THE BUCHAREST UNIVERSITY OF ECONOMIC STUDIES

DATABASE SYSTEMS JOURNAL

Vol. V, Issue 4/2014

LISTED IN

RePEc, EBSCO, DOAJ, Open J-Gate,
Cabell's Directories of Publishing Opportunities,
Index Copernicus, Google Scholar,
Directory of Science, Cite Factor,
Electronic Journals Library

BUSINESS INTELLIGENCE

ERP

DATA MINING

DATA WAREHOUSE

DATABASE

ISSN: 2069 – 3230
dbjournal.ro

Database Systems Journal BOARD

Director
Prof. Ion Lungu, PhD, University of Economic Studies, Bucharest, Romania

Editors-in-Chief
Prof. Adela Bara, PhD, University of Economic Studies, Bucharest, Romania
Prof. Marinela Mircea, PhD, University of Economic Studies, Bucharest, Romania

Secretaries
Lect. Iuliana Botha, PhD, University of Economic Studies, Bucharest, Romania
Lect. Anda Velicanu, PhD, University of Economic Studies, Bucharest, Romania

Editorial Board
Prof. Ioan Andone, PhD, A.I.Cuza University, Iasi, Romania
Prof. Anca Andreescu, PhD, University of Economic Studies, Bucharest, Romania
Prof. Emil Burtescu, PhD, University of Pitesti, Pitesti, Romania
Joshua Cooper, PhD, Hildebrand Technology Ltd., UK
Prof. Marian Dardala, PhD, University of Economic Studies, Bucharest, Romania
Prof. Dorel Dusmanescu, PhD, Petrol and Gas University, Ploiesti, Romania
Prof. Marin Fotache, PhD, A.I.Cuza University, Iasi, Romania
Dan Garlasu, PhD, Oracle Romania
Prof. Marius Guran, PhD, University Politehnica of Bucharest, Bucharest, Romania
Lect. Ticiano Costa Jordão, PhD-C, University of Pardubice, Pardubice, Czech Republic
Prof. Brijender Kahanwal, PhD, Galaxy Global Imperial Technical Campus, Ambala, India
Prof. Dimitri Konstantas, PhD, University of Geneva, Geneva, Switzerland
Prof. Hitesh Kumar Sharma, PhD, University of Petroleum and Energy Studies, India
Prof. Mihaela I.Muntean, PhD, West University, Timisoara, Romania
Prof. Stefan Nithchi, PhD, Babes-Bolyai University, Cluj-Napoca, Romania
Prof. Corina Paraschiv, PhD, University of Paris Descartes, Paris, France
Davian Popescu, PhD, Milan, Italy
Prof. Gheorghe Sabau, PhD, University of Economic Studies, Bucharest, Romania
Prof. Nazaraf Shah, PhD, Coventry University, Coventry, UK
Prof. Ion Smeureanu, PhD, University of Economic Studies, Bucharest, Romania
Prof. Traian Surcel, PhD, University of Economic Studies, Bucharest, Romania
Prof. Ilie Tamas, PhD, University of Economic Studies, Bucharest, Romania
Silviu Teodoru, PhD, Oracle Romania
Prof. Dumitru Todoroi, PhD, Academy of Economic Studies, Chisinau, Republic of Moldova
Prof. Manole Velicanu, PhD, University of Economic Studies, Bucharest, Romania
Prof. Robert Wrembel, PhD, University of Technology, Poznan, Poland

Contact
Calea Dorobanţilor, no. 15-17, room 2017, Bucharest, Romania
Web: http://dbjournal.ro
E-mail: editordbjournal@gmail.com; editor@dbjournal.ro

CONTENTS

'Shared-Nothing' Cloud Data Warehouse Architecture

Janina POPEANGĂ
University of Economic Studies, Bucharest, Romania
janina.popeanga@yahoo.com

Energy management systems from Romania do not have the capabilities of energy specific management due to lack of technology for real-time monitoring. As was the case in many other countries, the advent of smart metering technology will increase the level of energy data significantly. Therefore, the purpose of this paper is to present solutions that need to be taken to solve problems linked with the increasing amount of data recorded by sensors.
For a better demonstration of theoretical elements exposed, we considered a data warehouse specific to utility companies.
Section 2 of this article defines the three widely used parallel data warehouse architectures, while in Section 3 we clarify what architecture is suited to develop a data warehouse in the cloud. In the last part we transposed our tables in a "shared-nothing" architecture, trying to analyze queries performance.
Keywords: *Shared-nothing architecture, Data Warehouse, Replication, Distribution, Cloud*

1 Introduction

The amount of data being collected by utility companies has increased enormously in many countries with the advent of the smart metering technology. Before smart grids, utilities had collected from their customers data regarding their consumption on a monthly frequency. With the smart metering technology readings are taken at shorter intervals - every few seconds, so the increase is very significant.
The increasing volume of data and the variety of new data sources (devices and sensors) have created new challenges for utilities industry in terms of how to organize the amount of data from different sources in order to ease the data analysis.
Data warehouse is a centralized storage method of organizing data from different sources, undergo the process of extraction, transformation and loading, and of aggregate storing on hierarchical levels, data which are used in various complex processing and dynamic analysis. A data warehouse is hosted on an enterprise mainframe server or in the cloud.
Unlike operational systems, data structures in a data warehouse are optimized for dynamic queries and fast analysis. Data are historical and are updated at regular intervals, depending on the reporting requirements.
However, faced with the growing weight of explosive volumes of data and the expansive variety of data types, the capacity of a centralized data warehouse seems too limited.
Consider a data warehouse designated to customers application which permits them visualizing their consumption data over a period of time and several analysis on those data. (**Fig. 1**)
Consider that the data warehouse that we mentioned contains 3 dimensions and one fact table:
User – table which contains the main user information, like: UserID, User Name, Street, City, District, Region, Phone, Email, Income Level, House Area (square meters), Persons (number of persons in the house), El_Heaters (number of electric heaters in the home), HeatingH (heating time), Refrigerators (number of refrigerators), AirCond (number of reverse cycle air conditioners), HWLights (number of high-watt lights), HinHome (number of hours spent at home during the day), Requip (renewable energy equipment), Password.

Sensors – table which contains data about sensors: SensID (sensor ID), SensType (sensor type), Status (Down, Down (Partial), Down (Acknowledged), Warning, Up, Paused, Unusual, Unknown), Installation date, Last Revision.
Time – a dimension on data warehouse, since utilities will frequently want to aggregate about it. Full_Date, Time (hh:mm:ss), Hour, Day, Month, Quarter, Year, Day of Week, Holiday and Weekend flags are implemented in application by using the dimensional attributes.
In the **EnergyConsumption** fact table, energy consumption is measured below the level of calendar day, down to hour or minute or even seconds.

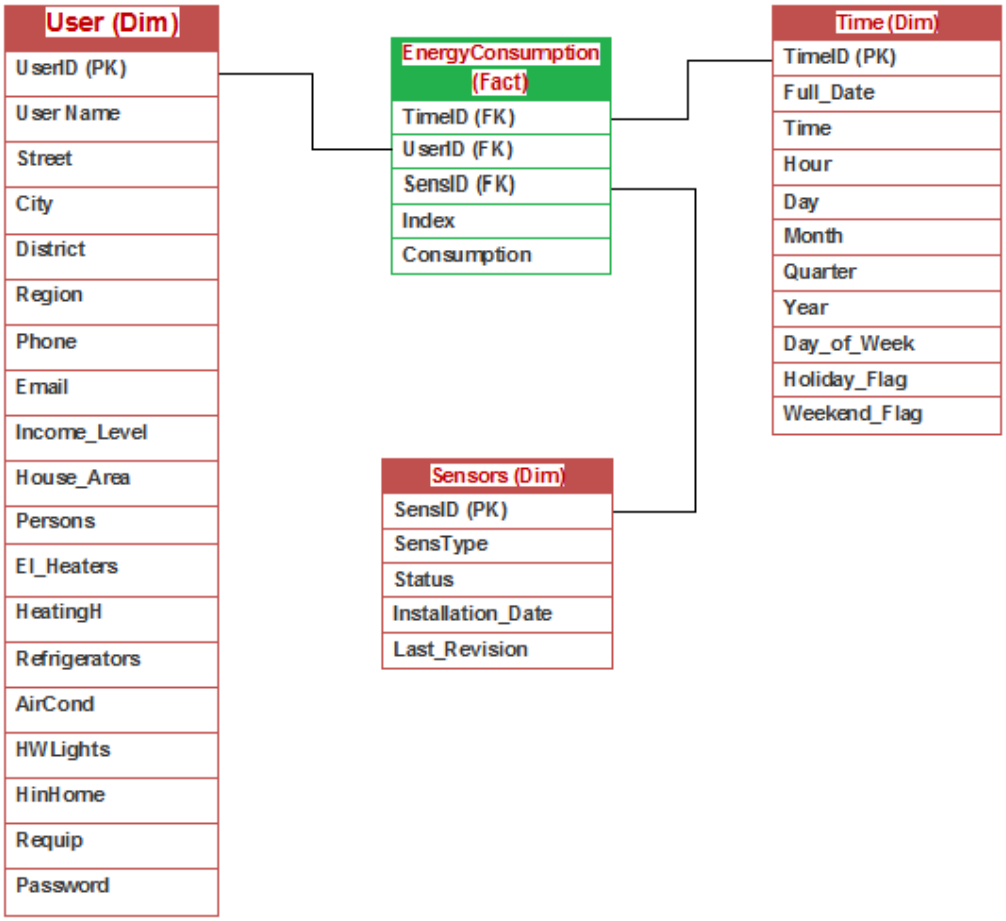


Fig. 1. Considered Energy Data Warehouse

The volume of data collected from consumers is increasing as time goes by, the number of concurrent queries is also rising.
The scalability issue becomes a huge challenge for centralized data warehouses. The solution addressing this dare is to distribute the large-scaled dataset and calculate the queries in parallel.
This paper focuses on the importance of solving these issues when creating a data warehouse in the cloud. The rest of the paper is structured as follows:

- Section 2 defines parallel data warehouse arhitechtures
- Section 3 discusses what should utilities look for in their cloud data warehouse
- Section 4 outlines how our tables should be split up across the nodes using a shared-nothing architecture – case study Microsoft SQL Server Parallel Data Warehouse (PDW)

Finally, we conclude this article in Section 5.

2 Better performance through Parallelism

Three widely used parallel hardware architectures for data warehousing exist, including shared-memory, shared-disk and shared-nothing. Consider also three basic elements in a parallel system: the central processing unit (CPU), the storage device (S) and memory (M).

The **Shared Memory (Shared Everything)** architecture is a system architecture where all existing CPUs share a global memory (M) and a single collection of disks (S). **(Fig. 2)**

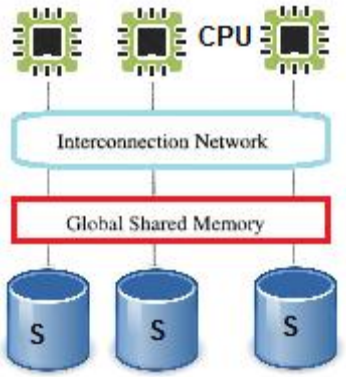


Fig. 2. Shared Memory

Only one Database Management System (DBMS) is present and can be executed in multiple processes or threads, in order to utilize all processors. [1]
Since there is a single memory, the lock manager and buffer poll are both stored there and this gives the chance to be easily accessed by all the CPUs.
There are two variations of Shared-Everything architecture [2]:

- The *symmetric multiprocessing architecture (SMP)*, where all the processors share a single pool of memory for read–write access concurrently and uniformly without latency.
- The *distributed shared memory architecture (DSM)*, where the latency to access memory depends on the relative distances of the processors and their dedicated memory pools.

The **Shared Disk** architecture is characterized by a number of independent processors (CPUs), each with its own memory (M), but a shared collection of disks (S) that is accessible to the DBMS of any Processing Node (PN). This means there is no longer a competition for shared memory, but for access to the shared disk. **(Fig. 3)**

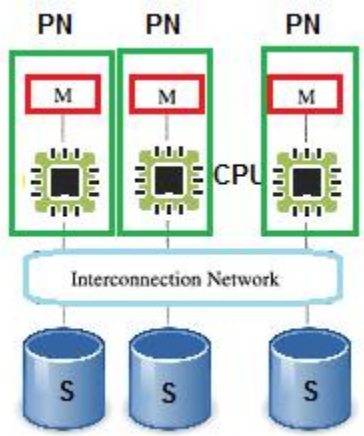


Fig. 3. Shared Disk

Since there is no pool of memory that is shared by all the CPUs, there is no place for the lock table or buffer pool to reside. To set locks, one must either centralize the lock manager on one processor or resort to a complex distributed locking protocol. [3]

The **Shared Nothing** architecture is a distributed computing architecture where nodes are networked to form a scalable system.
Each node has its own private memory (M), processor (CPUs) and storage devices (S) independent of any other node in the configuration. **(Fig. 4)** This means that every node stores its own lock table and buffer pool.
Such architectures are especially well suited to the star schema queries present in data warehouse workloads, as only a very limited amount of communication bandwidth is required to join one or more (typically

small) dimension tables with the (typically much larger) fact table. [3]
Data is horizontally partitioned across nodes, such that each node has a subset of the rows from each table that was distributed and all the replicated tables.

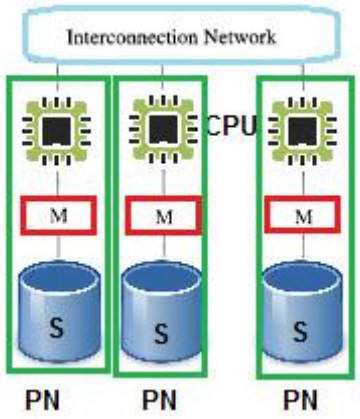


Fig. 4. Shared Nothing

The key feature of shared-nothing architecture is that the operating system not the application server owns responsibility for controlling and sharing hardware resources. [2]

3 What architecture is suited for a cloud data warehouse?

Cloud computing is probably the simplest and best fitted way for smart grids due to its scalable and flexible characteristics, and its capability to manage large amounts of data. [4]
Exactly these two major characteristics have contributed to the importance of parallel data warehousing:

- Data warehouses can become very large and exceedingly resource demanding
- queries and analyses must be answered within acceptable time limits.

One major advantage of shared-memory architecture is that the responsibility to handle the concurrency issues that result from the multiple parallel executions belongs to the operating system.

Unfortunately, shared-memory systems have fundamental scalability limitations, as all I/O and memory requests have to be transferred over the same bus that all of the processors share, causing the bandwidth of this bus to rapidly become a bottleneck. It is unusual to see shared-memory machines of larger than 8 or 16 processors unless they are custom-built from non-commodity parts, in which case they are very expensive. [3]
Therefore, there are significant scalability limits to any data warehouse based on shared-memory architecture.

One major advantage of shared-disk is that all the data is stored in the shared collection of disks. This means that there is no need to distribute parts of the data in each node.
It has the disadvantage of creating a possibly critical bottleneck and scalability limitations in the storage subsystem and interconnections, as all processing units share the same storage system. [1]
Therefore, shared disk architecture gives extremely limited capacity to scale.

Shared nothing does not typically have nearly as severe bus or resource contention as shared-memory or shared-disk machines, shared nothing can be made to scale to hundreds or even thousands of machines. Because of this, it is generally regarded as the best-scaling architecture [5].
Shared-nothing architecture scales better and is well suited for a cloud data warehouse considering very low-cost commodity PCs and networking hardware.
Nowadays, the popular distributed systems have almost all adopted the shared-nothing architectures, including peer-to- peer, cluster, Grid, and the Cloud. [6]
The *shared-nothing architecture* is used for overcoming the scalability limitations, to improve performance when loading and querying data concurrently as well as performing complex joins.

4 Parallel Data Warehouse

SQL Server Parallel Data Warehouse is a *Massively Parallel Processing* (MPP) solution, which means PDW uses a "*shared-nothing*" architecture, where there are multiple physical nodes, each running its own instance of SQL Server with dedicated CPU, memory, and storage.

PDW has two primary types of tables: *replicated* and *distributed*.

The purpose of **replicating tables** is to improve performance by keeping a copy of them on each compute node to support local joins, without having to handle complicated types of parallel queries or dimension-only queries between nodes. This type of table is most often used for dimension tables, but for a very large dimension table is recommended to use distribution. **(Fig. 5)**

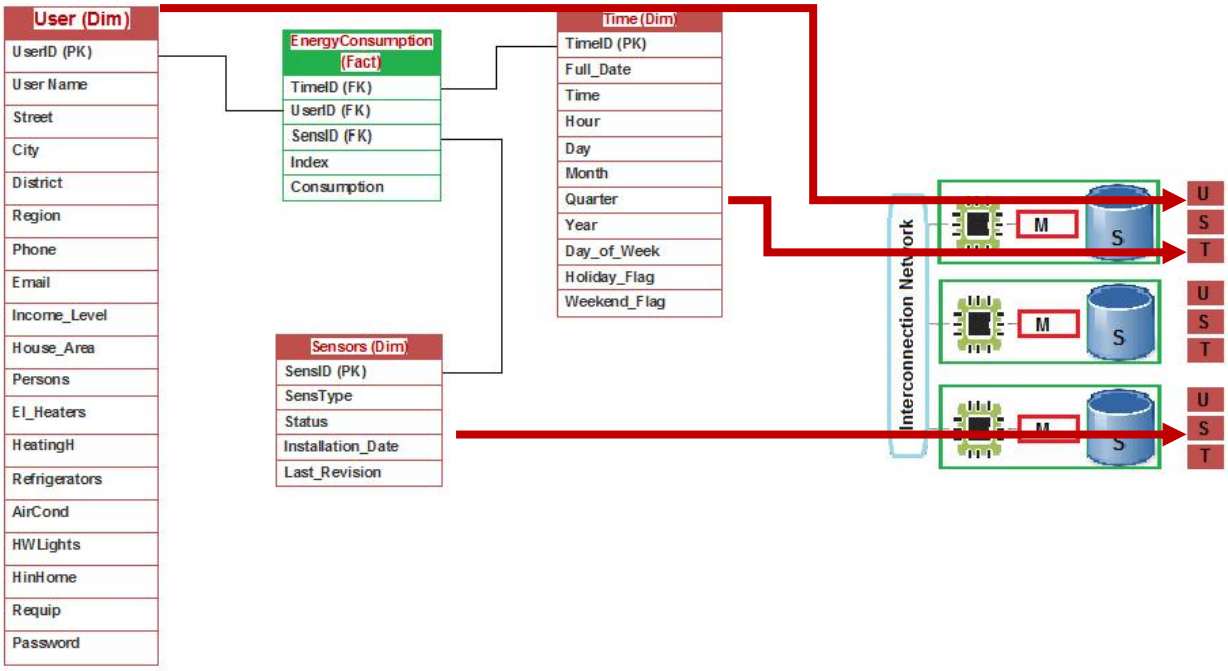


Fig.5. Dimension tables are replicated on every node

The code for replicating a dimension table from the Energy Data Warehouse looks like this:

```
CREATE TABLE User (
UserID varchar(10) NOT NULL,
UserName varchar(50),
Street varchar(50),
City varchar(20),
District varchar(15),
Region varchar(20),
Phone varchar(10),
Email varchar(50),
Income Level varchar(20),
House_Area int,
Persons int,
El_Heaters int,
HeatingH decimal(3,1),
Refrigerators int,
AirCond int,
HWLights int,
HinHome decimal(3,1),
Requip varchar(30),
Password varchar(10))
WITH
(DISTRIBUTION = REPLICATE);
```

If the distribution clause is omitted, the default is REPLICATE.

The same applies for creating the other dimension tables (Sensors and Time).

The purpose of **distribution** is to improve performance by spreading all the rows of a distributed table across all nodes, with the condition that each row from the source table ends up in only one node. **(Fig. 6)**

Large fact or transaction tables that contain billions or even trillions of rows are usually distributed.

The rows are mapped using a hash function on a distribution key from the table.

The distribution key must be a single attribute column, selected considering several criteria:

- High cardinality and even row counts – data must be distributed as evenly as possible across all the distributions on all nodes;

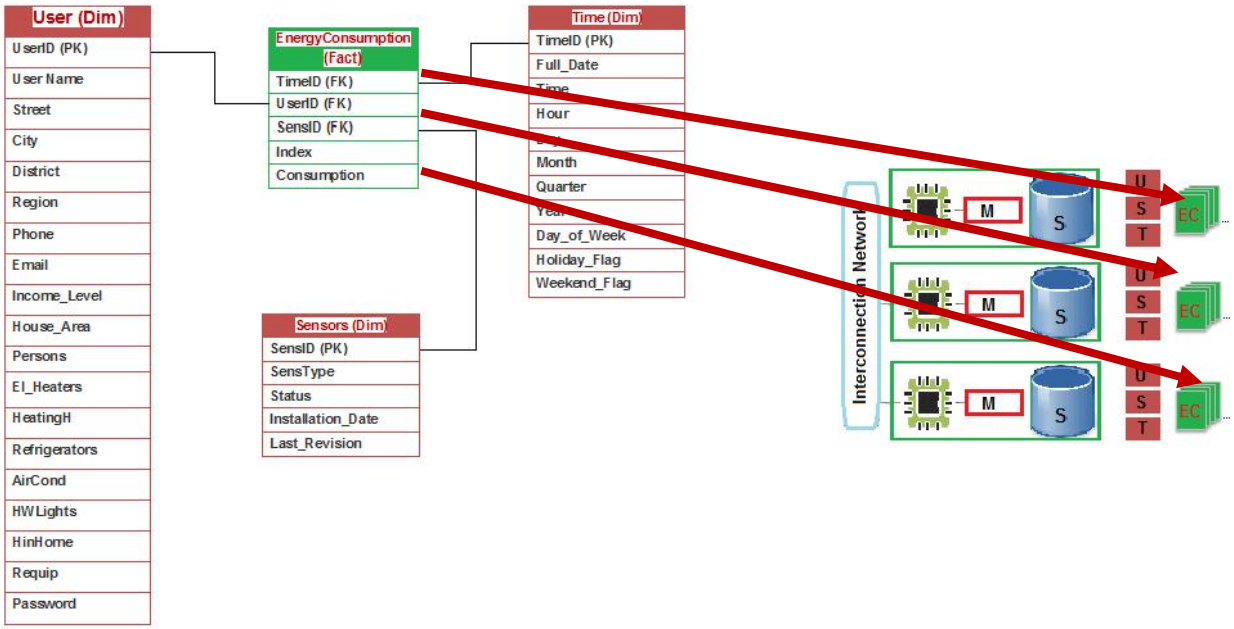


Fig. 6. Fact table is distributed on every node

- Is recommended not to use as distributed key a column that is frequently restricted to a single value in queries.
- When distributing multiple fact tables, the needed analyzes are other considerations that must be taken into account.

  For example, by distributing multiple fact tables on the same distribution key, rows from the first table will be evenly distributed across all nodes and the rows from the other tables will be co-located on the same distribution. This way, queries that may need to join multiple distributed fact tables will perform fast.

The code for distributing the **EnergyConsumption** fact table from the Energy Data Warehouse, having **UserID** as the distribution key, looks like this:

```
CREATE TABLE EnergyConsumption (
TimeID varchar(20) NOT NULL,
UserID varchar(10) NOT NULL,
SensID varchar(10) NOT NULL,
Index decimal(10,3),
Consumption decimal(10,3))
WITH
(DISTRIBUTION = HASH(UserID));
```

The rows are mapped to the distributions using a hash function on the column that was chosen as the distribution key from the table, in our case **UserID**. **(Fig.7)**

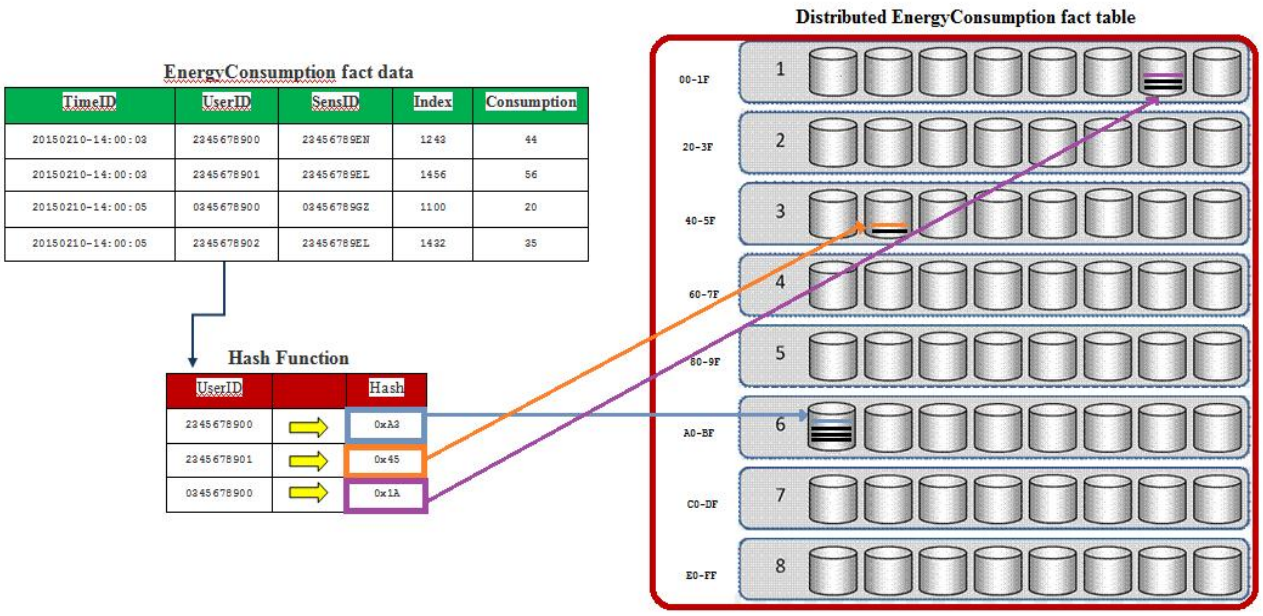


Fig. 7. Distributed **EnergyConsumption** fact table

The UserID key from each row from the EnergyConsumption fact data is passed through a Hash Function. The hashed values map to a single distribution on a single node.

For example, the row for UserID 0345678900 hashes to 0x1A, which maps to the second-last distribution of the first compute node. (Node:1, Distribution:7, Row:3)

The row for UserID 2345678901 hashes to 0x45, which maps to Node:3, Distribution: 2, Row:2.

After the tables are loaded with data, users will use SQL statements to query PDW tables.

For example, the following basic "shared-nothing" join runs against the distributed EnergyConsumption table created above and the replicated tables User and Time:

```
SELECT  u.UserID, sum(ec.Consumption)
FROM User u
JOIN EnergyConsumption ec ON u.UserID =
ec.UserID
JOIN Time t ON ec.TimeID = t.TimeID
WHERE u.Region = 'Oltenia' and t.Year =
'2014'
GROUP BY u.UserID
```

In this case user wants to find out the amount of energy consumption in 2004 for every consumer from Oltenia.

The query will be sent to each node, it will be executed in parallel on each compute node, then the PDW's MPP engine collects and merges all the parallel result sets from the nodes and sends back to the client a single result set. **(Fig. 8)**

This query performs very well even if EnergyConsumption is a very large distributed table. Giving the fact that the dimension tables are replicated on every node, each compute node can answer its part of the parallel query.

Some of the many benefits provided by the SQL Server Parallel Data Warehouse are *query performance* (10-100 times increased due to parallel execution), *data loading performance* (10-40 times faster due to parallel loading data), *the integration with cloud-born data* (Windows Azure HDInsight, Windows Azure blob storage) and *the HDInsight integration* into the PDW rack.

HDInsight integrates Hadoop within a parallel data warehouse processing (PDW). Scaling to the cloud is also easily enabled with the HDInsight Hadoop service on Windows Azure. [7]

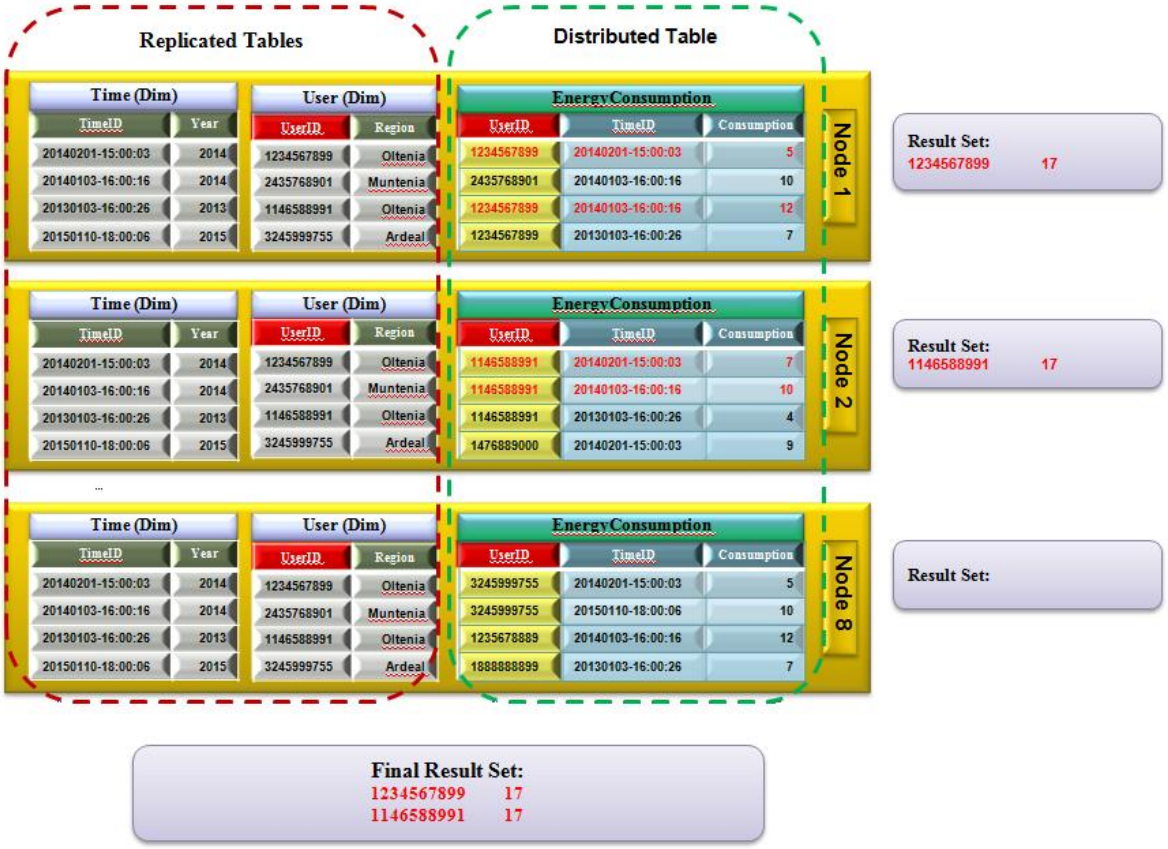


Fig. 8. "Shared-Nothing" join

5 Conclusions

The volume of data collected by utility companies has increased a lot with the progress of the smart metering technology.

A scalable data warehouse architecture is critical, not only to manage very large volumes of data from different sources, but also to assure great response time to users.

Shared-memory and shared disk architectures cannot scale well with the rising amount of data, because these two architectures involve a large volume of data exchange over the interconnection network. But, the interconnection network cannot be infinitely extended, which becomes the main limitation of these two architectures. On the other hand, the shared-nothing architecture minimizes resource sharing.

Between the three basic parallel hardware architectures for data warehousing, "shared-nothing" architecture scales better and is well suited for a cloud data warehouse considering hardware and communication costs, better performance when loading and querying data simultaneously, as well when running complex queries.

Due to parallel execution of joins, query performance has increased up to 10-100 times compared to a join executed in a centralized data warehouse.

Also, due to parallel execution, data loading performance has increased up to 10-40 times compared to a centralized data warehouse.

In conclusion, benefits like storage of large amount of data from different sources (inclusive devices and sensors) and the fast query response times against those data are vital opportunities for competitive advantage for utilities companies.

Acknowledgment
This work was cofinanced from the European Social Fund through Sectoral Operational Programme Human Resources Development 2007-2013, project number POSDRU/159/1.5/S/142115
„Performance and excellence in doctoral and postdoctoral research in Romanian economics science domain".

References

[1] P. Furtado, *A Survey on Parallel and Distributed Data Warehouses*, 2011, https://eden.dei.uc.pt/~pnf/publications/Furtado_survey.pdf
[2] K. Krishnan, Data Warehousing in the Age of Big Data, Publisher: Morgan Kaufmann, 2013, Print ISBN-13: 978-0-12-405891-0
[3] D. J. DeWitt, S.Madden and M. Stonebraker, *How To Build a High-Performance Data Warehouse*, 2009, http://db.csail.mit.edu/madden/high_perf.pdf
[4] J. POPEANGĂ, *Cloud Computing and Smart Grids*, Database Systems Journal vol. III, no. 3/2012.
[5] D. J. DeWitt and J. Gray, Parallel Database Systems: The Future of High Performance Database Processing, Communication of the ACM, Vol. 35(6), pp. 85-98, June 1992.
[6] Fr. Magoules, J. Pan and F. Teng, Cloud Computing: Data-Intensive Computing and Scheduling, Publisher: Chapman and Hall/CRC, 2012, Print ISBN-13: 978-1-4665-0782-1.
[7] Microsoft Corporation, *The Microsoft Modern Data Warehouse*, 2013, http://download.microsoft.com/download/C/2/D/C2D2D5FA-768A-49AD-8957-1A434C6C8126/The_Microsoft_Modern_Data_Warehouse_White_Paper.pdf

Janina POPEANGĂ graduated in 2010 the Faculty of Cybernetics, Statistics and Economic Informatics, Economic Informatics specialization. The title of her Bachelor's thesis is "*Distributed Databases*". In 2012, she graduated the Databases for Business Support master program with the thesis "*Monitoring and management of electric power consumption using sensorial data*". Janina's interests are broadly in the fields of databases and distributed systems. Since 2012 she is a Ph.D. Student in the Doctoral School of Bucharest Academy of Economic Studies. Her research focuses on real-time database systems, business intelligence analytics, sensor data management, smart grid and renewable energy.

Informatics Solutions for Prosumers connected to Smart Grids

Simona Vasilica OPREA
The Bucharest University of Economic Studies, Romania,
simona.oprea@csie.ase.ro

This paper gives a brief overview about electricity consumption optimization based on consumption profiles of electricity prosumers that are connected to smart grids. The main object of this approach is identification of informatics solutions for electricity consumption optimization in order to decrease electricity bill. In this way, larger scale integration of renewable energy sources is allowed therefore entire society will gain benefits. This paper describes the main objectives of such informatics system and stages for its implementation. The system will analyze the specific profile and behavior of each electricity consumer or prosumer, automatically assist him to make right decisions and offer optimal advice for usage of controllable and non-controllable appliances. It will serve, based on big data transfer from electricity consumers or prosumers, as a powerful tool for grid operators that will be able to better plan their resources.

Keywords: *smart metering, advanced tariffs system, big data, electricity consumption optimization, prosumer, renewable energy sources*

1 Introduction

Our society is facing a significant change of paradigm and many challenges in energy fields. Large scale renewable sources integration has required innovative solution for the benefit of entire society. Nowadays traditional consumption prediction is almost of no use because the electricity modern consumers is beginning more and more active, acting in controlled mode based on clear and efficient incentives. They may have several options: consume energy from the grid when it is cheaper, act as prosumer and introduce power into the grid out of their own micro-generation, when it is more expensive, store energy for later usage, control their appliances and make better decisions for their budget. Smart metering, advanced tariff schema and informatics systems play an important role so that electricity consumer has more options. The price of electricity in each time period is given by availability of renewable sources. Whenever there is more wind then the price will drop and this will stimulate electricity consumption. In case of the rigid tariff schema there is no incentive to increase electricity consumption and therefore integrate less renewable sources. Flexible tariff schema should be well implemented so that consumers will not lose interest in optimization of their electricity consumption. By means of communication technology, smart metering systems involve big data transfer from electricity consumer or prosumer to grid and vice versa. This requires a scalable and powerful informatics system that integrates, analyzes, monitors and assists the behavior of electricity consumers or prosumers and grid performance.

For development of such system, specific objectives are taking into account:

- study of electricity consumption models in smart grid environment and model load profiles (households and non-households) based on type of appliances/specific activities, frequency and duration of their usage;
- identify informatics solutions to process data from smart metering devices taking into account the big data paradigm;

- design the data mining solutions for optimization of consumption.

The paper presents some informatics solutions designed and implemented into a prototype that will assist decisional process and propose optimal solutions for energy supply at minimal prices. It will assist decisional process at the operator grid level by integrating and analyzing consumption data in order to use their resource in an efficient manner. The solutions are design based on some specific conditions of electricity sector in Romania (load profiles, generation mix, volume and type of RES), but they can be extended for wide scale application.

2 Challenges related to new trends of electricity sector

2.1 Smart grid concepts

The concept of smart grid had been defined since October 1997 in the paper Grids get smart protection and control, by authors Khoi Vu, Miroslav M. Begovic, Damir Novosel, published in IEEE Journal Computer Applications in Power [1]. Power systems based on smart grid are characterized by a sum of equipment and modern systems, based on informatics and communications that collect and automatically make decisions in accordance with producers and consumers behavior in order to improve efficiency, reliability, economic aspects and sustainability of generation, transmission, distribution and electricity supply.

The most known components of smart grid are: automation and relay protection, acquiring, monitor and control data systems (SCADA), intelligent metering devices or smart metering, integrated communication systems, decision support systems and advanced interfaces, etc.

2.2 Renewable sources integration

Smart metering systems (SM), along with electricity generation based on renewable sources (RES) are advanced technologies recently introduced as a future solution that solves issues regarding conventional energy sources insufficiency, greenhouse gas emissions and dependency of primary energy sources located outside European Community, etc. European strategies have been taken over as national targets in the Romanian legislation.

Directive no. 28/2009 of the European Parliament and of the Council of 23 April 2009 on the promotion of the use of energy from renewable sources and amending and subsequently repealing Directives 2001/77/EC and 2003/30/EC establishes a common framework for promoting renewable energy sources. It sets mandatory national targets consistent with a 20 % share of energy from renewable sources and a 10 % share of energy from renewable sources in transport in Community energy consumption by 2020 [2]. The Romania's objective is 24%, as a consequence of its RES potential.

RES integration raises a couple of major problems as a matter of electricity production volatility, less predictable character and higher generation volume at night when electricity consumption is low. Integration of more renewable sources will be possible by means of SM along advanced tariff schema (ATS) and informatics systems that analyses the electricity consumption profile and decide based on electricity price and internal factors such as frequency and duration of appliances usage on the appropriate level of consumption.

2.3 Smart metering

In order to implement SM, National Regulatory Authority for Energy (ANRE) has been approved national targets for grid distribution operators that should be reached until 2020, implementing SM gradually up to about 80% of final customers. SM includes metering subsystems that contain measuring equipment (metering device, transformer and access security equipment), information transfer subsystems and information management subsystem.

SM are electronic systems that measures electricity consumption, ensure bidirectional

secured transmission of information in both ways from/to electricity consumer and supply much more information than a regular meter, using electronic communication ways [3].
The role of SM, ATS and informatics systems for consumption decisional assistance is to flat as much as possible daily load curve and in this way to allow integration of larger volume of renewable sources into the power system.
The main objective of SM consists in measuring electricity adding more functions to the conventional measuring systems. These functions are useful for electricity consumers or prosumers and for strategic management of grid operators.
At the electricity consumption level, comparing with previous period before SM implementation, the consumers can play an active role, having the administration opportunity of automatic controllable consumption (washing machine, batteries, electric oven, heating/cooling systems, etc.). Also they can choose when to use non-controllable appliances, storage equipment (such as electric cars batteries) that can be unloaded back into the grid and/or supply energy produced by micro-generation (photovoltaic panels installed on the roofs or buildings' facades, small wind turbines) according to the electricity price. Electricity consumers that periodically supply energy to the grid are known as prosumers.
Informatics system interacts with smart metering system, controllable and non-controllable appliances, storage equipment, grid, micro-generation, etc.

3. Intelligent solutions for electricity consumption optimization

As a consequence of innovative character of SM, at the electricity consumer level in Romania does not exist such systems that optimize consumption according to their profile, hourly (or even quarterly) electricity prices and availability of micro-generation sources, but there are research activities at national level [5], [6]. At international level there are dedicated studies and also research activities regarding implementation of such solutions [7], [8], [9].
Load to prosumer profiles
Based on activities that are carried out by consuming electricity, final consumers are categorized into two: household and non-household consumers.
Final non-household consumers are characterized by specific consumption, defined by profile curve or profile that is built based on a certain procedure [4] according to the activities that are carried out, such as: electric transportation, gas stations, civil works, hospitals, public utilities, services, schools, agricultural activities, etc. These categories can be again split into two or more category (for instance services may have different profile curves) base on the volume of such activities and detailed profile that is required for specific purposes.
Final household usually have certain types of appliances: refrigerator, washing machine, dish machine, electric oven, electrical grill, water heater, iron, electric centralized heating system, electrical pot, vacuum, air-conditioning, car batteries, etc. Out of these appliances only some of them can be automatically controlled and used at certain time intervals when electricity price is lower (e.g. washing machine, dish machine, electric oven, car batteries can be charged at night).
Existing load profiles are obtained based on historical behavior of each typical consumer, but nowadays more and more non-household and household consumers become prosumers by installing photovoltaic (PV) panels, wind generating turbines or other types of RES.
Therefore load profile has to be transformed into prosumer profile that is automatically accomplished by prototype, through consumption optimization component (figure 1) that is capable of tuning prosumer profile.

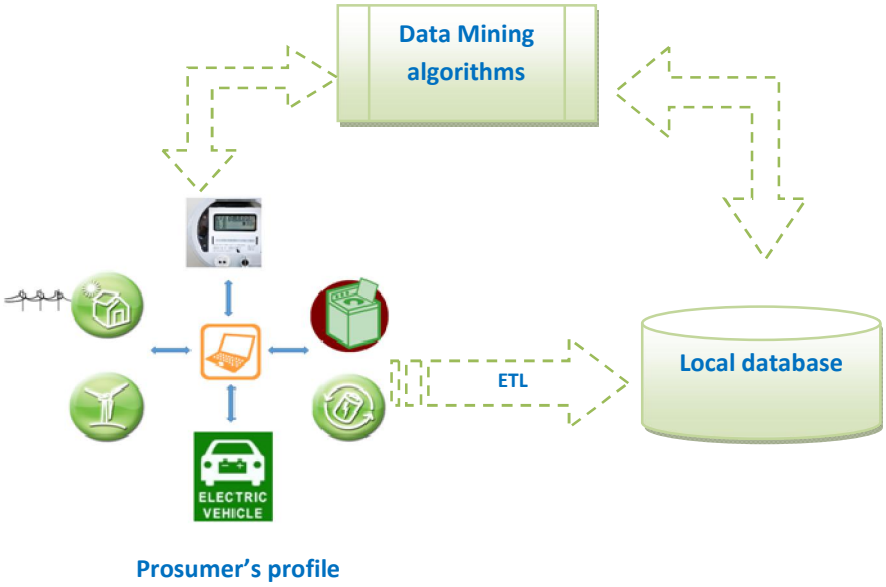


Fig. 1. – Consumption optimization component for prosumers

This component is based on data mining algorithms that take into account: type of working machines for non-household prosumers, type appliances for household prosumer, diurnal or nocturnal hourly consumption, seasonal changes (heating or cooling requirement) including efficiency of buildings, local self-generation volume, electricity price, etc.
Classification and clustering algorithms are adapted in order to optimize electricity consumption. Due to the fact that these algorithms are applied on very large volume of data that are generated mainly by consumption, they require an extract, transform and load processes that prepare data for the learning phase.

4. Decision support for strategic management

At strategic management of grid operators level, data integration received from SM permit a better planning of resources, reducing loses and integrating a larger volume of RES. By taking into advantages of Business Intelligence tools for aggregated data reporting through dashboards and also forecasting components related to the consumption profiles opposite to RES generation, at this level of management the decision process can be improved to efficiently analyze consumers' behavior and optimally plan resources of grid operators. Thus, a possible solution implies incentives for prosumers to consume or generate electricity from/to the grid by means of advanced tariff system. Adequate hourly or quarterly electricity prices are capable to motivate consumption at night during off-peak time intervals so that it will allow higher generation from RES when meteorological condition are favorable and to discourage consumption at peak hours.
Operational experience of wind power plants in Romania shows that they have better condition at night when consumption is reduced. In other words, their operation is less reliable and not helping the operation of the power system [11], [12].

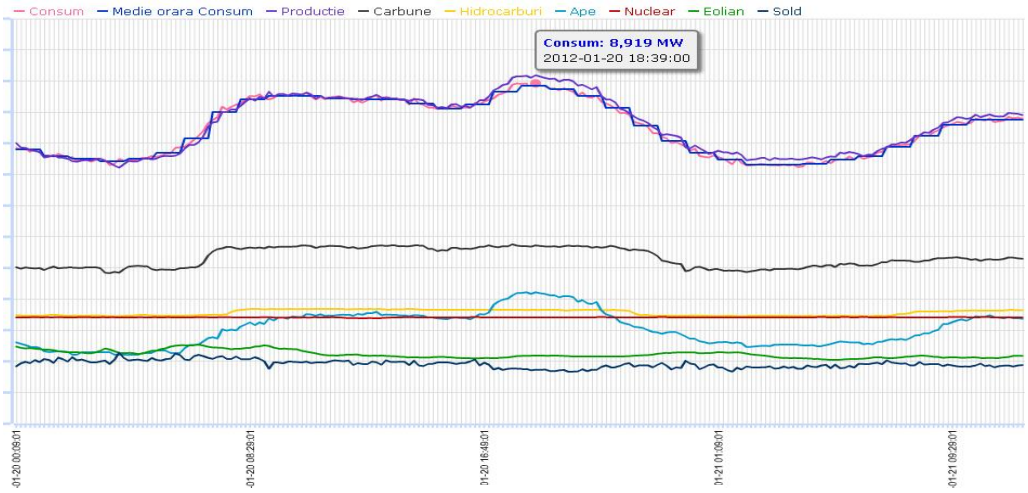


Fig. 2 - Power system load curve

Figures 2 and 3 depict typical power system load curve profile and 24-hour average generation of a large wind power plant in Romania named Tariverde.

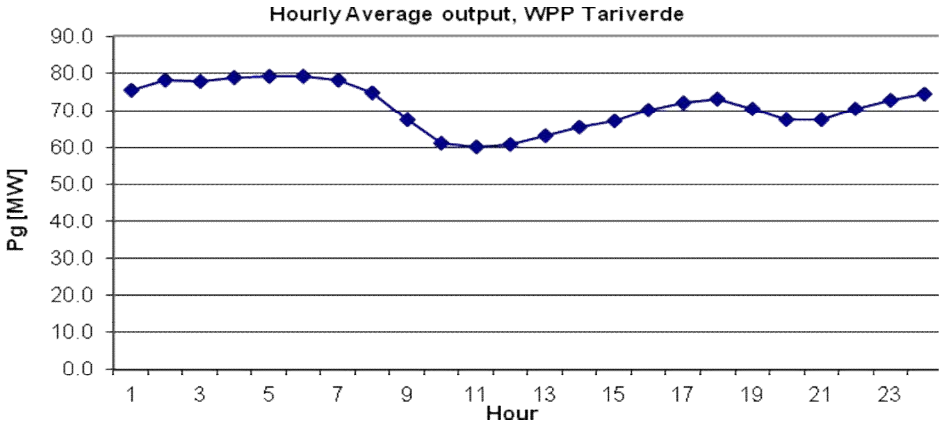


Fig. 3 - Hourly average of generation for a large wind power plant in Romania

It can be easily seen that the two curves are opposite, at night wind blows stronger than it blows during day time. Desirable case would be when second curve follows the first curve, by simultaneously increasing and decreasing. In Romania wind power plants operate at less than 20% of their total capacity and many times they were restricted especially at night, therefore more electricity consumption at night will let them operate with fewer restrictions.

At the national level, electricity consumers do not play an active role and there is no informatics system to assist decision makers. Thus, the prototype, through the prosumers' module can analyze electricity consumption based on their profile and to find optimal solutions for electricity supply that lead to actions meant to diminish electricity bill and also to avoid the limitation of wind power plants' generation. This system can change the consumer behavior as a result of ATS and SM that indirectly lead to large scale integration of RES (especially wind).

5. Research methodology

For analysis and design of informatics systems prototype, an object oriented methodology is used within an iterative development cycle. Iterative development cycle represent a succession of stages, one after the other, at each iteration a part of the entire system is achieved and validated and it is considered as starting point for the next

iteration. In this cycle, each new iteration contains previous iterations with more details, modifications, by adding or eliminating some elements defined prior to it. Prototype's architecture is designed on three levels, accordingly to the decision support systems architecture described in [11], [13]: *data, models* and *interfaces* and each level contains specific methods and techniques (figure 4):

- *at data level*, data collecting and processing technologies are used. Data, provided by smart metering systems from all electricity consumers located in a certain control area, is integrated into several models of analysis and reporting at each grid operator responsible for its control area;
- *at models level*, in order to determine changes in electricity consumers' behavior and optimize consumption, algorithms and methods for extracting knowledge from data are used. Data mining techniques have an important impact because the process of automatic learning focuses on extracting regularities from the available set of samples;
- *at interface level*, for reports and dynamic analyses, business intelligence technologies are used.

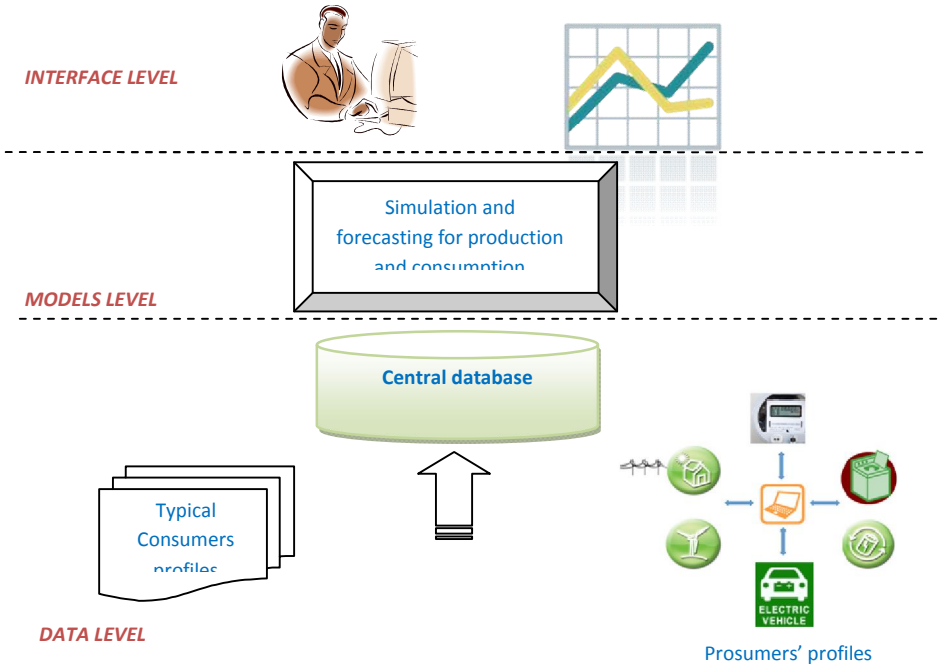


Fig.4 – Prototype's architecture

The following stages are followed in order to design and develop the prototype:
Stage 1 – Global analyses with following activities:

- planning activities for identifying the main and detailed requirements for designing informatics system;
- study of metering system functionalities emphasizing links and correlations with informatics system;
- SWOT analysis for advanced tariff systems;
- study of load profile for household and non-household consumers or prosumers based on activities and working regime;
- analyze the optimization solutions for electricity consumption supplied from smart grids that include SM, controllable and non-controllable appliances, micro-generation (prosumers), storage equipment, etc. Optimization module contains all possible options; each of them could be

activated by request (for instance micro-generation and storage should be flexible options depending on each type of consumer).

Stage 2 – Models designing with following activities:

- design the data model that collect and integrate data from SM. This activity consist in identification of data sources for desirable prototype and identification of methods for integration and processing of data into a central database;
- develop data mining algorithms for detailed analyses of electricity consumers or prosumers types in order to estimate their behaviour. Based on the data model, algorithms such as clustering, regression, classification and association are applied, comparing their performances;
- develop the electricity consumption optimization model at consumers or prosumers level that covers its needs with minimum costs for electricity and considering even revenues in case of prosumers;
- develop the analysis model at grid operators level that help them to know real time electricity consumers/prosumers' behaviour and better plan their resources for grid operation.

Stage 3 – Building of informatics system prototype. This stage consists in the following activities:

- achievement of prototype functionalities at electricity consumer or prosumer level. Online friendly interfaces are needed, available on mobile devices;
- achievement of data model at grid operators level. The central database allows the implementation of analysis models for obtaining reports for grid operation activities;
- achievement of prototype functionalities at grid operator level by business intelligence tools that deliver dynamic reports, dashboards for information related to real time electricity consumption;
- testing and evaluation of informatics prototype. Prototype functionalities are tested by using real time data and simulations. Future development directions and main risks about informatics system implementation are identified. In this sense, hedging measures are defined.

Informatics system prototype based on proposed solutions analyses types of electricity consumers or prosumers (householders, non-householders, controllable and non-controllable appliances) and evaluate supply of electricity consumption from own micro-generation, opportunity of generation/consumption to/from the grid, frequency and duration that characterize consumption, etc. The controllable appliances can be automatically shut down/started up based on optimization model via online interfaces and for the other appliances, based on data mining algorithms that extracted knowledge from previous behavior, it offers optimal variants of operation according to some factors such as frequency and duration of operation and electricity price.

In this way, electricity consumers or prosumers play an active role being capable to contribute to generation/consumption balance in the power system, flatting fluctuations that come from RES and helping to better integrate large size RES.

Conclusions

In this paper a brief overview about electricity consumption optimization based on consumption profiles of electricity consumers or prosumers that are connected to smart grids is given. The main object of this approach is identification of informatics solutions for electricity consumption optimization so that electricity bill to significantly decrease. In this way larger scale integration of

renewable sources is allowed therefore entire society will gain benefits. This paper describes the main objectives of such informatics system and stages for its implementation. The prototype analyzes the specific profile and behavior of each electricity consumer, automatically assists him to make right decisions and offers optimal advice for usage of non-controllable appliances. It serves based on big data transfer from electricity consumers or prosumers as a powerful tool for grid operators that is able to better plan their resources.

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] K. Vu, M.M. Begovic, D. Novosel, "Grids get smart protection and control", IEEE Computer Applications in Power (IEEE) Journal, ISSN 0895-0156, October 1997.

[2] Directive no. 28/2009 of the European Parliament and of the Council of 23 April 2009 on the promotion of the use of energy from renewable sources and amending and subsequently repealing Directives 2001/77/EC and 2003/30/EC

[3] Ordinul ANRE nr. 91/2013 privind implementarea sistemelor de măsurare inteligentă a energiei electrice

[4] Decizia ANRE 143/23.01.2013 de aprobare a procedurii privind elaborarea si aplicarea profilurilor specifice de consum in zona de licenta a CEZ Distributie

[5] Eremia M, Toma L, Bulac C, Tristiu I, Otomega B – Smart Grids – Innovative solutions for electric transmission and distribution networks, Symposium on ELECTRIC POWER GRIDS AND SYSTEM, Bucureşti, 2011

[6] Păunescu C, Toma L, Bulac C, Eremia M – Energy management system in the smart home, CIEM, Bucureşti, 2013

[7] Rathnayaka A, Dinusha J, Vidyasagar M.P, Dillon T, Hussain O, Kuruppu S – Analysis of Energy Behaviour Profiles of Prosumers, 10th IEEE International Conference on Industrial Informatics (INDIN), ISBN 978-1-4673-0312-5, 2012

[8] Guo Y, Wu C, Tsinalis O, Silva D, Gann D – WikiSensing: Towards a Cloud-based Sensor Informatics Platform for life in a Digital City, 3rd Digital Economy conference (DE2012), Digital Futures, Aberdeen, UK, 2012

[9] Dedrick J, Zheng Y – Information systems and smart grid: New directions for the IS community, iConference 2013 Proceedings, Fort Worth, TX, USA, 2013

[10] Auer J, Keil J – State-of-the-art electricity storage systems, DB Research, martie 2012, ISSN 1612-314X, http://proclimweb.scnat.ch/portal/ressources/2892.pdf

[11] S-V. Oprea, A. Bâra, V. Vlăducu, A. Velicanu - Data Level's Integrated Model for the National Grid Company's Decision Support System, Recent Advances in Computers, Communications, Applied Social Science and Mathematics, International Conference on Computers, Digital Communications and Computing (ICDCCC'11), Barcelona, Spain, 15-17 septembrie 2011, pp 167-172, ISBN: 978-1-61804-030-5, WSEAS Press

[12] Oprea S.V, Petrescu D.E, Bolborici D, Stanescu O.R – Aspects related to wind power plants operation in Romania, Conference of Energy Engineering 2012, Oradea, Romania

[13] S-V. Oprea, I. Botha, A. Bâra, A. Velicanu - Prototype of a Decision

Support System for analyzing and forecasting the Wind Energy Production in Romania. The 1st International Conference on INFORMATION TECHNOLOGY and COMPUTER NETWORKS (ITCN '12), Vienna, Austria, Recent advances in computer engineering Series, Latest trends in Information Technology, ISSN: 1790-5190, ISBN: 978-1-61804-134-0, pp. 123-129, , WSEAS Press

Simona Vasilica OPREA is a Senior Engineer and she has graduated the Polytechnic University in 2001, holds a Master Diploma in Infrastructure Management Program, Yokohama National University, Japan in 2007 and a PhD diploma from 2009. She is the author of over 20 articles, from which 3 ISI Web of Science indexed and 2 included in SCOPUS international data-base. Domains of competence: wind farm, investment opportunity analysis, studies of prognosis, stationary and dynamic regimes, short circuit calcula-tions. Since 2014 she is PhD candidate at the Bucharest University of Economic Studies.

Reshaping Smart Businesses with Cloud Database Solutions

Bogdan NEDELCU, Andreea Maria IONESCU, Ana Maria IONESCU,
Alexandru George VASILE
University of Economic Studies, Bucharest, Romania
bogdannedelcu@hotmail.com, andreea_maria_ionescu@yahoo.com,
anamaria.ionescu27@gmail.com, alexandru.vasile26@gmail.com

The aim of this article is to show the importance of Big Data and its growing influence on companies. We can also see how much are the companies willing to invest in big data and how much are they currently gaining from their big data. In this big data era, there is a fiercely competition between the companies and the technologies they use when building their strategies. There are almost no boundaries when it comes to the possibilities and facilities some databases can offer. However, the most challenging part lays in the development of efficient solutions - where and when to take the right decision, which cloud service is the most accurate being given a certain scenario, what database is suitable for the business taking in consideration the data types. These are just a few aspects which will be dealt with in the following chapters as well as exemplifications of the most accurate cloud services (e.g. NoSQL databases) used by business leaders nowadays.

Keywords: *Smart business, cloud database, cloud solutions, NoSQL databases, weaknesses, key-value databases, Riak, columnar databases, Hbase, document-oriented databases, MongoDB, graph databases, Neo4j, cloud services*

1 Introduction

It is becoming increasingly clear that the cloud technology brings a different outlook to the digital world, making it easier to understand, powerful and more efficient. Not only is the everyday consumer affected by these continuously shifts in technology, but also the businesses around the world. What the business leaders should bear in mind is that their organizational strategies must keep up with the new trends in technologies, and, therefore make use of the cloud in order to better manage their resources, improve internal performance among with the necessary IT-related knowledge that helps accomplish success.

The companies should start building their strategies by using the latest trends in technology considering the high applicability of the applications within cloud as well as the benefits which could arise when using cloud integrated solutions. As a matter of fact, cloud computing sometimes could be the most effective answer to common problems that the businesses usually encounter such as the reduction of the costs, the implementation of new services and applications. Companies can add computational capacity faster using the cloud than they usually can using in-house stuff. Nowadays, the business models tend to gain access to services instead of have ownership of products. For example, Microsoft Office 365 and Google Apps provide applications that are available over the Internet (instead of via traditional software packages that must be purchased and installed). [1]

Despite the fact that the cost of implementing cloud setups has reduced and the performance improved significantly, the cloud reliability has left a question mark over, especially because most of the enterprises have concerns about placing sensitive data on a third-party cloud. The evolution of the cloud and how far has this trend gone in terms of time and space are also some subjects

which will be tackled within the following pages. “The Long Nimbus” article released by the Economist magazine about the impact of cloud computing on company organization structures states that ‘‘Businesses are becoming more like the technology itself: more adaptable, more interwoven and more specialized. These developments may not be new, but cloud computing will speed them up.” [2]

The Riak, Apache HBase, MongoDB, and Neo4J cloud databases services are just a few examples that will be presented in this article. It is particularly important when creating a business strategy to understand the capabilities and constraints of each type of the databases in order to choose the right one for the specific job. The focus should be more on whether a particular database is suitable when considering a business problem space, the usage patterns as well as the available resources.

Overall, one of the major benefits that the companies can gain by using the cloud technology comes not from cost savings for IT resources on a per-use basis, but from the revenue they earn by becoming more flexible and responsive when it comes to customers’ changing needs. This would further enable businesses efficiently deliver their new products and services as well as expand successfully into new markets.

“While enterprise IT use will continue to grow, the largest source of economic impact through 2025 will likely come from enabling the delivery of services and applications to Internet users. We estimate the total potential economic impact for cloud technology across sized applications could be $1.7 trillion to $6.2 trillion in 2025.”3 Taking into consideration this huge potential for the global economy and the fact that the cloud technology would definitely reshape the world through its fast changing development, the smart corporate business should come up with strategies using the tools of the cloud services that would boost both their productivity and performance.

2. The Cloud Concept

The cloud is not simply the latest fashionable term for the Internet. Though the Internet is a necessary foundation for the cloud, the cloud is something more than the Internet. The cloud is where you go to use technology when you need it, for as long as you need it, and not a minute more. You do not install anything on your desktop, and you do not pay for the technology when you are not using it.

NIST (National Institute of Standards and Technology) defines cloud computing as “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” [4]

Nevertheless, confusion remains about exactly what it is and when it's useful, causing Oracle's CEO Larry Ellison to vent his frustration: "The interesting thing about cloud computing is that we've redefined cloud computing to include everything that we already do.... I don't understand what we would do differently in the light of cloud computing other than change the wording of some of our ads." [5]

Cloud computing is a computing platform that resides in a large data center and is able to dynamically provide servers with the ability to address a wide range of needs, from scientific research to e-commerce. The provision of computing resources as if it were a utility such as electricity, while potentially revolutionary as a computing service, presents many major problems of information policy, including issues of privacy, security, reliability, access, and regulation. This article explores the nature and potential of cloud computing, the policy issues raised, and research questions related to cloud computing and policy. Ultimately, the policy issues raised by cloud computing

are examined as a part of larger issues of public policy attempting to respond to rapid technological evolution [6].

2.1 Short History

When we think of Cloud Computing, we think of situations, products and ideas that started in twentieth century. Historically, behind all of these modern concepts it is another story.

It all begun with a gradual evolution that started in the 1950s with mainframe computing. Multiple users could access a central computer for dumb terminals, whose only function was to provide access to the mainframe. Because of the costs to buy and maintain mainframe computers, it was not practical for the organizations to buy and maintain one for every employee. The best solution to save money, in this complicated piece of technology was to share access to a single resource.

Around 1970, the virtual machine (VMs) was created. Using virtualization software like VMware, it became possible to execute one or more operating systems simultaneously. This kind of operating system took the 1950s shared access mainframe to the next level, permitting multiple distinct computing environments to reside on one physical environment. Virtualization came to drive the technology, and was an important catalyst in the communication and information evolution.

Telecommunications companies only offered single dedicated point–to-point data connections. The newly offered virtualized private network connections had the same service quality as their dedicated services at a reduced cost. Instead of building out physical infrastructure to allow for more users to have their own connections, telecommunications companies were now able to provide users with shared access to the same physical infrastructure. Nowadays, SoftLayer is one of the largest global providers of cloud computing infrastructure which was founded in 2005, but was acquired in 2013 by IBM to form IBM Cloud Services Division. In 2011 the company reported hosting more than 81,000 servers for more than 26,000 customers. IBM already has platforms in its portfolio that include private, public and hybrid cloud solutions. The purchase of SoftLayer guarantees an even more comprehensive infrastructure as a service (IaaS) solution. While many companies look to maintain some applications in data centers, many others are moving to public clouds.

In the end, the story is not finished here. The evolution of cloud computing has only begun and lead us to a widespread area. Even if companies are creating their own internal cloud called “private” or others are moving to clouds from external services known as “public”, this process of “moving” is the most profound evolution and produce significant changes in the way they run.

2.2 Cloud Services

There are three models of cloud services:

- SaaS (Software as a Service)

It refers to the capability of the clients to access the provider’s applications which are running on a cloud infrastructure. The applications are available from various client devices, through a simple client interface, like a web browser, or an interface of the program. Examples: Email services provided by big companies like Microsoft (Hotmail), Google (Gmail), or Yahoo! (Yahoo Mail).

- PaaS (Platform as a Service)

It refers to the ability of customers to install their applications (created or acquired) on the cloud infrastructure using programming languages, libraries, services and tools provided by the supplier.

Allows access to information about new software, given low cost and preset distribution channels to attract more

efficient the customer, which proves that the cloud is a way to improve your business strategy.

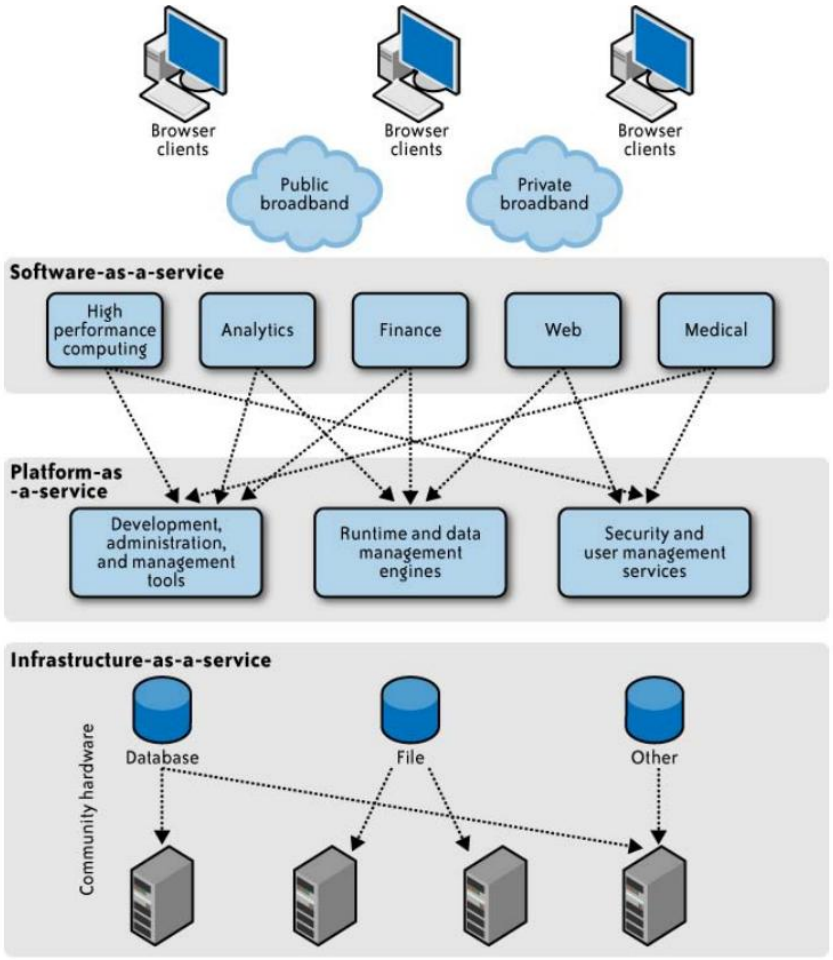


Fig 1. Models of cloud services [8]

- IaaS (Infrastructure as a Service)

It refers to the customers' capability to use processing power, media storage, networking and other basic computing resources from the provider to install and run operating systems, applications and other software on a cloud infrastructure.

The basic technique is virtualization (the ability to provide a consistent view on a set of resources), namely: virtualization of servers, equipment storage or networks.

IaaS services provided are: interface for resource management and interface for system monitoring.

Cloud computing is not so much a technology as it is the combination of many preexisting technologies. These technologies have matured at different rates and in different contexts, and were not designed as a coherent whole; however, they have come together to create a technical ecosystem for cloud computing. New advances in processors, virtualization technology, disk storage, broadband Internet connection, and fast, inexpensive servers have combined to make the cloud a more compelling solution.

When it comes to the usage of databases as a cloud service, they are more than columns and rows. It is not compulsory to know the profile of a business and the field it operates, but the most important aspect is the way the collected data is stored. Even though it might be a mailing list or an extensive product inventory, the type of database you chose might be beneficial or harm the way data is structured.

Although the businesses today focus their competitive advantage on how fast they

react when making a decision, sometimes this solution does not guarantee the proper answer to the complex business issues they were dealing with. In this fiercely and challenging economic environment, collecting more and more data and accessing it as fast as possible is mandatory especially because the transformation of data assets into innovative strategies can most of the times maximize the productivity of resources, and therefore lead to sustainable growth.

Most companies today have plenty of data. Creating intelligence and gleaning real insight and value from this data is what continues to elude organizations. Despite years of talk about scorecards and metrics, gut feelings and experience are often still the guides for making important, sometimes critical decisions. [7] As James Taylor, the CEO of Decision Management Solutions, states "organizations need to be much more focused on directing analysts towards business problems. Find the decisions that are going to make a difference to business results..." [8]

In the past, companies have been using relational databases to store their structured data. Today, even though they had a huge impact on the world of databases and unlocked data for many applications, relational databases lack in characteristics necessary to cope with the fast-changing transaction of data in the big data era.

NoSQL databases are the answer that solves many of these problems because they make you see the database world in a new light. In 1998 Carlo Strozzi first mentioned the NoSQL to name his lightweight, open-source relational database which did not convey with the standard SQL interface. However, the term became popular in the late 2009 when people realized how beneficial they were. Since then, the NoSQL movement has continuously developed. Although they have no schema, the NoSQL databases are fast and adapt easily to the businesses leaders' needs. For example, the NoSQL can work with the non-relational distributed and unstructured data which is the type of data most of the companies typically collects. [1]

The NoSQL databases are particularly known for scalability, agility and flexibility. There are different types of NoSQL databases and each focuses on different applications: Key-value stores, Wide-column (Columnar) stores, Document database, and Graph store.

The Key-value (KV) stores the pairs of keys to values in approximately the same way that a map (or hashtable) would in any popular programming language. They might have different functionality: some KV implementations might provide a means of iterating the keys, while others can allow complex value types such as hashes or lists. For example, a filesystem can be considered a key-value store, if you think of the file path as the key and the file contents as the value. Although databases of this type can be incredibly performant, they would lack the capacity to manage complex query and aggregation needs.[2]

The columnar (column-oriented) databases took their name from a particular feature in their design – data from a given column (in the two dimensional table sense) is stored together. On the other hand, a row-oriented database (like an RDBMS) is used to keep information about a row together. Although it is hard to see the difference, the impact of this design decision has a more deep meaning than it seems. An inexpensive feature of the column-oriented databases is that adding columns is done a row-by-row basis. It is not compulsory for a row to have a set of columns, the tables would still remain sparse without paying the cost of storage for null values. With regard to the structure, the columnar type of databases is about midway between relational and key-value types.

As its name sounds, the Document-oriented databases store documents. Basically, a document resembles a hash, it has a unique ID filed and its values can be

any of a variety of types, including more hashes. Documents can permit a high degree of flexibility because they have the possibility to form nested structures. Moreover, there are few restrictions on incoming data imposed by the system. The basic condition it has to meet is to be expressible as a document. Depending on what type the document database is, it can have different approaches regarding the indexing, ad hoc querying, replication, and consistency. Only by understanding these differences and the impact of them on the documents can lead to a wise decision and, thus, contribute to the creation of a feasible business strategy.

Less commonly used, the Graph databases are usually considered the best at coping with the highly interconnected data. Their structure is made up from two elements: nodes and the relationship between the nodes. Both elements have a variety of properties – key-value pairs – that store data. Of course, the real strength of graph databases lays in the capacity to travel the nodes by following relationships.

"A variety of NoSQL databases are available, each intended to focus on a particular data storage and access strategy. While a typical NoSQL database might not be as comprehensive as a relational database, its focus on a well-defined range of tasks enables it to be highly optimized for those tasks. The key to success is to understand the features of different NoSQL databases, and then use these features to implement a repository that matches the specific requirements of your applications."[3]

3. The Cloud Computing impact on the business

3.1 Cloud Movement

Cloud computing can be considered primarily as a cost-saving technology that's used here and there on cost-cutting projects and for quick fixes to provide point solutions to specific operational problems. On the other hand, cloud computing can be understood in the context of an overall business strategy based on agility and responsiveness. Cloud computing certainly provides cost savings in some situations, but cost savings is not the most important benefit. The real value of cloud computing is the way in which it can be used to support an overall strategy designed to create agility for the business.

The spread of cloud computing is the best example of "creative destruction". The phenomenon was popularized by the economist Joseph Schumpeter who described it as the "process of industrial mutation that incessantly revolutionizes the economic structure from within, incessantly destroying the old one, incessantly creating a new one."

Why move to the cloud? There are plenty of good reasons, but mainly it makes good business sense: cloud computing lets you focus on what's important, your business. This is called efficiency. This field can be used for almost all types of applications and it is clear that it saves its users money.

First of all, the hardware is fully utilized. Cloud computing brings natural economies of scale. The practicalities of cloud computing mean a high utilization and smoothing of the inevitable peaks and troughs in workloads. Sharing sever infrastructure with other organizations, allows the cloud-computing provider to optimize the hardware needs of its data centers, which means lower costs for business.

Secondly, when you run your own data center, your servers won't be fully. Idle servers waste energy, so a cloud service provider can charge you less for energy used than you're spending in your own data center. In conclusion, power costs are lower. When you run your own servers, you're looking at up-front capital costs. But in the world of cloud-computing, financing that capital investment is someone else's problem. Sure, if you run the servers yourself, the accounting wizards do their amortization magic which makes it appear that the cost gets spread

over a server's life. But that money still has to come from somewhere, so it's capital that otherwise can't be invested in the business—be it actual money or a line of credit. Moving to the cloud will save you money, not just for your cloud security needs, but for many other types of data center workloads.

To sum up, Bernard Golden, CEO of HyperStratus, draws an insightful analogy between the early adoption of the Internet by business and the growing business use of cloud computing:

"At a certain point in time, the technology vendor community, especially startups, just caught fire about the Internet. They were convinced that, once experienced, no one could avoid adopting their work lives to the Internet. At that same point in time, mainstream IT looked at the Internet with a skeptical eye, focusing on its shortcomings. At that time, I heard statements like ''nobody is going to let their data cross insecure public networks'' and ''Nobody is going to put real business functionality out on the Web.'' Of course, the indisputable benefits of the Internet overwhelmed the dubious responses. As we look back now, the chaos and cynicism is hard to remember, but believe me, it was there—and strong. But those attitudes didn't stand a chance against easy access to information, and I think it's unlikely that a jaundiced view of cloud computing is going to prevail, either." [4]

3.2. A Cybernetic Economy

The size of a company can be measure by the number of contractual relations it creates and by the number managed internally versus externally. But because of the expansion of the wireless Internet, mobile computing and business application services delivered over the Internet, it is becoming easier and less expensive to manage external contractual relationships and transactions. The original organization structure of twentieth century companies was modified and optimized for outside-in communications. This new change is described by the co-chair of the President's Information Technology Advisory Committee:

"Since we can now use technology, the Internet and open standards to begin to automate, standardize and integrate business processes, those transaction costs described by Ronald Coase are dropping precipitously. Consequently, the whole nature of the firm, and what it means to run an efficient business, is going through very extensive changes. These are not easy changes. Not only is there a great deal of innovation required to automate and integrate business processes, but perhaps more important, there are even greater changes in culture required to transform Industrial Age business models to something more appropriate to our Internet era." [5]

One of the companies that changed the old structure is Cisco Systems. In 2002, the organization hit hard in the collapse of the dot-com bubble when their stock went from around $77 a share to around $11. [5] This step was a wake-up call and the company took it as an opportunity. Cisco could learn some lessons and changed its structure, the traditional pyramid-shape corporate hierarchy where all the decisions were made by a small group of senior executives, with a network organization structure which is an efficiency one. Now, the decisions are made by people who have the authority to figure out what is happening and are powered by Internet-based collaborative technologies like blogs, wikis and social media tools.

Cisco's CEO John Chambers makes the case that Cisco's new business model is ''the best possible model for how a large, global business can operate: as a distributed idea engine where leadership emerges organically, unfettered by central command.''

Jeremy Rifkin is a senior lecturer at the Wharton School's Executive Education Program and has spent 10 years as an advisor to the European Union. He is president of the Foundation on Economic

Trends and author of several bestselling books on the impact of scientific and technological changes on the economy, the workforce, and the environment. One of the bestselling books is The Empathic Civilization. The author considers the latest phase of communication and energy regimes as bringing people together, but also creating environmental problems. He calls it an industrial revolution and also the business models are "cybernetic, not linear". Nowadays, business is about accessing the services instead of owning the products. For example, customers buy membership that provides them access to information, instead of purchasing them.

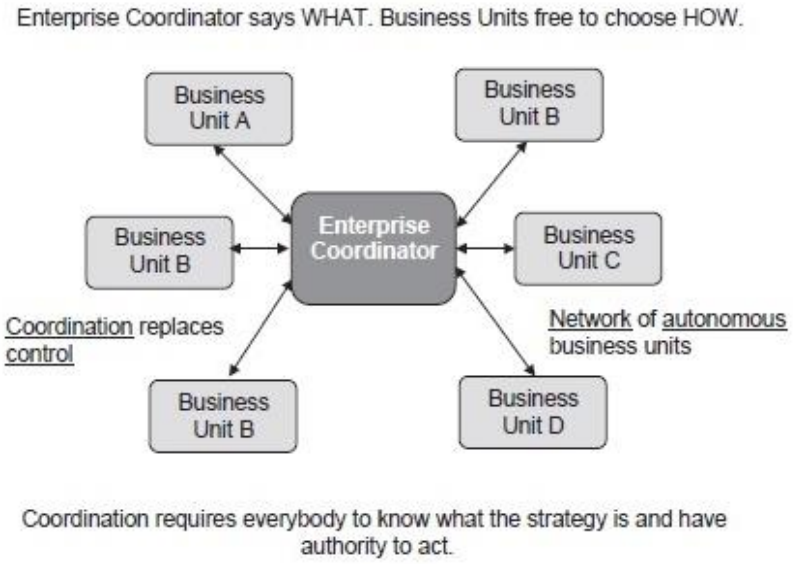


Fig 2. New Organization Structure [5]

The science of cybernetics describes the control and communication processes that work best for network organizations. So familiarity with some basic principles of cybernetics is helpful in exploring how responsive network organizations operate.

4. Famous NoSQL databases

4.1. Key-value Database - Riak

The key/value databases do not concentrate on structure of the data, but on the ability to store and retrieve that data. As a consequences, queries are more efficient and quicker to implement, lending themselves fast scalable applications that are supposed to read and write a wide and variety of data in this big data era.

This key/value type database allow clients to read and write values using a key as follows:

- Get(key), returns the value associated with the provided key.
- Put(key, value), associates the value with the key.
- Multi-get(key1, key2,.., keyN), returns the list of values associated with the list of keys.
- Delete(key), removes the entry for the key from the data store. [6]

Riak is a distributed key-value where values can be anything – from plain text to text, JSON, or XML to images or video clips - all accessible through a simple HTTP interface. It does not matter what type of data you have, the Riak database can store it.

Riak is also fault-tolerant. This means that servers can fluctuate at any moment without giving any notifications or reasons why they have failed. Your cluster continues humming along as servers are

added, removed, or (ideally not) crash. One of the big advantages is that with Riak you do not have to worry about your cluster because, even though, a node failed, is not an emergency and you do not need to solve the problem right away.
However, this flexibility also has disadvantages. Riak lacks robust support for ad hoc queries, and key-value stores, by design, have trouble linking values together (in other words, they have no foreign keys).
Riak is a great choice for datacenters like Amazon that must serve many requests with low latency. If every millisecond spent waiting is a potential customer loss, Riak is hard to beat. It's easy to manage, easy to set up, and can grow with your needs. If you've ever used Amazon Web Services, like SimpleDB or S3, you may notice some similarities in form and function. This is no coincidence. Riak is inspired by Amazon's Dynamo paper. [7]
Riak allows us to control reads and writes into the cluster by altering three values: N, W, and R. N is the number of nodes a write ultimately replicates to, in other words, the number of copies in the cluster. W is the number of nodes that must be successfully written to before a successful response. If W is less than N, a write will be considered successful even while Riak is still copying the value. Finally, R is the number of nodes required to read a value successfully. If R is greater than the number of copies available, the request will fail. [8]
There is a great difference between the relational databases and Riak. The absence of transactions, of the SQL and of the schema leads to difficulties in understanding and usage of the Riak database. Although there are keys, the procedure of linking between buckets is not like a table join. However, some problems can better be solved using Riak because it has the ability to adapt to increase demands of the servers in terms of performance rather than increase in size for larger single servers which, thus, helps to solve the unique scalability problems of the Web. What is more, Riak transmits data bi-directionally when considering the HTTP structure, allowing maximum flexibility for any framework or web-enabled system.

Riak's Strengths:
One of Riak's strengths refers to the fact that it removes the possibility of failure and supports maximum uptime and grow (or shrink) to meet changing demands. It does not matter if your data is complex or not, Riak can store both simple data and allow you to introduce sophisticated information if needed. There are many client libraries for Riak, including Java, Python, Perl, Erlang, Ruby, PHP, .NET, and many others. [9] If you are in the case in which you need more speed than HTTP can handle, communicating via Protobuf [10], might be a better solution because it is a more efficient binary encoding and transport protocol.

Riak's Weaknesses:
There are several drawback when it comes to Riak. There are features things which this type of databases cannot support, for example simple queryability, complex data structures, a rigid schema or the possibility to scale horizontally with your servers.19
One of the major disadvantage about Riak is it the fact that the querying framework remained the same - easy and robust ad hoc. Although, the MapReduce provides strong functionality, there should have been more built-in URL-based or other PUT query actions. Finally, if Erlang is not your favorite programming language, there are few limitations when using JavaScript, such as the unavailability of post-commit or the execution of MapReduce is slow.

4.2. Column-Oriented Database – Hbase

HBase is a column-oriented database management system that runs on top of HDFS. It is perfect for sparse data sets, which are common in many big data use cases. HBase does not support SQL, which

is a structured query language. HBase applications are written in Java much like a typical MapReduce application. HBase does support writing applications in Avro, REST, and Thrift.
An HBase system contains a set of tables. Each table contains rows and columns, much like a traditional database. Each table must have an element defined as a Primary Key, and all access attempts to HBase tables must use this Primary Key. An HBase column represents an attribute of an object. In fact, HBase allows for many attributes to be grouped together into what are known as column families, such that the elements of a column family are all stored together. This is different from a row-oriented relational database, where all the columns of a given row are stored together. With HBase you must predefine the table schema and specify the column families. [12]

Key Features of HBase:

- Scale-out Architecture - add servers to increase capacity
- Automatic Sharding - Transparently and efficiently scale out your data across machines in the cluster
- Full Consistency - Guard against node failures or simultaneous writes to the same record
- Active-active Replication - Stream data across locations for disaster recovery and data protection
- High Availability - Multiple master nodes ensure continuous access to data
- Full-text, Faceted Search - Give non-technical users and your applications a familiar yet powerful, interactive search experience [13]
- Security - Secure table and column family-level access via Kerberos
- SQL Access - Query data interactively with Cloudera Impala and for batch processing with Apache Hive

HBase's Strengths:
Noteworthy features of HBase include a robust scale-out architecture and built-in versioning and compression capabilities. For example, keeping the version history of wiki pages is a crucial feature for policing and maintenance. By choosing HBase, we don't have to implement page history—we get it for free.
Talking about performance, HBase is meant to scale out. If you work with large amounts of data, measured in many gigabytes or terabytes, HBase may be for you.

HBase's Weaknesses:
Although HBase is designed to scale out, it doesn't scale down. The Hbase community seems to agree that five nodes is the minimum number you'll want to use. Because it's designed to be big, it can also be harder to manage. Solving small problems isn't what HBase is about.
HBase doesn't offer any sorting or indexing possibilities except the row keys. Rows are kept in sorted order by their row keys, but no such sorting is done on any other field, such as column names and values. So, if you want to find rows by something other than their key, you need to scan the table or maintain your own index. Another missing concept is datatypes. There is no distinction between, say, an integer value, a string, and a date. They're all bytes to HBase, so it's up to your application to interpret the bytes.[14]

4.3. Document-Oriented Database - MongoDB

MongoDB is in many ways like a power drill. Your ability to complete a task is framed largely by the components you choose to use (from drill bits of varying size to sander adapters). MongoDB's strength lies in versatility, power, ease of use, and ability to handle jobs both large and small. Although it's a much newer invention than the hammer, it is

increasingly a tool builders reach for quite often.
First publicly released in 2009, MongoDB is a rising star in the NoSQL world.
It was designed as a scalable database—the name Mongo comes from "humongous"—with performance and easy data access as core design goals. It is a document database, which allows data to persist in a nested state, and importantly, it can query that nested data in an ad hoc fashion. It enforces no schema (similar to Riak but unlike Postgres), so documents can optionally contain fields or types that no other document in the collection contains.23
MongoDB Features: focuses on flexibility, power, speed, and ease of use:
Flexibility - MongoDB stores data in JSON documents (which we serialize to BSON). JSON provides a rich data model that seamlessly maps to native programming language types.
Power - MongoDB provides a lot of the features such as secondary indexes, dynamic queries, sorting, rich updates, upserts (update if document exists, insert if it doesn't), and easy aggregation.
Speed/Scaling - By keeping related data together in documents, queries can be much faster than in a relational database where related data is separated into multiple tables and then needs to be joined later. MongoDB also makes it easy to scale out your database. Autosharding allows you to scale your cluster linearly by adding more machines. It is possible to increase capacity without any downtime, which is very important on the web when load can increase suddenly and bringing down the website for extended maintenance can cost your business large amounts of revenue.
Ease of use - MongoDB works hard to be very easy to install, configure, maintain, and use. To this end, MongoDB provides few configuration options, and instead tries to automatically do the "right thing" whenever possible. This means that MongoDB works right out of the box, and you can dive right into developing your application, instead of spending a lot of time fine-tuning obscure database configurations. [24]

Mongo's Strengths
Mongo's primary strength lies in its ability to handle huge amounts of data (and huge amounts of requests) by replication and horizontal scaling. But it also has an added benefit of a very flexible data model.
Finally, MongoDB was built to be easy to use. You may have noticed the similarity between Mongo commands and SQL database concepts (minus the server-side joins). This is not by accident and is one reason Mongo is gaining so much mind share from former object-relational model (ORM) users. It's different enough to scratch a lot of developer itches but not so different it becomes a wholly different and scary monster.

Mongo's Weaknesses
How Mongo encourages denormalization of schemas (by not having any) might be a bit too much for some guys to take. It can be dangerous to insert any old value of any type into any collection. A single type can cause hours of headache if you don't think to look at field names and collection names as a possible culprit. Mongo's flexibility is generally not important if your data model is already fairly mature and locked down.
Because Mongo is focused on large datasets, it works best in large clusters, which can require some effort to design and manage. Unlike Riak, where adding new nodes is transparent and relatively painless for operations, setting up a Mongo cluster requires a little more forethought. [15]

4.4. Graph Database – Neo4J

Neo4j is a new type of NoSQL datastore called a graph database. As the name implies, the data stored look like a graph (in the mathematical sense). You can refer

at the Neo4J database using the term "whiteboard friendly," which means that it does not matter what you draw, there might be boxes or lines, the drawings can be stored in Neo4J. This type of database concentrates more on the relationships between values rather than on the commonalities among sets of values (such as collections of documents or tables of row). [16] As a matter of fact the data can be stored in a natural and straightforward way. Being small in size, it is possible to embed the Neo4j into approximately any application. Nonetheless it has the capability to store tens of billions of nodes and as many edges. And with its cluster support with master-slave replication across many servers, it can handle most any sized problem you can throw at it.

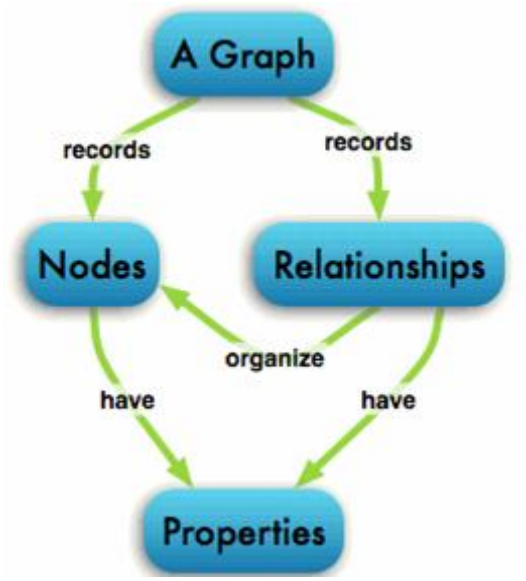


Fig 3. The structure of a graph database [15]

Neo4j's Strengths

Neo4j is one of the finest examples of open source graph databases. Graph databases might be considered the answer for unstructured data, in many ways even more so than document datastores. Even though Neo4j has no type and no schema Neo4j, the constraints it puts on how data is related are essential. At the moment, Neo4j has the ability to support 34.4 billion nodes and the same number as many relationships, which is sufficient for most uses (For example, Neo4j could hold more than 42 nodes for each of Facebook's 800 million users in a single graph [17]).
Beyond ease of use, Neo4j is fast. In spite of join operations in relational databases or map-reduce operations in other databases, the benefit is that graph traversals are constant time. The majority of the databases usually join the values in bulk and filter the desired results. The graph databases, however, act just like the data is only a node step away and it is not compulsory to know how large a graph becomes, moving from node A to node B is always a one-step if they share a relationship.

Neo4j's Weaknesses

There are also a few drawbacks for Neo4j. One of them is the fact that edges in Neo4j cannot direct a vertex back on itself. There is also a problem with nomenclature because it is called node rather than vertex, and relationship rather than edge, thus, adding more complexity when communicating. HA can only replicate a full graph to other servers, even though it is excellent at replication. Finally, if you are in search of a business-friendly open source license (like MIT), Neo4j may not suitable for your company.

Table 1. Comparison between Riak, HBase and Mongo DB

Databases name / Features	Riak	HBase	Mongo DB
Data Model	*Riak stores key/value pairs in a higher level namespace called a bucket.	*HBase stores data in a pre-defined column family format; *Data in HBase is sorted, sparse, and physically grouped by column family	*MongoDB's data format is BSON (binary equivalent to JSON) stored as documents (self-contained records with no intrinsic relationships). Documents in MongoDB may store any of the defined BSON types and are grouped in collections.
Storage Model	*Riak has a modular, extensible local storage system which features pluggable backend stores designed to fit a variety of use cases. The default Riak backend store is Bitcask.	*Hadoop Distributed File System (HDFS) is the storage system used by HBase. Data is stored in MemStores and StoreFiles, where data is streamed to disk (implemented via HFiles, a format based on BigTable's SSTable). Implementations generally use the native JVM-managed I/O file stream.	*MongoDB's default storage system is the Memory-Mapped Storage Engine. It uses memory mapped files for all disk I/O. It is the responsibility of the OS to manage flushing data to disk and paging data in and out.
Query Types and Query-ability	*There are currently four ways to query Riak. ▪ Primary key operations (GET, PUT, DELETE, UPDATE) ▪ MapReduce ▪ Secondary Indexes ▪ Riak Search ▪ Comparing MapReduce, Search, and Secondary Indexes	*HBase has two query options: looking up values by getting/scanning through ordered keys (optionally filtering out values or using a secondary index), or by using Hadoop to perform MapReduce.	*MongoDB has a query interface that has some similarities to relational databases, including secondary indexes that can be derived from the stored documents. MongoDB also has a facilities for performing MapReduce queries and ad-hoc queries on documents. Hadoop support is available, too.

Concurrency	*In Riak, any node in the cluster can coordinate a read/write operation for any other node. Riak stresses availability for writes and reads, and puts the burden of resolution on the client at read time.	*HBase guarantees write atomicity and locks per row. HBase has also recently added multi-action and multi-row local transactions (though you cannot mix read/write actions).	*MongoDB exhibits strong consistency. Eventually consistent reads can be accomplished via secondaries. A MongoDB cluster (with auto-sharding and replication) has a master server at a given point in time for each shard.
Replication	*Riak's replication system is heavily influenced by the Dynamo Paper and Dr. Eric Brewer's CAP Theorem. Riak uses consistent hashing to replicate and distribute N copies of each value around a Riak cluster composed of any number of physical machines. Under the hood, Riak uses virtual nodes to handle the distribution and dynamic rebalancing of data, thus decoupling the data distribution from physical assets. *The Riak APIs expose tunable consistency and availability parameters that let you select which level of configuration is best for your use case. Replication is configurable at the bucket level when first storing data in Riak. Subsequent reads and writes to that data can have request-level parameters.	*HBase supports in-cluster and between-cluster replication. In-cluster replication is handled by HDFS and replicates underlying data files according to Hadoop's settings. Between-cluster replicates by an eventually consistent master/slave push, or more recently added (experimental) master/master and cyclic (where each node plays the role of master and slave) replication.	*MongoDB relies on locks for consistency. As of version 2.2, MongoDB has a DB Level Lock for all operations.

Scaling Out and In	*Riak allows you to elastically grow and shrink your cluster while evenly balancing the load on each machine. No node in Riak is special or has any particular role. In other words, all nodes are masterless. When you add a physical machine to Riak, the cluster is made aware of its membership via gossiping of ring state. Once it's a member of the ring, it's assigned an equal percentage of the partitions and subsequently takes ownership of the data belonging to those partitions. The process for removing a machine is the inverse of this. Riak also ships with a comprehensive suite of command line tools to help make node operations simple and straightforward.	*HBase shards by way or regions that automatically split and redistribute growing data. A crash on a region requires crash recovery. HBase can be made to scale in with some intervention on the part of the developer or DBA.	*Mongo relies on sharding for scaling out. This involves designating a certain server to hold certain chunks of the data as the data set grows. *To scale in, MongoDB has support for removing shards from your database.
Multi-Datacenter Replication and Awareness	*Riak features two distinct types of replication. Users can replicate to any number of nodes in one cluster (which is usually contained within one datacenter over a LAN) using the Apache 2.0 licensed database. Riak Enterprise, Basho's commercial extension to Riak, is required for Multi-Datacenter deployments (meaning the ability to run active Riak clusters in N datacenters).	*HBase shards by way of regions, that themselves may be replicated across multiple datacenters.	*MongoDB can be configured to run in multiple datacenters via various options.

Graphical Monitoring/Admin Console	*Riak ships with Riak Control, an open source graphical console for monitoring and managing Riak clusters	*HBase has a few community supported graphical tools, and a command-line admin console.	*MongoDB does not ship with a graphical monitoring/admin console. However, several community projects have developed graphical monitoring/admin programs. *10Gen offers a hosted monitoring service. [18]

5. Conclusions

Even if not many people know about cloud computing, it became popular in the latest years. Also, famous companies like IBM adopted or created their own cloud. The major advantage is probably the effects on costs. You can save money, time, even work from home. But there are also some disadvantages like security. If your cloud architecture it is weak you can lose valuable data and the company may lose money.

"If you think you've seen this movie before, you are right. Cloud computing is based on the time-sharing model we leveraged years ago before we could afford our own computers. The idea is to share computing power among many companies and people, thereby reducing the cost of that computing power to those who leverage it. The value of time share and the core value of cloud computing are pretty much the same, only the resources these days are much better and more cost effective." [21]

When creating a business strategy, the main reasons why you should choose NoSQL databases is for better performance, scalability, &flexibility. In this modern era, owning a business sometimes could put you in the position to get in touch with clients and provide them with several applications.

These apps developed in-house not only strengthen the relationship with the customer, but also have the role to protect data and keep the company gain access easily and better management the incoming data.(for example, the Adventure Works Shopping application [22])There are some megatrends that had a huge impact on these applications' needs. One of them is the fact that the number of users that applications must support is growing continuously. Moreover, elevated users' expectations for how applications should perform increase proportionally with the number of users. Secondly, the shift in applications should also consider the increase in the volume and the variety of data available. [23]

In conclusion, the main reasons why the usage of NoSQL technology is increasing among companies and enterprises are because it offers data management capabilities that meet the needs of modern applications, including:

- Better application development productivity through a more flexible data model.
- The ability to scale out dynamically and cost effectively to support more users and big data.
- Improved performance that satisfies user expectations for highly responsive applications and allows more complex processing of data.

NoSQL is increasingly seen as a viable alternative to relational databases, and should be considered especially for interactive web and mobile applications.

References

[1] "How the Next Generation of Databases Could Solve Your Problems", htttps://datafloq.com/read/generation-database-solve-problems/139

[2] "Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement", Eric Redmond, Jim R. Wilsonreleased in 2012 by Pragmatic Programmers, LLC

[3] "Data Access for Highly-Scalable Solutions: Using SQL, NoSQL, and Polyglot Persistence", Douglas McMurtry,Andrew Oakley, John Sharp, Mani Subramanian,, Hanz Zhang, Microsoft 2013

[4] Bernard Golden, ''McKinsey Cloud Computing Report Conclusions Don't Add Up,'' CIO.com (April 27, 2009), www.cio.com/article/490770/ McKinsey Cloud Computing Report Conclusions Don t Add Up.

[5] Irving Wladawsky Berger, ''The 'Outside In' Enterprise,'' October 10, 2005, blog post http://irvingwb.typepad.com/blog/2005/10/the outsidein e.html.

[6] 3 Pillar Global, Exploring the Different Types of NoSQL Databases Part II http://www.3pillarglobal.com/insights/exploring-the-different-types-of-nosql-databases

[7] "Dynamo: Amazon's Highly Available Key-value Store",http://allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf

[8] "Dynamo: Amazon's Highly Available Key-value Store",http://allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf

[9] Basho, Riak Distributed Database http://basho.com/riak/

[10] Protocol Buffer http://code.google.com/p/protobuf/

[11] "Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement", Eric Redmond, Jim R. Wilson

[12] IBM, Hadoop http://www-01.ibm.com/software/data/infosphere/hadoop/hbase/

[13] Cloudera, CDH http://www.cloudera.com/content/cloudera/en/products-and-services/cdh/hbase.html

[14] "Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement", Eric Redmond, Jim R. Wilson

[15] "Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement", Eric Redmond, Jim R. Wilson

[16] "Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement", Eric Redmond, Jim R. Wilson

[17] "Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement", Eric Redmond, Jim R. Wilson

[18] Basho, Riak Comparisons http://docs.basho.com/riak/1.3.1/references/appendices/comparisons/

[19] Joe McKendrick, 10 Quotes on Cloud Computing That Really Say it All, Forbes Contribuitors, March 2013 http://www.forbes.com/sites/joemckendrick/2013/03/24/10-quotes-on-cloud-computing-that-really-say-it-all/

[20] Douglas McMurtry, Andrew Oakley, John Sharp, Mani Subramanian, Hanz Zhang, "Data Access for Highly-Scalable Solutions: Using SQL, NoSQL, and Polyglot Persistence" released in 2013 by Microsoft

[21] David Linthicum**,** *Cloud Computing and SOA Convergence in Your Enterprise: A Step-by-Step Guide*

[22] Douglas McMurtry, Andrew Oakley, John Sharp, Mani Subramanian, Hanz Zhang, "Data Access for Highly-Scalable Solutions: Using SQL, NoSQL, and Polyglot Persistence" released in 2013 by Microsoft

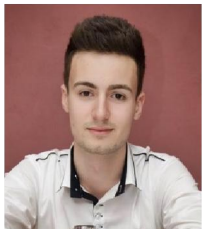
[23] CouchBase, Why NoSQL? http://www.couchbase.com/nosql-resources/what-is-no-sql

[24] MongoDB, Introduction to MongoDB http://www.mongodb.org/about/introduction

Bogdan NEDELCU graduated Computer Science at Politehnica University of Bucharest in 2011. In 2013, he graduated the master program "Engineering and Business Management Systems" at Politehnica University of Bucharest. At present he is studying for the doctor's degree at the Academy of Economic Studies from Bucharest.

Andreea Maria IONESCU is currently pursuing an undergraduate degree at the Bucharest University of Economic Studies. The field of study is Cybernetics, Statistics and Economic Informatics.

Ana-Maria IONESCU is studying Informatics and Economics at Academy of Economic Studies from Bucharest since 2013. She is volunteer at SiSC, a NGO in faculty.



Alexandru George VASILE studies at Academy of Economic Studies from Bucharest, Faculty of Cybernetics, Statistics and Economic Informatics since 2013.

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.

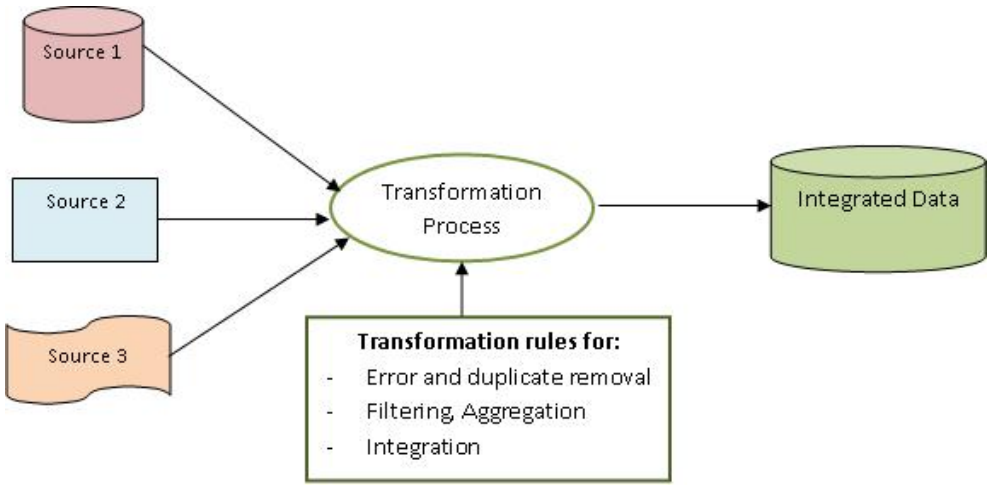


Fig.1 - Transforming data for integration (adapted after [6])

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).

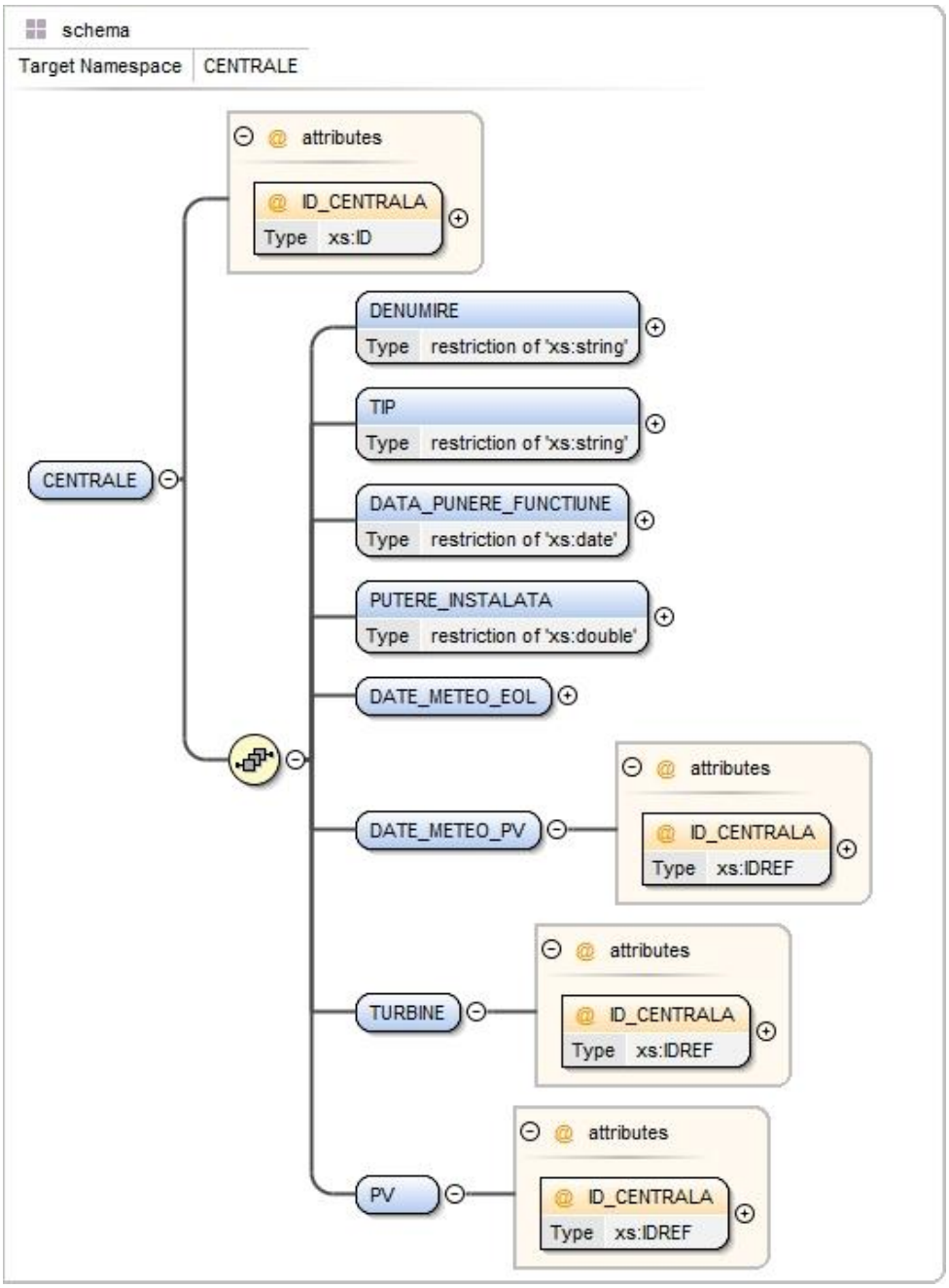


Fig.2 – The XSD schema for CENTRALE

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

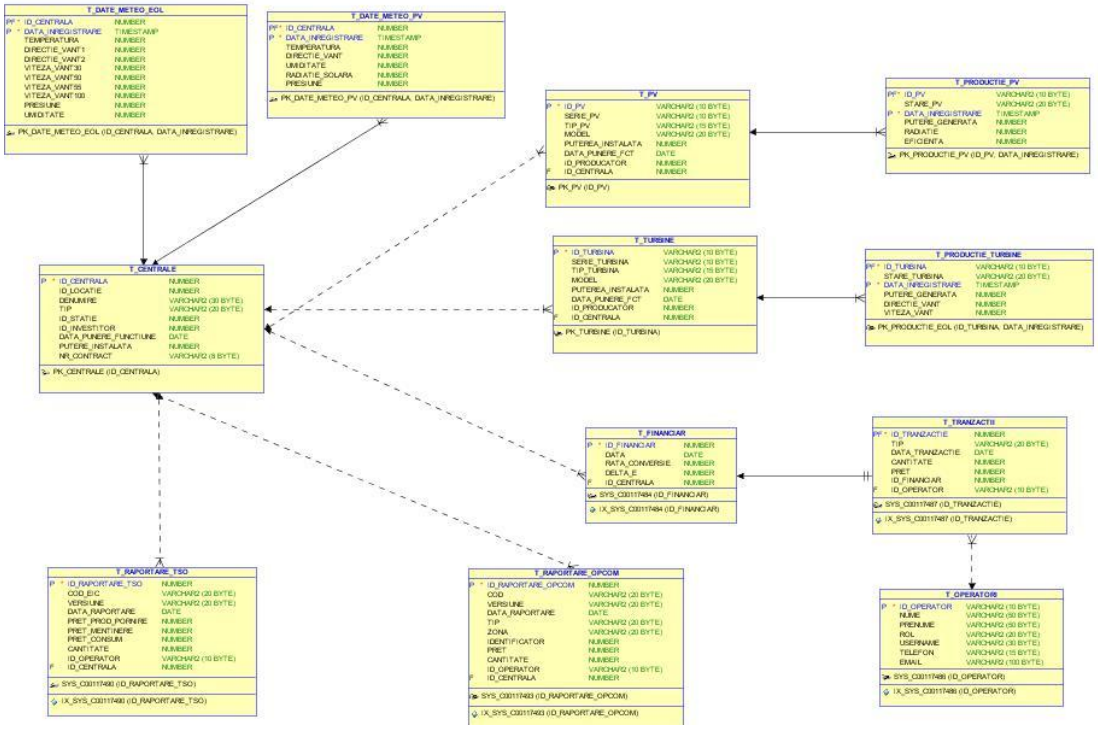


Fig. 3 – The database schema for RES producers

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.

Cloud Computing and Business Intelligence

Alexandru Adrian TOLE
Romanian – American University, Bucharest, Romania
adrian.tole@yahoo.com

The complexity of data resulting from business process is becoming overwhelming for the systems that don't use shared resources. Many aspects of the business process must be recorded and analysed in a short period of time with no errors at all. In order to obtain these results, so that management and other departments know what their next decision/job will be, there must be a continuous exchange and processing of information.
"Cloud Computing" is the solution to overcome the problem of processing large amounts of data. By using this technology organizations have the benefit of using shared resources from various systems that are able to face large amount of data processing. This benefits does not only resume to a high performance system but also the costs of using such architecture are much lower.

Keywords: *Cloud Computing, Business Intelligence, shared resources, hardware architecture, SaaS, PaaS, IaaS*

Introduction

In the past decade Business Intelligence systems have evolved to a level that software solutions overwhelmed traditional hardware architecture. By traditional I refer to the simple model that organizations used to store their data in-house on a server (or multiple ones) and process it with the existing software. Business Intelligence solutions offer these days a complex overview of the business process. They integrate software that is able to process, store and analyze data from various departments. A Business solution not only integrates specialized software but also consists in a custom built hardware architecture that is able to handle the amount of processes that result from software operations.
In order to adapt hardware architecture to software solutions, the costs of building and maintaining such system can get extremely expensive. In this case, financial factors are decisive in whether the benefits of such investment worth it comparing to the costs of it. Ideal in this case is to implement Business Intelligence software solutions into a shared hardware resources environment. To accomplish this, "Cloud Computing" solutions where developed so that the costs of implementing custom software solutions that helps the business process will be lower.
The Cloud Computing environment is based on a simple but effective principle: resource sharing. This principle helps companies and other organizations to share complex system architecture in order to implement the solutions that they need. This practice raises an important ethical matter that refers to whether this sort of solution provides privacy or not. From a theoretical point of view organizations that share the same Cloud environment can't access each other data because the software implemented is designed not only to handle the processes that was designed for, but is built to ensure a certain level of privacy regarding the data that is working with.
Cloud solutions providers offer maintenance for their clients. This service can cover both hardware and software problems that can appear. For the clients, this ensures a continuous workflow that is reflected into the business process.
By using Cloud Computing, organizations experience cost-effective solutions that will not put pressure on their financial planning regarding investments comparing to the

classical model where companies invest in datacenters and the maintenance of those.
In almost every aspect, using a Cloud environment to implement software solutions is a gain to organizations.

2. Cloud Computing

Cloud Computing, simply described, represents a pool of systems that are connected into a network and provide a scalable infrastructure so that software systems and data can take use of it. By using this type of solution the cost of implementing software solutions and storage of data is reduced significantly.
To have a better understanding of what Cloud Computing represents we have to compare it to traditional concepts such as grid computing.
Grid Computing (**Fig. 1.** Grid Computing –> Simple model) represents the use of resources from many computers connected in a network to solve a single problem at the same time. The problem with grid computing is that if one system fails there is a high risk of others to fail. Cloud Computing (**Fig. 2.** Cloud Computing) tries to overcome this issue by using all the systems in the network as a whole so that if one system fails, another will automatically replace it.

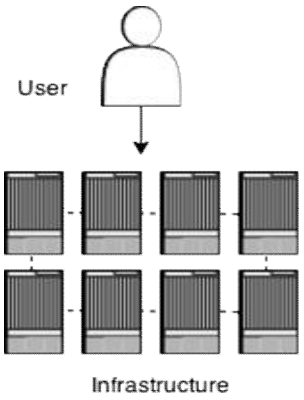


Fig. 1. Grid Computing –> Simple model

Grid computing can be compared to a "super computer" which consists of multiple systems that are connected into a network so the resources can be used in order to handle a single problem. The systems that take part in such infrastructure can be scattered around the globe.
Cloud Computing can be described as an evolved grid computing infrastructure.

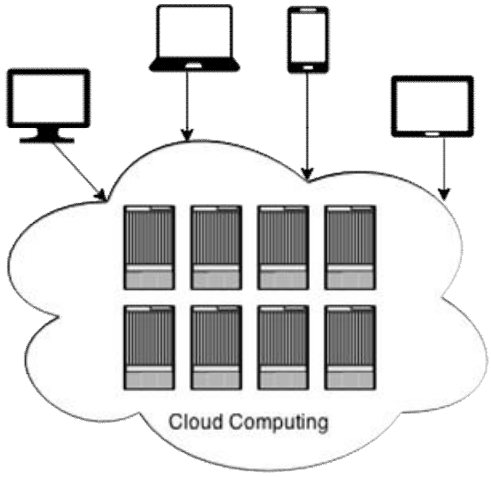


Fig. 2. Cloud Computing

Cloud resources can be reallocated on demand to fulfill the client needs. For example, some countries don't allow storing of user data outside their border. In order to accomplish that, cloud providers can create an infrastructure that can reside in that country. Also, cloud infrastructure can be extremely flexible and can offer for example multiple time zones to work with. If the cloud solution is located in Canada it can be used by European clients because it can adapt to different time zones.

2.1 Cloud Computing – Characteristics

Cloud solutions can be described as having high scalability, agility, high availability and reliability and multi-sharing.[1]
Regarding the "High scalability" characteristic, this refers to enabling the use of resources for a large pool of users that have different needs. The agility characteristic is the one that describes the response time of the system regarding the tasks that are submitted by users. In this case, the response time is very short considering the complexity of the infrastructure.
A cloud infrastructure is also highly available and reliable. Thus, the

organizations/users that take advantage of the speed and scalability of the system can also enjoy a very high rate of availability for their implemented solution. Also, cloud infrastructure is very reliable because is continuously adapted to the user needs. The multi-sharing characteristic describes the defining part of cloud computing. This is the main purpose of this technology: resource sharing for the users that are using it.

2.2 Cloud Computing – Models

Cloud computing providers offer service according to client needs. Based on this, the implemented cloud solutions are cost-effective and reliable. There are three Cloud Computing models (**Fig. 3.** Cloud Computing -> Models):

- **Infrastructure as a service (IaaS)**

The IaaS providers offer physical or virtual machines that are able to fulfill customer needs to implement software solutions on it. It also provides various resources such as firewalls, load balancers, software solutions and many more. This type of service offers a great advantage to clients that need a solid and flexible infrastructure. It can also provide the security that they need. This model helps users deploy their own software at reduced costs. Another advantage of using IaaS is that users can deploy and maintain the operating system of the infrastructure. Thus, the provider is not forcing the clients to use a specific OS. Usually the clients pay only for the resources allocated for their implemented solution, without worrying for hardware maintenance.

- **Platform as a service (PaaS)**

This type of model has installed already an operating system, a programming language, a web server, etc. This is probably the most common type of cloud that is used because it facilitates the implementation and testing of software solutions. This model also provides the needed resources for an application to run. The resources are automatically allocated so that the user does not need to do that manualy.

- **Software as a service (SaaS)**

The SaaS model is described as a pay-per-use service where the providers offer clients a fully configured (hardware and software) solution. To access this, clients must pay a subscription fee. The advantage of using this system is that clients don't have to worry about any maintenance, hardware or software. The SaaS provider is taking care of that for the client.

For a better understanding, SaaS can provide access for a company to a BI solution. For this, the company will pay a monthly or yearly subscription that will depend on the customization of the BI solution and/or the resources allocated for that company.

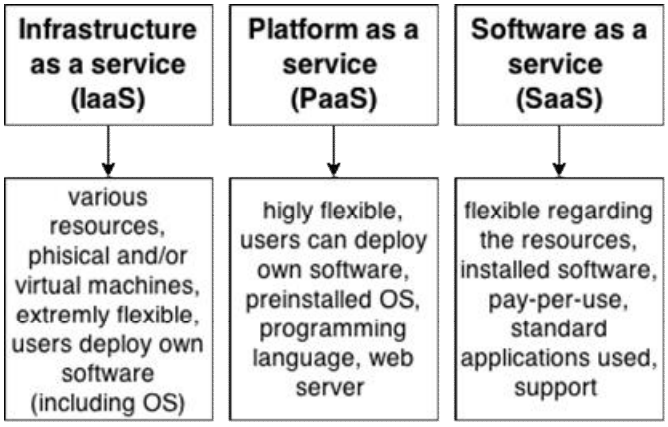


Fig. 3. Cloud Computing -> Models

There are three main types of Cloud deployment (**Table 1.** Cloud Computing -> Deployment):

- **Public Cloud** – which generally means that is open for public use. Users can store data on it without having the privacy that a secure network provides. This type of Cloud can also be offered as free to use. The main concern is privacy because providers don't guarantee it.

- **Private Cloud** – this is an infrastructure that is used by one user/company. This can reside whether internally the company building (data center) or externally (provider). This is a highly secured Cloud and is preferred by most companies because, when is resided externally, maintenance can be offered by the provider. Another advantage is the resources allocated to a single client. By doing this, the infrastructure is custom built to offer great performance and stability.

- **Hybrid Cloud** – this is a combination of private and public Cloud. By doing this, users can specify which data resides on public Cloud and which on private. The advantage of using Hybrid Cloud is that the expenses can be reduced. The data is aggregated from both sources in order to provide the results that users needed.

Table 1. Cloud Computing -> Deployment

	Public Cloud	*Private Cloud*	*Hybrid Cloud*
Cost	free	depends on resources allocated	lower than Private Cloud
Security	low	high	moderate
Flexibility	high	very high	very high
Resources	high	high – based on needs	high – based on needs

There is also a fourth deployment type of Cloud which is called **Community Cloud**. This is refers to a Cloud infrastructure that is used by a group of organizations that share data on it. Such type of infrastructure is used by government institutions, universities from a region or even country, suppliers, etc.
The security provided on this type of Cloud is similar to that on a Private Cloud because in reality is a Private Cloud that is shared by numerous entities that share the same purpose.

2.3 Cloud Computing – Obstacles and Opportunities

The obstacles of Cloud Computing can be cataloged in terms of adoption, growth, policy and business.[2]
One of the first obstacles that need to be overcome in Cloud Computing is data privacy. To many, keeping data on a shared infrastructure raises a privacy issue. Even thought that data can only be accessed with specific software under specific security circumstances (password protected, encryption, etc.), this is not enough.
The cloud user takes responsibility for the software security that he implemented in the Cloud environment. This is a problem because in a closed and controlled infrastructure (such as intranet), the user can only access the system from within the network. A solution like this can provide

enough security because it can't be accessed from outside the trusted network. Cloud is not able to do that, even though the service provider is ensuring the security of the infrastructure by adding firewalls and other hardware or software solutions that prevents outsiders from stealing data. In most cases this is efficient but there are situations in which Cloud users are victims of phishing sites and/or emails. For example if a company uses a customized BI implemented in Cloud to which its employees login solely with username and password than if anyone that have those credentials can login into the system. Unfortunately users are not always careful and whether they use easy or common passwords or they complete phishing forms.

In this case, Cloud provider can't be held responsible. If the security of the Cloud infrastructure is affected, data from many clients may be exposed to risk.

Another security issue is protecting the user from the provider. This means that the provider can access all the data that is stored on the infrastructure and do what he wants with it, although is illegal. To protect clients from that they can use software that allows them to encrypt sensitive data. By doing this, even if the provider has access to data he can't use it since he is unable to read it. By encrypting data the user might experience some issues in the response-time of the implemented software because on one side the information is encrypted and transmitted to cloud infrastructure (**Fig. 4.** Cloud Computing -> Data encryption). This process must be reversed in order for the information to be read by the user.

Another obstacle in the use of Cloud computing is software licensing. Software providers usually sell their software with a license/machine. Since in Cloud there is a pool of machines that are working at the same time this type of license can't be applied. Cloud providers are using, in general, open source software. Software developers are starting to change their license agreement so that the software they create can be used in a Cloud environment.

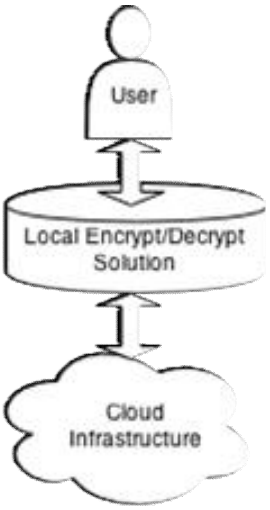


Fig. 4. Cloud Computing -> Data encryption

2.4 Cloud Computing – Conclusions

Using a cloud infrastructure can have a lot of benefits in terms of costs. From small sized business to a large corporation, cloud computing can be the best solution for implementing commercial or open source software. Another upside of using cloud services is that this type of infrastructure is highly scalable which means that will always fulfill your software need for resources. Also, by using an external cloud system (from a provider) will keep organizations from spending money on maintenance and also on data redundancy (backup). The backup solution is very useful in case of system failure. This will allow the user to recover his data with no workflow interruption.

Cloud computing provides great flexibility regarding software implementation. Whether the user need an infrastructure that provides only hardware capabilities, with no software installed on it or he needs BI software ready to use in cloud, a cloud provider can offer that.

Unfortunately, when information resides on a hard drive that the user does not posses, security issues might appear. Most of these concerns gravitates around one question: *Are your data private?* The answer to this

question is not only a *yes* or *no*, instead is another question: *How important are those data to you?* By answering this question plenty of software and hardware solutions can be used so that sensitive information won't go into wrong hands. The problem with high security is that it involves high costs. Also, implementing security solutions might generate additional cost for local hardware implementation. It can also affect the response time of data transactions, independent from the cloud infrastructure.

Software licensing is another issue that software companies must handle. They have to allow their clients to use the software in a cloud environment. Changing the license agreement by making it independent from the number of machines that is used on might be a solution. Also, they can sale their products for two types of environments: one that is dependent on the number of machines that is being used on and another that can be called "Cloud licensing" and will need a proper verification by the vendor.

The need for using cloud computing is growing and will soon become imperative. Giving the fact that information and processes grow exponentially each year, sharing resources is the answer that can solve many problems.

3. Cloud BI or integrating Business Intelligence solutions into Cloud

3.1 What is Business Intelligence?

The term business intelligence or simply BI represents a set of software tools and hardware infrastructure that has the purpose to offer support in the strategic planning of a corporation. BI systems provides for the company solutions to gather, store and analyze data in order to help in decision-making.

Most companies gather large amount of data from different departments using numerous software solutions. Is hard to put all that data together and use it in the decision-making process if is not filtered previously. Here is where BI solutions help. These solutions can offer a quick and readable overview of data from different departments. The data can be easily analyzed and can be used in the decision-making process.

BI software is designed to extract important data from raw data in order to reveal insights that can help a manager (and not only) take faster and accurate decisions. Business intelligence software uses features like data mining, statistics and predictive analytics that can reveal certain patterns.

For a complete usage of BI capabilities, BI must be flexible and must provide various facilities for employees, teams and total organization levels.[3] This means that the software provided must be able to grant access for all the employees to structured data that concerns their line of work.

BI solution is helpful for remaining competitive. Having the necessary information will help a company to assess its weaknesses and strengths. Also is helpful to make correct predictions so that management will take decisions that will help the business to reach its goal (**Fig. 5.** Business Intelligence -> Business goal)

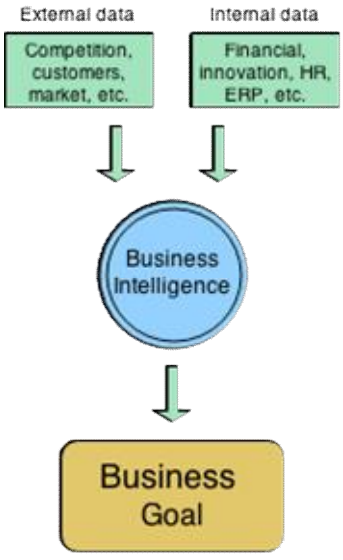


Fig. 5. Business Intelligence -> Business goal

Business Intelligence software is able to provide detailed reports to the management based on the information collected from various sources, internal or external.

Implementing a successful BI solution is also related to information relevancy that is processed. Working with raw data is time consuming and extremely hard. To avoid analyzing useless data the perfect solution is to use custom software for each department that is involved in the business process. After this objective is achieved, the BI solution implemented will provide accurate results.

In deploying a BIS there are many risks involved: system design, data quality and technology obsolescence.[4] The system design of BI plays an important role in order to obtain all the information necessary for the decision-making process. If there is one area that is not covered by the BI processing and analytics tools this might result in an erroneous report that can jeopardize the reach of the business objective.

Also, data quality is of high value. As stated before, the quality of information determines the quality of the results. This is also important for predictive reports.

Technology obsolescence is an obstacle in the path of reaching the information and processing it. While a company grows the amount and complexity of data that needs to be processed becomes so big that the hardware architecture which was designed five years ago is now obsolete. In this case the hardware system is not the only one affected by the growth of a company. The BI system will start having flaws.

3.2 Integrating Business Intelligence software into Cloud

Integrating a BI into a cloud environment will overcome the technology obsolescence problem. By doing this scalability will be achieved. In this case, no matter how much the data complexity and amount from a company will evolve a BI integrated into a cloud infrastructure can handle it.

Integrating BI into Cloud is an advantage to a company not only for its scalability but also for elasticity and ease of use. By elasticity I refer to the ability of a BI to continuously absorb information from newly added software. For example, a company decides at a point that it needs an online helpdesk for its customers. This can be implemented in cloud and integrated into BI processes in a very short time, with no need for purchasing additional hardware, such as servers, so that helpdesk software will run on. The ease of use of a Cloud BI is determined also by the ease of access of BI software on various devices. Implementing a web solution for helping the decision-making process has the advantage of being able to access it whether the user resides in the company or anywhere else. This will keep users that are on the move permanently informed. This provides also accessibility. The BI can be accessed on any web browser.

Shimaa Ouf and Mona Nasr suggest that BI solutions can be moved to cloud using PaaS. [5] By using platform-as-a-service model a BI solution can be implemented in a short period of time. The solution can take advantage of the PaaS preinstalled software such as relational database software. Users will not engage into a continuous software managing and patching activity. The database platform that is present on cloud is scalable so it will meet the needs of resources available to handle an organization information volume.

Also, another advantage that Cloud provides to BI integration is reduced costs. Sharing resources will result in a lower cost per machine. Also, hardware maintenance will be handled by the cloud provider as well as the implementation of hardware firewalls and load balancers to handle the traffic towards the data center.

Availability of BI solutions is another gain for companies that chose to integrate it into Cloud. By availability I refer to the so called “up-time” of a service. Cloud infrastructure is capable of working regardless of a system

failure. Also, cloud providers ensure their internet connectivity by one or more alternate connection so if one fails, another will take over the traffic.

Pros and cons regarding the integration of a BI solution into Cloud

Pros:

- Scalability and elasticity;
- Reduced costs;
- Ease of use and access;
- Cloud relational database;
- Availability;
- Hardware maintenance.

Cons:

- Privacy;
- Government regulations (where applied).

As stated before in the Cloud Computing section of this article, privacy remains an issue. BI solutions are not an exception. The security provided by BI solutions is only at an UI (user interface) level. The data stored on Cloud database is exposed to the provider.

Government regulations are, in some cases, a barrier in the migration of BI solutions of companies to a Cloud infrastructure outside the border. This represents a downside in terms of cloud computing expenses. The Cloud providers that are located in the same country with an organization might have higher costs than foreign providers.

Eumir P. Reyes describes a basic BI architecture based on cloud infrastructure in his paper "A Systems Thinking Approach to Business Intelligence Solutions Based on Cloud Computing". (**Fig. 6.** Cloud BI or integrating Business Intelligence solutions into Cloud -> Basic Architecture).

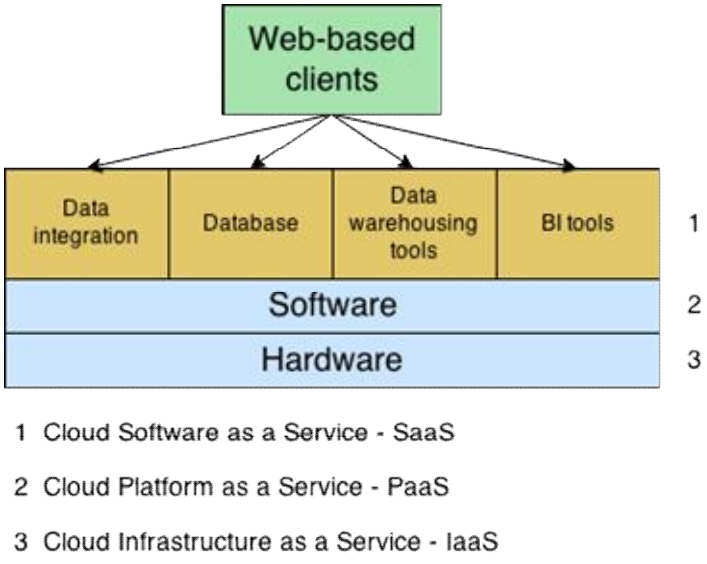


Fig. 6. Cloud BI or integrating Business Intelligence solutions into Cloud -> Basic Architecture [6]

Reyes takes the architecture of a BI in Cloud further than Shimaa Ouf and Mona Nasr did. Reyes considers that a BI solution is located in a SaaS model.

Cloud computing providers offer fully integrated BI solutions that are capable of fulfilling most of the companies' needs regarding BI software. By doing this, a company has the advantage of externalizing this type of service. In this case there is no need for software maintenance at all or any other backup concerns. Any software update will be handled by the Cloud provider.

There is another advantage in using web based BI software. Web development is gaining more and more control and traditional software programs are replaced by web based software. A huge advantage in web developing is the use of frameworks. This accelerates the software building process and eases the future developing of modules for the software created. Also, web based software allows users to access data from almost any device through a web browser or a specially designed application. Developing BI solutions with web technology can also reduce the costs of resulted software. In some cases there can be no costs at all.

4. Conclusions

Cloud computing represented, for a few years now, the leading technology in terms of scalability and flexibility. Using shared resources offers a great advantage for an expanding company. This is also reflected in investments. By not investing a lot into hardware architecture and the maintenance of it will allow organizations to expand much faster by investing into innovation, marketing, etc.

Integrating Business Intelligence software into a Cloud environment is necessary if the organization wishes to gain an advantage. This solution will provide a company the necessary tools to get in front of its competitors.

The ease of use and access that a Cloud BI offers will allow the employees to have mobility without harming the decision-making process.

Still, the primary concern of using a Business Intelligence solution in a cloud environment is related to privacy. This issue will persist until a viable solution will be found. This solution might be encryption software that can reside in Cloud.

References

[1] Cloud Computing: an overview - http://www.jatit.org/volumes/research-papers/Vol9No1/10Vol9No1.pdf

[2] Above the Clouds: A View of Cloud Computing, http://mars.ing.unimo.it/didattica/ingss/Lec_SaS/CloudView.pdf

[3] Review Study: Business Intelligence Concepts and Approaches, http://www.researchgate.net/profile/Vahid_Mirhosseini/publication/256667827_Review_Study_Business_Intelligence_Concepts_and_Approaches/links/00b7d5239641b7b653000000.pdf

[4] A model for Business Intelligence Systems' Development, http://revistaie.ase.ro/content/52/10%20-%20Bara,%20Botha.pdf

[5] The Cloud Computing: The Future of BI in the Cloud, http://www.ijcte.org/papers/404-G1093.pdf

[6] A Systems Thinking Approach to Business Intelligence Solutions Based on Cloud Computing, https://www.google.ro/url?sa=t&rct=j&q=&esrc=s&source=web&cd=10&cad=rja&uact=8&ved=0CFwQFjAJ&url=http%3A%2F%2Fdspace.mit.edu%2Fbitstream%2Fhandle%2F1721.1%2F59267%2F667715636.pdf&ei=VOUFVY3NFcLuaJzZgqgF&usg=AFQjCNGNbQ106ZK5v6337ClWEDOFUizPlQ&sig2=F2lvDePELe5vWNtTpfXrqQ

Alexandru Adrian TOLE (born 1986 in Romania) graduated from the Faculty of Domestic and International Commercial and Financial Banking Relations of the Romanian – American University in 2009. He also graduated the Scientific Master in Finance, Banking, Insurances. He works at the Ministry for Information Society. He is pursuing a Ph. D. in the area of Business Intelligence systems.

Master program Databases - Support for Business (BDSA)
Development competition BDSA - Oracle Academy 2015

General information and rules

The contest is open to all students from the master program Databases - Support for Business (BDSA).

Students will form teams of max. 3 students and they choose a team name and a mentor - a teacher from the master program BDSA. The team can involve first year students and / or second year.

Students communicate constantly with their mentor and provide them the application and documentation until the established dead-lines.

The application will be developed using Oracle technologies, MySQL and / or Java.

Each team made a project proposal that presents the team, objectives, technical solution development steps and impact.

The project proposal is presented in a Word document max. 10 pages, in English, containing the following information:

1. Project title
2. Team's description
3. Abstract (max 200 characters)
4. Keywords – max 5 words
5. Contents:
    a. Introduction – objectives and general description of the application
    b. Technical solution: the architecture and technologies
    c. Use Case diagrams and class diagrams in UML
    d. Roles and tasks for each member team
    e. Screenshots of the application
    f. Work plan, stages and methodologies
    g. Impact: social, economic and environment;
6. Conclusions

The project proposal is approved by the mentor that can brings additions / reviews.

Between 18 to 22 May, teams will deliver a 15-minute presentation of the project, including a demo of the application, according to the following structure:

1. Project title
2. Objectives and general description of the application
3. Technical solution: the architecture and technologies
4. Stages for the development
5. Roles and tasks for each member team
6. Demo of the applicaton
7. Conclusions and further development
8. Q&A

Important dates:

- 16 march - 11 april – Teams enrolment;
- 12 april - 9 may – Development;
- 9 – 18 may – Mentoring and reviewing;
- 18 – 22 may – Final presentation (1 day). Award Ceremony.