distributed database. Processing data from distributed databases should consider the presence of a communications network which supposes new circumstances to be taken into account [2]. For this purpose, the Databases Management Systems have been extended to the distributed ones (DDBMS).

One of the most important issues, with respect to distributed databases, lies in minimising the number and volume of messages used in the network. That is why is significant to redistribute databases in the nodes of the communications network to get an efficient, if not optimal exploitation of the system. The query optimising process which requires distribution needs passing through two steps, a global and a local step. Also important are issues of propagating updates, recovery and concurrency control. Software making use of distributed databases are associated, in general, to a connecting graph defining the communications network. Be the network nodes *X={1,2,...,n}*, and the links among the nodes, respectively its arches, noted *U*, i.e. the connecting graph *G=(X,U)*. Let us note with B_i the database in node i, and with B the distributed database in all the nodes of the network considered, hence:

$$B = \{B_1, B_2, ..., B_n\}$$

Relational databases are defined by the relational schemes $B_i = (A_1^i, A_2^i, ..., A_m^i)$, where A_j^i, $j = \overline{(1,m)}$, ∀ i ∈ *{1,2, ... ,n}* represent attributes.

The following refers to an operational distributed information system managing the databases in the nodes of the communications network defined by graph *G=(X,U)*. Through the DBMSs found in the nodes, computers are required services, respectively the databases in the network nodes are queried. Some of the nodes are under heavy stress, others are not so much under stress. To increase the efficiency of the information system a migration of the databases or of parts of them from the nodes under heavy stress to less stressful nodes could be useful. A balanced network nodes loading is a significant issue. The solving of the issue requires every node to send out and receive messages carrying information with respect to the number of queries in every node and to the number of nodes (databases) used. The number of these messages should be as small as possible. Every node i will contain information regarding the system loading. Let us suppose that for every node, the loading or request level or the number of queries is measured on a scale from *1*, that is *weak* load, to p, that is *heavy* load; we identify these pieces of information in node i with elements of the set *Inc*^{i}*=(1,2, ... ,p)* (*Inc*^{i} -node *i loading*). If *p=3*, we get the loadings: *weak, normal, heavy*, which some authors consider as sufficient.

Message migration together with databases (resources) always begins from the heavy loaded nodes towards the weak loaded nodes. A small number of messages exchange will occur in the nodes of the system under question until it reaches the desired equilibrium.

To make the distributed database exploitation more efficient, a balanced loading of the computer network should be provided, with some dynamic balance algorithms. The algorithm proposed by Suen and Wong [12] for distributed systems shall be adapted to the present situation.

In every node i, a piece of information regarding the nodes to which we *send out* query messages for the databases B_j in the nodes shall be preserved; the set of nodes is noted S^i . In the same node i, B_i database query messages *will be received* from a set of nodes noted R^i . It is obvious that $S^i \subseteq X, R^i \subseteq X$.

The two sets associated to node i, respectively S^i, R^i must satisfy some properties (constraints) to provide the operation of the efficiency algorithm of the database operated upon. These constraints, relative to S^i, are:

- Any of two distinct nodes $i, j \in X, i \neq j$ have at least one node to which they send messages, that is

$$S^i \cap S^j \neq \varnothing$$

- Whatever node i, this node is allowed to send a message to it and to receive a message from it, i.e. $i \in S^i \wedge i \in R^i$

- There is a property of equal responsibility for any $i \in X, i.e. |\{j/i \in S^j\}| = k$

- There is a property of equal work for any $i \in X$, i.e. $|S^i| = k$
- There is an optimal condition *N= k(k-1) + 1.*

The last two properties specify that every node i receives and sends out k messages to have an equal responsibility and work amount, so that the activities to be performed by the network nodes are in perfect balance.

The properties of R^i are written in the same manner, replacing S^i with R^i in the list above.

We will now present another approach regarding the efficient operation of a database distributed in the nodes of a computer network [10]. We consider that in graph *G=(X,U)*, we have a zoning made up of M components, if on the set of nodes X we have made a partition of M components noted $Z_i \subset X, Z_i \neq \varnothing, i = \overline{1,M}$ so that:

$$\bigcup_{i=1}^{M} Z_i = X \ \ and \ \ Z_i \cap Z_j = \varnothing, \forall i \neq j, i,j = \overline{1,M}.$$

We note $G_i = (Z_i, U_i),\ where\ U_i = \{u \in U | u \in Z_i \times Z_i\}, \forall i = \overline{1,M}.$

The purpose of this distribution of the databases on the M zones lies in getting a certain autonomy, as well as some constraints, hence a certain level of efficiency for their operation. We have in view the following properties:

- *equity;* following the loading in every node i, for a defined time interval, are determined statistically some weights h_i of services request offered in the nodes in question (for instance, a percentage of the number of queries of basis B_i, percentage of the number of messages received by node i, percentage of the use time for node i etc.). Hence, $\sum_{i=1}^{n} h_i = 1$.

Absolute equity for zone loading to fulfil the tasks will occur if the volume of the tasks for every zone will be $\frac{1}{M}$, i.e. $\sum_{j \in Z_i} h_j = \frac{1}{M}, i = \overline{1,M}$. A maximum deviation from the ideal value $\frac{1}{M}$ is accepted, though probably it will be difficult to meet, equal to α, $0 \leq \alpha \leq 1$. Hence, equity will exist if:

$$\left| \sum\nolimits_{i \in Z_j} h_i - \frac{1}{M} \right| \leq \frac{\alpha}{M}, \ \ 0 \leq \alpha < 1, \ \ 1 \leq j \leq M.$$

- *contiguity*; zone $G_i, i \in$ {1,2,...,M} is contiguous, if and only if this graph is connecting (there is a chain between any two of its nodes).
- *compactity*; zone G_i is compact if the shortest chain between any two nodes a and b from Z_j , $1 \leq j \leq M$, is smaller than or equal to a predefined constant, named exclusion distance.
- *exclusion of some enclaves;* an enclave is a node or a set of nodes that cannot form a zone if the criteria mentioned earlier are fulfilled.

We can achieve such zones with a not very complex algorithm and in the zones we can efficiently exploit the distributed database [11], [13].

3 Data Warehouse architecture and their efficiency

Data preservation without processing cannot be a purpose in itself. The applications of databases are numerous and for many domains. There are extremely many situations when databases are created as decision-making supports. Such situations can be encountered mainly when data refer to economy, though similar cases are applicable for health-related data or other fields. The volume of stored data on data supports has increased and continues to increase. During this development of information technology, the need to *restructure* data inside such supports for an efficient use has come up. The data collection sources are many-sided and their structure is heterogeneous. This is due to the fact that the types of databases can vary and differ, as mentioned earlier.

The information systems designed many years ago showed to be unable to cope with such situations. Consequently, the reconsideration of the manner of data collection in a joint place and the processing of the mentioned data as decision making supports became a necessity. Such applications mainly appeared in the field of economics, as they could be used as two solutions to solve the difficulty. One of the solutions supposes the increase of computer technical performance, the other solution can be applied immediately and resorts to developing proper software for the purposes envisaged. The second direction underwent an avalanche of research and articles published in this domain.

The new concept regarding the management of large volumes of data as decision-making supports is represented by the *Data Warehouse* introduced by W.H. Inmon in 1992, which, in fact, is a special database, as also shown well in [3]. The simple idea consists in building warehouses as an intermediate form among various data sources, usually operational databases and selecting/extracting some subbases of special data of reasonable size from these warehouses. These special data subbases should serve as support for informing decision makers or for writing reports, research documents or other, in relation with the processed data. The decision in a company can be related to sales, customers, suppliers, finances, costs, profit, human resources, etc. [4]. In a hospital networks, they can refer to patients, medical staff, resources, medicines required, etc. This requires the division of the functional space of a data warehouse in three distinct areas: the data sources zone, in general, operational databases, but also other types of data received by means of other software, then the zone comprising data in the data warehouse, and the third area, the special data marts zone, which are used for analyses, reports drawing or forming the so-called data mining.

Data sources can be of different nature, different formats or different types. All the data will be subjected to significant preparation process before reaching the data warehouse. During the preparation process, through a well-defined selection - considering the objectives established for the use of the information system in question - there takes place *Data Cleaning, transformation and unification.* After *their loading*, they need to be refreshed and at each refreshing process, integrity, consistency and other constraints need to be provided to maintain a reasonable volume of data which are necessary to make a decision. We have mentioned earlier some of the features a data warehouse should possess. According to W.H. Inmon, a data warehouse is defined by enumerating its properties, that is:

Definition [8]: *Data warehouses are non-volatile data collections, oriented to the subject, integrated, variable in time and supporting the managerial decision making process.*

The creation of a *data warehouse* passes through the following:

{Data sours, ... , Data sours} $\rightarrow$ ECTL $\rightarrow$ Data Warehouse;

where *Data sours* represents operational databases or even data sources of other nature. Through a process of *Extraction, Cleaning, Transformation and Loading*, in brief *ECTL* (Extraction, Cleaning, Transformation and Loading) - a data base of uniform structure is developed, i.e. a *Data Warehouse* (DW). A *Data Warehouse* whose data cannot be modified (they can, however, be deleted) is formed of the following components: *(Heavily, slightly) Aggregated Data, Detailed data and Meta data.* A warehouse can be accessed only by repeated reading and can be used (distributed) to form some *views* representing data mart s to be used by decision makers.

Any standard architecture leading to the use of a data warehouse to make a decision in an organisation (for example, a company or a hospital) has the shape given in Fig.1.

When the data warehouse is distributed, then the middle part of the drawing above takes,

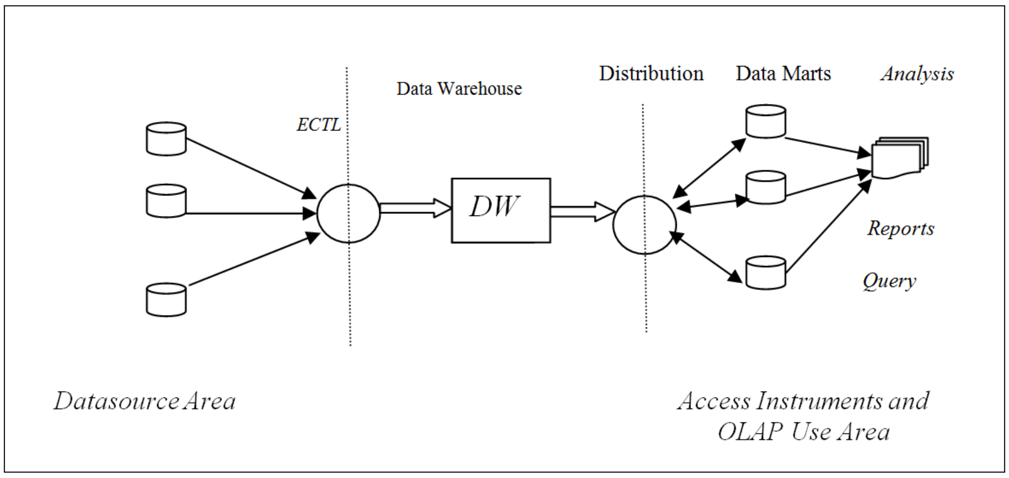
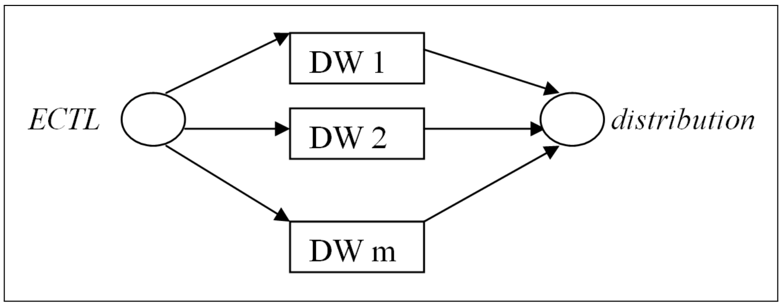


Figure 1: Architecture Standard Data Warehouse

in general, the shape below:



The loading of distributed data warehouses $DWi, i \in \{1,2, \dots ,m\}$ uses any of the data sources to which some aggregation processes are applied, so that in *Y*, data will have a unitary structure. Let us note X the set of data sources, and Y the set of Data Warehouses *DW i*. The two sets have a application multivalued $f : X \to Y$.

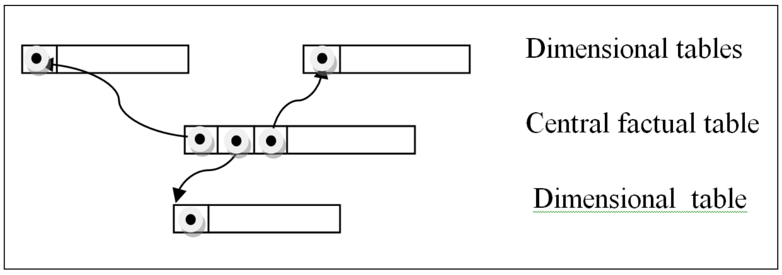
For the destination area, in the drawing above, we can notice that the often mentioned data marts are highlighted. An analogous correspondence g exists between the Data warehouses set Y and the set Z of *Data Marts*, i.e. $g : Y \to Z$. It is possible to conceptually design directly a data mart from a source, such as *XML* as specified in [2], [7]. Though the term *data marts* is still disputed, we have chosen the definition below to be suitable for them:

Definition [2]: *We call a Data Mart a* ***specialised*** *data warehouse, oriented to the subject, including* ***volatile*** *integrated data, variable in time and being the subsupport for managerial decisions.*

In the destination area, specific technologies and instruments of processing data should be present. *OLAP (On-line Analytical Processing)* is the best known technology for the aggregation, access and use of data from a data warehouse, for the purpose of decision making. The term *OLAP* was proposed by Codd in 1993. *OLAP* operations with regard to multidimensional data (data cubes) are well-known. Let us mention the most significant of theme: *Roll-up, Drill-down, Slice, Slice-and-dice, Pivoting, Drill-across.*

In Fig. 1, a standard architecture of a Data Warehouse is given. The architectures are designed for organisations, having a well-defined purpose, and consequently, a specificity. It is always important that the architecture is less costly and more efficient for its users, possessing certain hardware and software resources, while data, in general, are heterogeneous. Many authors (such as Jenning in 2011 [9]) specify that "the best solution for one enterprise is not necessarily the best for another". Optimal solutions are only for concrete situation. Anyway, the information systems designed for organisations for decision making purposes, based on Data Warehouse , have extended a lot. In Poland, for instance, as mentioned in [1] DW architectures are used in 20%, respectively 23% of cases for Data marts and only 7% Federal Architectures, from all the systems created in 2011.

The issue of preserving large volumes of data remains a constraint today, too. We will always want to use a smaller memory space. The model of *multidimensional tables* or *hypercubes* for data provides solutions to diminish the memory space. A *dimensional scheme* is made up of a factual table and more *dimensional tables* with connections established as keys. Such a multidimensional scheme can also be called *star-shaped scheme.* A theoretical example is illustrated in the drawing below.



In the factual table given, at the beginning three attributes, in fact, keys, representing links to the three dimensional tables are found. The dimensional tables also contain keys under the same name of attribute, and the references are indicated by the three arrows. In the factual table, after the attributes-keys, other attributes are listed. The dimensional tables also contain a list of attributes after the key attribute.

In principle, when designing a DW, to produce factual schemes the following successive steps need to be followed: *building the attributes tree, cleaning and completing the attributes tree, specifying dimensions, measurements and hierarchies* [5]. When attributes are known from the operational databases schemes (dimensions, measurements, hierarchies), they are defined together with the final users. The conceptual factual scheme must be designed, definitely, before implementation. The effective scheme of a DW shall be a sum up of DM (Data Marts), data sources and profiles required by users. A DW design methodology such as the one based on the *Dimensional Fact Model* (DFM) should be useful as it proposes a formalisation of the diagrams E/R in a graph [5]. Using these elements, as the example above shows, one can built particular graphs, respectively *quasi-trees*, whose roots are facts; Golfarelli and Rizz acted in this manner in [5]. The majority of a DW elements, in our opinion, have a dynamic character and, in consequence, they should be considered cybernetic systems, where returns to the system under the form of feed-backs are admitted.

In order to select some options (actions) defining the conceptual schemes of data warehouses (DWs) and able to provide short time answers to queries made by users, some efficiency indicators can be used. Thus, such an evaluation used successfully in various domains is given below.

Be a set of options $A = \{a_1, a_2, ..., a_n\}$, their selection being made on the basis of well-defined criteria, forming set $C = \{c_1, c_2 ..., c_m\}$. Let us suppose that after evaluations, the level of influence of criterion c_i in the selection of option a_j with the value $q_i^{a_j}$, from a scale of values is known.

If we consider options a and b, in the mentioned order, from set A, we convene, to note with $p_i(a, b)$ the preference for option a versus b, a preference determined by criterion c_i. This degree can be expressed as a percentage. If, for the two options a and b we take into account all the criteria c_i to select option a related to b, the result of the evaluation noted *r(a,b)* will be:

$$r(a,b) = \sum_{i=1}^{m} p_i(a,b) \cdot q_i^a$$

As every option a is compared to *n-1* options , a positive score for option a can be defined as $S^+(a)$ as well as a negative sc ore *S(a)*, as follows:

$$S^+(a) = \frac{1}{n-1} \sum_{x \in A} r(a,x) \quad ; \quad S^-(a) = \frac{1}{n-1} \sum_{x \in A} r(x,a)$$

The positive score of option a represents the maximum global influence of option a versus the rest of options, while the negative score of option a represents its minimal global influence. In this way, we will associate a *full degree* to each option a from set A as follows:

$$S(a) = S^+(a) - S^-(a)$$

Based on these indicators, we can define efficient conceptual schemes for DW and MS. For Data Marts constitution, option groupings following given criteria can also be useful.

4 Conclusions

It is obvious that high performance DW architectures are desired, as they could be capable to give answers at the shortest time and expense to various queries found in an organisation. As this paper shows, this optimisation can be performed only in exact and concrete environments, through successive improvements and applying various methods, among which those presented here. A high performance DW architecture depends on the computer network involved, on the concept schemes defined, including multidimensional tables, respectively hypercubes, on the quality of the software used as well as the *OLAP* processing.

Bibliography

[1] M. Alsqour, K. Matouk, M.L. Owoc, A survey of data warehouse arhitectures - preliminary results, *Proc.of the Federated Conference on Computer Science and Information Systems*, 1121-1126, 2011.

[2] C.J. Date, *On Introduction to Database Systems*, 8th Edition. Pearson Education Inc, Addison Wesley Higher Education, 2004.

[3] F.G. Filip, Decizie asistata de calculator. Concepte, metode si tehnici pentru deciziile centrate pe analiza datelor, *Rev. Informatica Economica*, 4(16):8-22, 2000.

[4] D. Fusaru, Z.Gherasim, I. Lungu, A. Bara, Tehnici si arhitecturi pentru micsorarea timpului de raspuns in sistemele cu depozite de date (Data Warehouse), *Analele Universitatii Spiru Haret. Seria Economie*, Ed. Fundatia Romania de maine, Bucuresti, 4(4):419-426, 2004.

[5] M. Golfarelli, S. Rizz: A Methodological Framework for Data Warehouse Design. In: *Proceedings ACM First International Workshop on Data Warehousing and OLAP (DOLAP)*, Washington, pp. 3-9, 1998.

[6] M. Golfarelli, S. Rizzi, E. Turricchia, Modern Software Engineering Methodologies Meet Data Warehouse Design: 4WD. In *13th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2011)*, pp. 66-79, Touluse, France, 2011.

[7] M. Golfarelli, S. Rizzi, B. Vrdoljak, Data warehouse design from XML sources, *Proceedings ACM Third International Workshop on Data Warehousing and OLAP (DOLAP'01)*, Atlanta, pp. 40-47, 2001.

[8] W.H. Inmon, *Building the Data Warehouse*, Third Edition. Pub. John Wiley&Sons, Inc. USA, 2002

[9] C.Jennings, L. Tristan, The Perils of Over Complicating Data Warehouses How Avoiding Unnecessary Complexity Con Provide Satisfaction and Savings, *Business Intelligence Journal*, 16(2):39-43, 2011.

[10] G. Moldovan, M. Valeanu, The Performance Optimization for Date Redistributing System in Computer Network, *INT J COMPUT COMMUN*, Vol.1, Supplementary Issue, 1(S):470-473, 2008.

[11] G. Moldovan, O problemă de redistribuire a serviciilor într-un sistem partiţionat în zone distincte după anumite criterii. "Babeş-Bolyai" University Cluj-Napoca, Fac. Math. and Computer Sci., *Res. Sem. Preprint*, no.5, 5-8, 1993.

[12] T.T.Y. Suen, J.S.K. Wong, Efficient Task Migration Algoritm for Distributed Systems, *IEEE Transactions on Parallel and Distributed Systems*, 3(4):488-499, July 1992.

[13] M. Valeanu, S. Cosma, D. Cosma, G. Moldovan, D. Vasilescu, Optimization for Date Redistributed System with Applications, *INT J COMPUT COMMUN*, 4(2):156-161, 2009.

[14] EG. Ularu, F. Puican, Noua generatie de baze de date NoSQL. *Rev. Romana de Informatica si Automatica*, Vol.22, no.4, 2012