

**THE BUCHAREST ACADEMY OF
ECONOMIC STUDIES**

ISSUE

6

Database Systems Journal

ISSN: 2069 – 3230

Volume II (December 2011)



**Journal edited by Department
of Economic Informatics and
Cybernetics**

Database Systems Journal BOARD

Director

Prof. Ion LUNGU, PhD - Academy of Economic Studies, Bucharest, Romania

Editors-in-Chief

Prof. Adela Bara, PhD - Academy of Economic Studies, Bucharest, Romania

Prof. Marinela Mircea, PhD- Academy of Economic Studies, Bucharest, Romania

Secretaries

Assist. Iuliana Botha - Academy of Economic Studies, Bucharest, Romania

Assist. Anda Velicanu Academy of Economic Studies, Bucharest, Romania

Editorial Board

Prof Ioan Andone, A. I. Cuza University, Iasi, Romania

Prof Emil Burtescu, University of Pitesti, Pitesti, Romania

Joshua Cooper, PhD, Hildebrand Technology Ltd., UK

Prof Marian Dardala, Academy of Economic Studies, Bucharest, Romania

Prof. Dorel Dusmanescu, Petrol and Gas University, Ploiesti, Romania

Prof Marin Fotache, A. I. Cuza University Iasi, Romania

Dan Garlasu, PhD, Oracle Romania

Prof Marius Guran, Polytechnic University, Bucharest, Romania

Prof. Mihaela I. Muntean, West University, Timisoara, Romania

Prof. Stefan Nithchi, Babes-Bolyai University, Cluj-Napoca, Romania

Prof. Corina Paraschiv, University of Paris Descartes, Paris, France

Davian Popescu, PhD., Milan, Italy

Prof Gheorghe Sabau, Academy of Economic Studies, Bucharest, Romania

Prof Nazaraf Shah, Coventry University, Coventry, UK

Prof Ion Smeureanu, Academy of Economic Studies, Bucharest, Romania

Prof. Traian Surcel, Academy of Economic Studies, Bucharest, Romania

Prof Ilie Tamas, Academy of Economic Studies, Bucharest, Romania

Silviu Teodoru, PhD, Oracle Romania

Prof Dumitru Todoroi, Academy of Economic Studies, Chisinau, Republic of Moldova

Prof. Manole Velicanu, PhD - Academy of Economic Studies, Bucharest, Romania

Prof Robert Wrembel, University of Technology, Poznań, Poland

Contact

Calea Dorobanților, no. 15-17, room 2017, Bucharest, Romania

Web: <http://dbjournal.ro>

E-mail: editor@dbjournal.ro

Contents:

Oracle Exalytics: Engineered for Speed-of-Thought Analytics	3
Gabriela GLIGOR, Silviu TEODORU	
PL/SQL and Bind Variable: the two ways to increase the efficiency of Network Databases ...	9
Hitesh KUMAR SHARMA, Ranjit BISWAS, Aditya SHASTRI	
Problem Decomposition Method to Compute an Optimal Cover for a Set of Functional Dependencies	17
Vitalie COTELEA	
Business Intelligence using Software Agents	31
Ana-Ramona BOLOGA, Razvan BOLOGA	
Applications of Spatial Data Using Business Analytics Tools	43
Anca Ioana ANDREESCU, Anda VELICANU, Daniela MITROESCU	
Solutions for the Object-Relational Databases Design	51
Manole VELICANU, Iuliana BOTHA	

Oracle Exalytics: Engineered for Speed-of-Thought Analytics

Gabriela GLIGOR, Silviu TEODORU

Oracle Romania

Gabriela.gligor@oracle.com; silviu.teoforu@oracle.com

One of the biggest product announcements at 2011's Oracle OpenWorld user conference was Oracle Exalytics In-Memory Machine, the latest addition to the "Exa"-branded suite of Oracle-Sun engineered software-hardware systems.

Analytics is all about gaining insights from the data for better decision making. However, the vision of delivering fast, interactive, insightful analytics has remained elusive for most organizations. Most enterprise IT organizations continue to struggle to deliver actionable analytics due to time-sensitive, sprawling requirements and ever tightening budgets. The issue is further exasperated by the fact that most enterprise analytics solutions require dealing with a number of hardware, software, storage and networking vendors and precious resources are wasted integrating the hardware and software components to deliver a complete analytical solution.

Oracle Exalytics Business Intelligence Machine is the world's first engineered system specifically designed to deliver high performance analysis, modeling and planning. Built using industry-standard hardware, market-leading business intelligence software and in-memory database technology, Oracle Exalytics is an optimized system that delivers answers to all your business questions with unmatched speed, intelligence, simplicity and manageability.

Keywords: *engineered system, Business Intelligence, analytics, OLAP, architecture*

1 Introduction

The primary design principle of Oracle Exalytics is to enable fast and easy ad hoc analysis across large end-user communities using an in-memory processing engine. Speed of thought and instant response are the hallmarks of its functionality, making it highly applicable to a range of ad hoc, what-if analysis and forecasting and realtime planning applications.

The in-memory capabilities are key to enabling what Oracle calls a highly interactive and visual analytic experience for end users. Oracle claims that Exalytics introduces a new user interface designed to handle end-user queries "at the speed of thought." Peeling away the marketing speak, this means the product is designed to trawl through dense data sets (regardless of query, location, and device type and including desktops, laptops, tablets, and even smartphones) and provide results

almost instantaneously. Oracle claims that the algorithmic speed of the system means it can respond to queries as they are being typed – which is, in many ways, similar to how Google offers suggested searches based on a partially typed phrase [1].

What exactly is Oracle Exalytics? It is the latest addition to Oracle's "Exa"-branded family of pre-engineered software-hardware systems – in which the company integrates its software on its own hardware. Now that Oracle has its own line of hardware (through the acquisition of Sun), it appears to have fully embraced the "engineered systems" model as a way to sell both hardware and software while maintaining higher margins. Oracle Exalytics' immediate "engineered" siblings currently include the Exalogic Elastic Cloud middleware machine, the Exadata Database Machine, and the Oracle SPARC

SuperCluster T4-4 general-purpose servers [2].

2 Oracle Exalytics Architecture

Although Oracle Exalytics is a new product, the architecture is built on several existing Oracle products (Figure 1): parallelized versions of its TimesTen in-memory database and Oracle Essbase OLAP Server (a specialized in-memory version), together with an optimized version of the Oracle BI Foundation Suite (OBI EE 11g for standard

enterprise-grade BI query, analysis, reporting, dashboarding, and other visualizations). Most, if not all, of these software products have been modified to run in-parallel and in-memory data-processing architectures. All of these combine to deliver query optimization, complex multidimensional analysis and planning calculations, and enterprise-wide BI scale, respectively, through a revamped user interface that is designed for “speed-of-thought” analytics [1],[3].

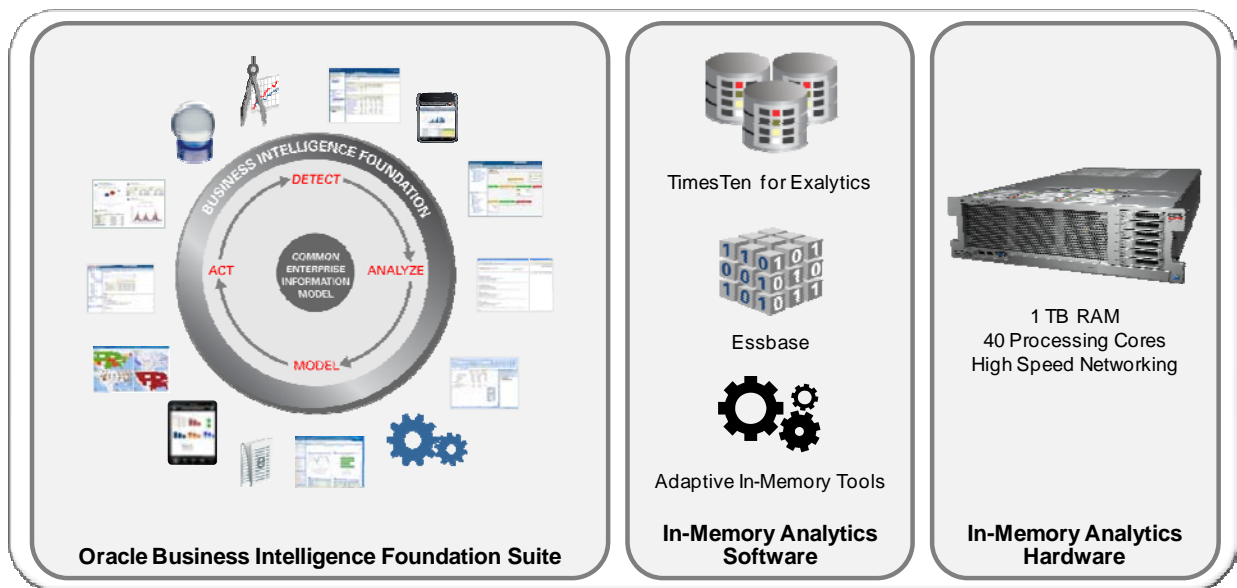


Figure 1. Oracle Exalytics Components

The Oracle Exalytics hardware is delivered in units of a single 3RU rack-mountable server that is optimally configured for in-memory analytics for business intelligence workloads. Multiple Oracle Exalytics machines can be clustered together to expand available memory capacity and to provide high availability. Oracle Exalytics includes powerful compute capacity, abundant memory, and fast networking options and is capable of direct attached storage options.

Oracle Exalytics is powered by four Intel Xeon® E7-4800 series processors and features high-speed interconnect technology between processors and I/O. Each processor supports 10 compute

cores providing a total of 40 cores for computation. The compute power is matched with 1TB of memory to provide sufficient capacity for in-memory analytics.

A high-performance business intelligence system requires fast connectivity to data warehouses, operational systems and other data sources. Besides, high-speed network connectivity is also required to create clusters which deliver high availability, load balancing and disaster recovery. Oracle Exalytics provides the following network interfaces to support the above requirements[1]:

- *InfiniBand*: Two quad-data rate (QDR) 40 GB/s InfiniBand ports

are available with each machine expressly for Oracle Exadata (only database machine that provides extreme performance for both data warehousing and online transaction processing (OLTP) applications) connectivity. When connected to Oracle Exadata, Oracle Exalytics becomes an integral part of the Oracle Exadata private InfiniBand network and has high-speed, low latency access to the database servers. When multiple Oracle Exalytics machines are clustered together, the InfiniBand fabric also serves as the high-speed cluster interconnect.

- *10 GB Ethernet*: Two 10 GB/s Ethernet ports are available for connecting to enterprise data sources and for client access.
- *1 GB Ethernet*: Four 1 GB/s Ethernet ports are available for client access.
- *Dedicated Integrated Lights Out Management (ILOM)*: Ethernet port for complete remote management.

All network interfaces support failover and can be used to setup a cluster without a single point of failure. Oracle Exalytics also includes redundant hot-swappable power supplies and fans.

Oracle Exalytics includes a high-performance direct attached storage system including a high-performance RAID HBA and 3.6TBs of raw disk capacity. Optionally, clusters of Oracle Exalytics machines can leverage network attached storage for storing shared metadata and configuration data.

3 Oracle Exalytics Software

Oracle Exalytics runs the Oracle Business Intelligence Foundation along with Oracle TimesTen In-Memory Database for Exalytics. Both BI Foundation and TimesTen In-Memory Database for Exalytics have been specifically enhanced to work together and have been optimized to

provide exclusive features on Oracle Exalytics hardware [1].

The *Oracle Business Intelligence Foundation* delivers the most complete, open, and integrated business intelligence platform on the market today. The Oracle BI Foundation provides comprehensive and complete capabilities for business intelligence, including enterprise reporting, dashboards, ad hoc analysis, multi-dimensional OLAP, scorecards, and predictive analytics on an integrated platform.

The Oracle BI Foundation includes the industry's best-in-class server technology for relational and multi-dimensional analysis and delivers rich end user experience that includes visualization, collaboration, alerts and notifications, search and mobile access.

Oracle TimesTen In-Memory Database (TimesTen) is a proven memory-optimized full-featured relational database with persistence. TimesTen stores all its data in memory optimized data structures and supports query algorithms specifically designed for in-memory processing. Using the familiar SQL programming interfaces, TimesTen provides real-time data management that delivers blazing-fast response times, and very high throughput for a variety of workloads.

Oracle TimesTen In-Memory Database for Exalytics, based on Oracle TimesTen In-Memory Database, has been specifically enhanced for analytical processing at in-memory speeds.

Oracle TimesTen In-Memory Database for Exalytics supports columnar compression that reduces the memory footprint for in-memory data. Compression ratios of 5X are practical and help expand in-memory capacity. Analytic algorithms are designed to operate directly on compressed data, thus further speeding up the in-memory analytics queries.

Oracle Essbase is the industry leading multi-dimensional OLAP Server for analytic applications. For Oracle Exalytics, Oracle Essbase has a number of optimizations for in-memory operation including

improvements to overall storage layer performance, enhancements to parallel operations, enhanced MDX syntax and a high performance MDX query engine. The Exalytics enhancements to Oracle Essbase provide up to 16X faster query execution as well as up to 6X reduction in write-back and calculation operations, including batch processes.

These enhancements are particularly important for advanced use cases such as planning and forecasting, providing faster cycle times and supporting more number of users than ever before.

Oracle Exalytics includes two in-memory analytics engines that provide the analytics capability - Oracle TimesTen *In-Memory* Database for Exalytics and Oracle Essbase with in-memory optimizations for Exalytics. These two data management engines are leveraged in the following four techniques to provide high performance in-memory analytics for a wide variety of business intelligence usage scenarios at workgroup, departmental and enterprise scale. These are:

- In-Memory Data Replication
- In-Memory Adaptive Data Mart
- In-Memory Intelligent Result Cache
- In-Memory Cubes

The Oracle Exalytics Business Intelligence Machine also supports *clustering* to provide scalability and high availability. It supports both active-active and active-passive configurations. A cluster configuration also can be configured to pool the available memory resources to accommodate larger data sets in-memory.

4. The Complementary Roles of Oracle Exalytics and Oracle Exadata

The Oracle Exadata Database Machine [4] is the only database machine that provides extreme performance for both data warehousing and online transaction processing (OLTP) applications, making it the ideal platform for consolidating onto grids or private clouds. It is a complete package of servers, storage, networking, and software that is massively scalable, secure, and redundant. With Oracle Exadata Database Machine, customers can reduce IT costs through consolidation, manage more data on multiple compression tiers, improve performance of all applications, and make better business decisions in real time.

Oracle Exalytics complements Oracle Exadata's (Figure 2) high performance query processing capabilities by delivering best in class user experience for analytical workloads including reporting, dashboards, ad-hoc and OLAP. Oracle Exalytics has been designed from the ground-up to be complementary to Oracle Exadata. Starting from the network interfaces, protocols to middleware to database interaction, Oracle Exalytics provides the best overall cost of ownership when connected to Oracle Exadata. Oracle Exalytics comes with pre-configured and pre-tested options to get the best performance, and the lowest Total Cost of Ownership (TCO) without extensive tuning with Oracle Exadata.

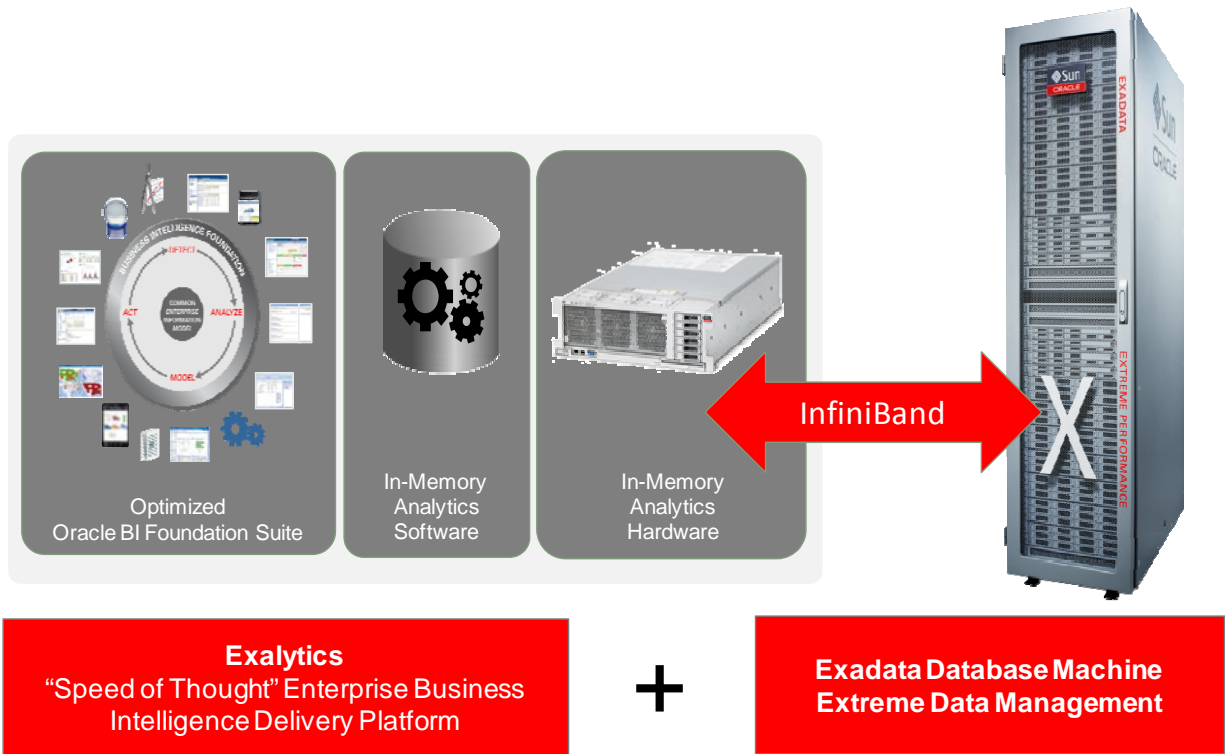


Figure 2. Complementary Roles of Oracle Exalytics and Oracle Exadata

The following figure illustrates key software supported in the Oracle Exadata Database Machine hosting a datawarehouse and the

Oracle Exalytics Business Intelligence Machine hosting the BI components.

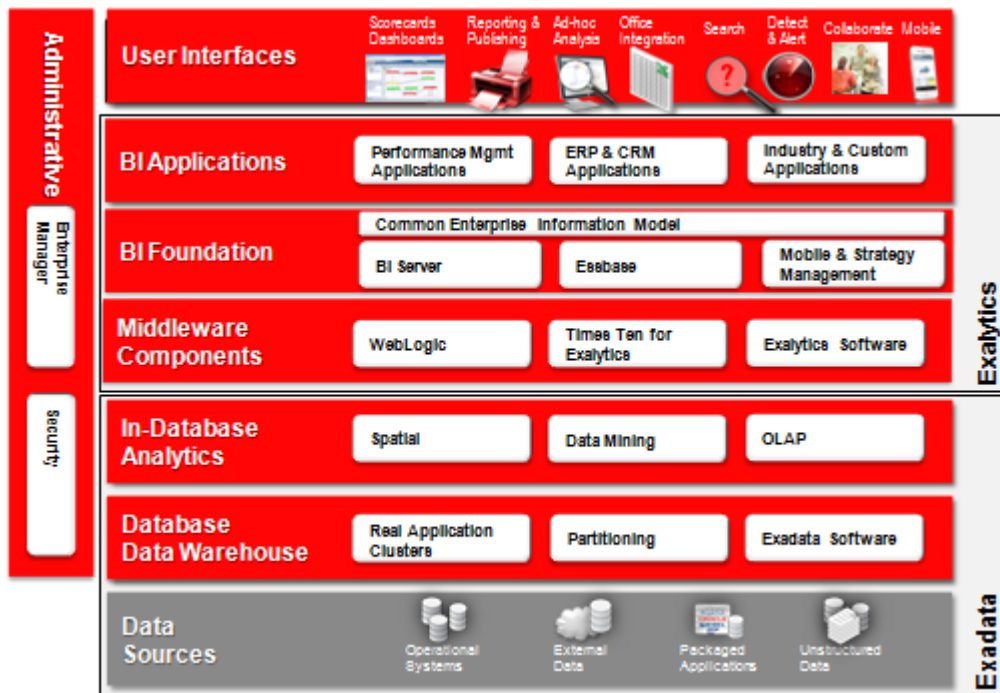


Figure 3. Key software supported in the Oracle Exadata and Exalytics

The Oracle Exalytics BI machine supports optimum SQL generation for Oracle

Exadata. For large analytics deployments where the data warehouse can't entirely fit

into Oracle Exalytics in-memory cache, Oracle Exalytics deployments can benefit by leveraging Oracle Exadata's massively parallel processing and extreme performance capabilities.

In addition, Oracle Exalytics can use Oracle Exadata as an extension to its in-memory cache/data mart. Such a configuration boosts the capacity of the in-memory cache/data mart and is especially suited for providing uniform responsiveness over large federated deployments.

5 Conclusions

Exalytics bundles in some mature and well-established technologies, and is the latest part of Oracle's strategy to parallelize its entire data processing stack. In fact, all of the new "Exa"-branded Oracle-engineered systems are parallelized configurations of servers, networks, and storage, while the software is a virtual machine, operating system, database, and middleware.

Data warehousing servers and BI servers have different roles and each provide value in a completing a business intelligence infrastructure. In the past, organizations have faced tradeoffs when trying to put together optimal configurations of the back-end and middle-tier components. Vendors often oversold the value of the individual components when they have offered only

one side of the equation. Now, the Oracle Exalytics Business Intelligence Machine and the Oracle Exadata Database Machine can be deployed together to provide the optimal end-to-end footprint to fulfill the needs of both IT and the LOBs, with support across the entire footprint coming from a single vendor.

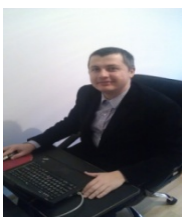
References

- [1] <http://www.oracle.com/us/solutions/ent-performance-bi/business-intelligence/exalytics-bi-machine/overview/exalytics-introduction-1372418.pdf>
- [2] <http://www.oracle.com/us/corporate/analystreports/infrastructure/ovum-oracle-exalytics-1377315.pdf>
- [3] <http://www.oracle.com/us/dm/h2fy11/nsl100013617-exalytics-514264.html>
- [4] "The Complementary Roles of Oracle Exalytics and Oracle Exadata", Oracle White Paper, October 2011, Oracle Corporation



Gabriela GLIGOR works as Business Intelligence Solution Sales Executive for Oracle Romania with more than 5 years of experience in Business Intelligence area, covering Sales, Project Management, Consulting and Training. She has in depth experience with business intelligence technologies and a comprehensive understanding of the financial services industry. My specialties include Business Intelligence and Data Warehousing, Database

Design (OLTP and DW), Oracle Business Intelligence Suite Enterprise Edition Plus, Oracle OLAP, Oracle Essbase, Oracle Real Time Decisions, SAS, Risk Management, Basel II, Scoring.



Silviu TEODORU works as a Technology Consultant in the Oracle Romania BI&DW team with more than 5 years experience in BI & DW area. He also acts as a solution architect for local engagements in the last three years.

He holds a PHD - "Informatics Solutions for Performance Development in Banking Domain", obtained at The Bucharest Academy of Economic Studies – Cybernetics, Statistics and Economic Informatics Faculty, Economic

Informatics Department.

PL/SQL and Bind Variable: the two ways to increase the efficiency of Network Databases

Hitesh KUMAR SHARMA

Assistant Professor, ITM University

Ranjit BISWAS

Associate Director Faculty, Manav Rachna International University Faridabad

Aditya SHASTRI

Vice-Chancellor, Banasthali University, Rajasthan-304022, India

hiteshsharma@itmindia.edu, adityashastri@yahoo.com, ranjitbiswas@yahoo.com

Modern data analysis applications are driven by the Network databases. They are pushing traditional database and data warehousing technologies beyond their limits due to their massively increasing data volumes and demands for low latency. There are three major challenges in working with network databases: interoperability due to heterogeneous data repositories, proactively due to autonomy of data sources and high efficiency to meet the application demand. This paper provides the two ways to meet the third challenge of network databases. This goal can be achieved by network database administrator with the usage of PL/SQL blocks and bind variable. The paper will explain the effect of PL/SQL block and bind variable on Network database efficiency to meet the modern data analysis application demand.

Key Words: Network Database, Web Application, Bind Variable, PL/SQL, Middleware.

1 Introduction

Most Web applications of any size involve the use of a database. They are pushing traditional database and data warehousing technologies beyond their limits. [1]. Typically, a web application allows the addition or creation of new records (for example, when a new user registers on the site), and the reading and searching of many records in a database. The most common bottleneck when developing a Web application is in the reading of a large number of records from a database, or executing a particularly complex SELECT statement against the database. Writing to or updating a database usually is performed on a small number of records at a time. This is often much less of an issue than cases that involve reading thousands of records at a time.

By eliminating unused fields from SELECT statements, we can reduce the complexity of the query and reduce the amount of data sent over the network (or at least between the database server and the

web script). The net affect of making such changes is a reduced database read time.

PL/SQL is the language of choice for data-centric application development in Network databases. In most programming languages, database work involves connecting to the server, mapping datatypes and manually preparing and processing result sets. PL/SQL is a procedural language that is so tightly integrated with the SQL language that most of these tasks are either eliminated completely or incredibly simple.

The two major costs decide the performance of a network database.

1.1. Cost 1: Round Trips

The first basic cost of retrieving data is the "round trip". Database programmers speak of a "round trip" as occurring whenever you send a request the server and retrieve some results. Each round trip to the server carries some overhead, as the server must do some basic work to allocate and release resources at the start and end of the request. This overhead is added to the base

cost the server must pay to actually go out to disk to find and retrieve your data.

If your application makes more round trips than are necessary, then the program will be slower than it could be.

1.2. Cost 2: Retrieval Size

An application runs code (such as a Java servlet) that makes queries to a backend database to customize the content, typically based on the user's request or a stored user profile. Overloading the work of the application server (e.g., executing the Java servlets) to proxy nodes is not difficult, but the central database server remains a performance bottle-neck [2].

Every byte that the application retrieves from the server carries a cost at several points. The server must go to disk and read it, the wire must carry the load from the db server to the web server, and the web server must hold the result in memory. If your web code regularly retrieves more information than it needs, then the program will be slower than it could be.

2. Related work

Companies across all industries are seeing very steep increases in the amount of data they must process. For example, one recent study [9] has estimated that the amount of data stored in data warehouses has been growing by an average of 173% per year across all industries. This rate of growth is substantially faster than the typical 12 to 18-month doubling of hardware capacity as dictated by Moore's law, Shuggart's law and others. As a result, for data analytics workloads hardware continues to become slower relative to the demands being placed on it.

The internet is a collection of millions upon millions of local, regional, national, and global networks. It commenced in 1969 with four supercomputers across the U.S. networked to the Pentagon.

Very few people used the Internet for the first 20 years. Explosive growth took place after 1992 when the Netscape Navigator web browser incorporated HTML

protocols to read HTML codes invented by particle physicists in Switzerland in 1990. This was the beginning of the "first generation" of network computing on what became known as the world wide web (WWW) or simply the "web." The first generation was mainly one way flows of information from web server computers to client user computers on the web. At this same time, the first generation of database interactive computing was confined to local and wide area networks (LANs and WANs). Although data files could be transmitted across the Internet using FTP and other protocols, databases could not interact on the WWW or the Internet as a whole. Most web applications are still in the first HTML generation.

The second generation of networking and databases followed quickly when web servers commenced to interact in a more formal way with remote clients on the Internet. With special types of **middleware** software, database servers could process data transmitted back from remote Internet client computers. For example, customer orders and market surveys could be processed and server-side databases could be updated without human intervention. Middleware CGI scripting and later ActiveX and Java software enabled web servers, database servers, and remote clients on the Internet to become more interactive. The second generation is relatively new and growing in popularity at this time.

The third generation is only just emerging and is hard to put into words. It is best described as distributed network computing. In the second generation, middleware links to "front ends" of database servers on the server side when clients transmit signals. In the third generation, databases can be distributed globally and can communicate with each other with "back-end" distributed network computing. There is virtually no difference between having all databases on one computer with one operator versus having

databases on 100 computers with 100 operators residing anywhere in the world. The third generation of network database is using three-tier architecture, Client/Server with Middleware layer. When business logic is processed in a client application, it is often necessary to pass a succession of statements between the application and the database server. Storing application code in the database takes this a step further as application logic is removed from the client layer and precompiled in the database, allowing modification without the need for a redeployment of client software. Each request and response involves network traffic, which can greatly affect overall performance. Running application logic on the database server can increase efficiency by reducing network traffic.

This paper explains the two ways to increase the efficiency of network databases by reducing network traffic.

3. The First way: Placing of PL/SQL block on Database server

PL/SQL is the language for data-centric application development [4] in Network databases. In most programming languages, database work involves connecting to the server, mapping datatypes and manually preparing and processing result sets. PL/SQL is a procedural language that is so tightly integrated with the SQL language that most of these tasks are either eliminated completely or incredibly simple.

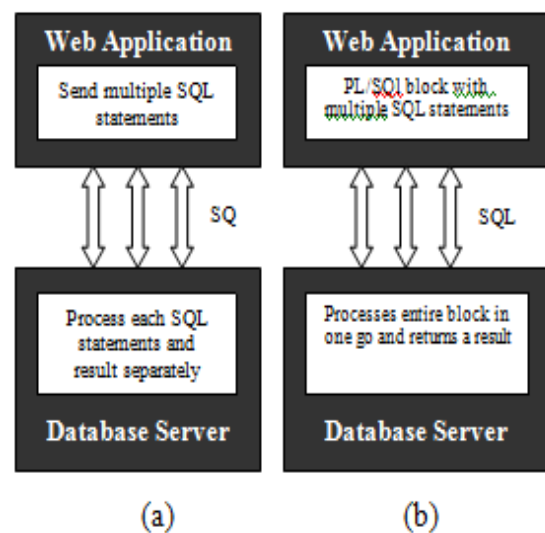
The datatypes available in PL/SQL are a superset of those available in SQL, so datatype conversions between SQL and PL/SQL are rarely needed. As a result PL/SQL allows interaction with both the data and metadata of database objects with greater ease and efficiency than is possible with most other languages. In addition PL/SQL supports dynamic SQL allowing statements to be created at runtime for greater flexibility.

Running application logic as PL/SQL on the database server can increase efficiency

by reducing network traffic. When business logic is processed in a client application, it is often necessary to pass a succession of statements between the application and the database server. Each request and response involves network traffic, which can greatly affect overall performance.

Passing a PL/SQL block containing multiple statements to the server can reduce network round trips, thereby improving performance. Storing application code in the database takes this a step further as application logic is removed from the client layer and precompiled in the database, allowing modification without the need for a redeployment of client software.

The PL/SQL language is available on all platforms, making it significantly more portable than many programming languages. When application logic is located within the database, changes in client programming models have a reduced impact, as only presentation of the data is controlled at that level. (See Fig 1.1)



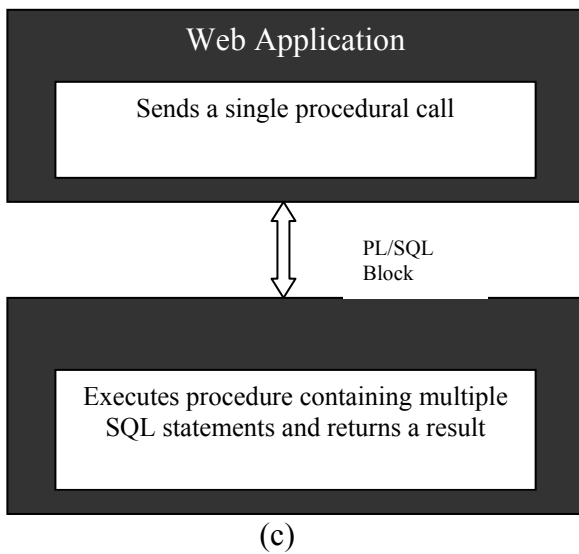


Figure 1: PL/SQL to improve performance , Fig 1(a) shows that heavy network usage to communicate with the server with separate SQL statements, Fig 1(b) shows that low network usage to communicate with the server using PL/SQL, Fig 1(c) shows that very low network usage to communicate with the server by placing PL/SQL processing logic on Server,

Centralizing application logic enables a higher degree of security and productivity. The use of Application Program Interfaces (APIs) can abstract complex data structures and security implementations from client application developers, leaving them free to do what they do best.

3.1 PL/SQL Architecture

The PL/SQL language is made up of both procedural code and SQL statements. When valid PL/SQL code is executed, the PL/SQL engine executes all procedural code and sends SQL statements to the SQL engine of the database server. Figure 1.2 represents this process in action for a PL/SQL block.

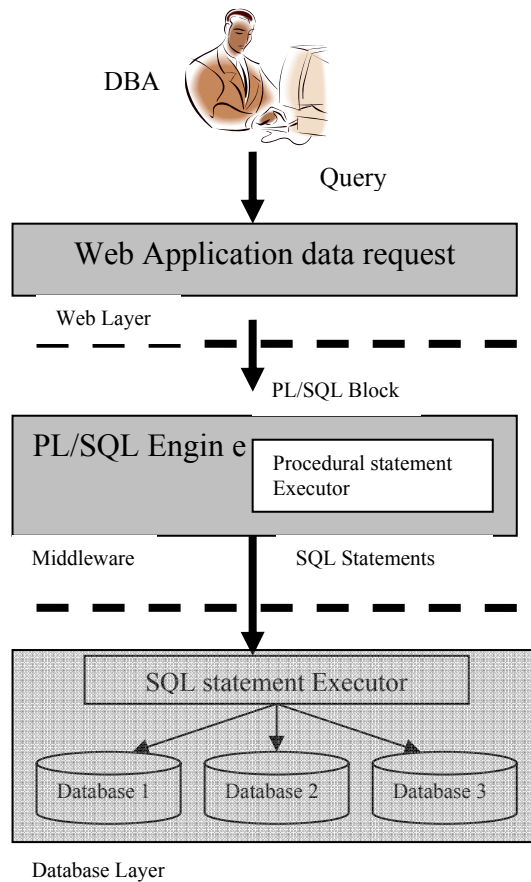


Figure 1.2 – PL/SQL Architecture.

The database contains a PL/SQL engine, which is used to execute all stored procedures, functions, packages, objects and triggers. This allows application logic to be processed entirely within the database layer.

Recently, a number of systems have been proposed with a similar architecture for scaling the delivery of database-backed dynamic content [5, 7, 8, 9]. In each of these systems users interact with proxy servers that mimic a traditional three-tiered architecture (containing a web server to handle user requests, an application server to generate dynamic content, and a database server as a backend data repository).

Some application development tools, such as Oracle Forms and Oracle Reports, have their own PL/SQL engine, allowing procedural logic to be processed with no reference to the database server.

3.2 Overview of PL/SQL Elements

Blocks in PL/SQL

Blocks are the organizational unit for all PL/SQL code, whether it is in the form of an anonymous block, procedure, function, trigger or type. A PL/SQL block is made up of three sections: declaration, executable and exception. Only the executable section is mandatory.

```
[DECLARE
  -- delarations]
BEGIN
  -- statements
[EXCEPTION
  -- handlers
END;
```

Based on this definition, the simplest valid block is shown below, but it does not do anything.

```
BEGIN
  NULL;
END;
```

The optional declaration section allows variables, types, procedures and functions do be defined for use within the block. The scope of these declarations is limited to the code within the block itself, or any nested blocks or procedure calls. The limited scope of variable declarations is shown by the following two examples. In the first, a variable is declared in the outer block and is referenced successfully in a nested block. In the second, a variable is declared in a nested block and referenced from the outer block, resulting in an error as the variable is out of scope.

```
DECLARE
  l_number NUMBER;
BEGIN
  l_number := 1;
  BEGIN
    l_number := 2;
  END;
END;
```

PL/SQL procedure successfully completed.

```
BEGIN
  DECLARE
    l_number NUMBER;
  BEGIN
```

```
    l_number := 1;
  END; l_number := 2;
END;
```

```
/
  l_number := 2;
  *
```

```
ERROR at line 8:
ORA-06550: line 8, column 3:
PLS-00201: identifier 'L_NUMBER' must be
declared
ORA-06550: line 8, column 3:
PL/SQL: Statement ignored
SQL>
```

The main work is done in the mandatory executable section of the block, while the optional exception section is where all error processing is placed. The following two examples demonstrate the usage of exception handlers for trapping error messages. In the first, there is no exception handler so a query returning no rows results in an error. In the second, the same error is trapped by the exception handler, allowing the code to complete successfully.

```
DECLARE
  l_date DATE;
BEGIN
  SELECT SYSDATE
  INTO l_date
  FROM dual
  WHERE 1=2; -- For zero rows
END;
```

```
/
DECLARE
  *
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 4
```

```
DECLARE
  l_date DATE;
BEGIN
  SELECT SYSDATE
  INTO l_date
  FROM dual
  WHERE 1=2; -- For zero rows
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    NULL;
END;
```

```
/
PL/SQL procedure successfully completed.
```

4. The Second way: Using Bind Variables

For every statement issued against the server, Oracle searches the shared pool to see if the statement has already been parsed. If an exact text match of the statement is already present in the shared pool a soft parse is performed as the execution plan for the statement has already been created and can be reused. If the statement is not found in the shared pool a hard parse must be performed to determine the optimal execution path.

The important thing to remember from the previous paragraph is the term “exact text match”, as different numbers of spaces, literal values and case will result in a failure to find a text match, such that the following statements are considered different.

```
SELECT 1 FROM dual WHERE dummy = 'X';
```

```
SELECT 1 FROM dual WHERE dummy = 'Y';
```

```
SELECT 1 FROM DUAL WHERE dummy = 'X';
```

```
SELECT 1 FROM dual WHERE dummy = 'X';
```

The first two statements only differ by the value of the search criteria, specified using a literal. In these situations exact text matches can be achieved by replacing the literal values with bind variables that have the correct values bound to them. Using the previous example the statement passed to the server might look like this.

```
SELECT 1 FROM dual WHERE dummy = :B1;
```

For every execution the bind variable may have a different value, but the text sent to the server is the same allowing for an exact text, which results in a soft parse.

There are two main problems associated with applications that do not use bind variables:

- Parsing SQL statements is a CPU intensive process, so reparsing similar statements constantly represents a waste of CPU cycles.
- Parsed statements are stored in the shared pool until they are aged out. By

not using bind variables the shared pool can rapidly become filled with similar statements, which waste memory and make the instance less efficient.

The Script_bind.sql script illustrates the problems associated with not using bind variables by using dynamic SQL to simulate an application sending insert statements to the server.

Script_bind.sql

```
CREATE TABLE bind_variables (
  code VARCHAR2(10)
);
BEGIN
  -- Perform insert without bind variables.
  FOR i IN 1 .. 10 LOOP
    BEGIN
      EXECUTE IMMEDIATE
        'INSERT INTO bind_variables (code)
VALUES ('' || i || '')';
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        NULL;
    END;
  END LOOP;
  -- Perform insert with bind variables.
  FOR i IN 1 .. 10 LOOP
    BEGIN
      EXECUTE IMMEDIATE
        'INSERT INTO bind_variables (code)
VALUES (:B1)' USING TO_CHAR(i);
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        NULL;
    END;
  END LOOP;
  COMMIT;
END;
```

```
/
-- Display the associated SQL text.
COLUMN sql_text FORMAT A60
COLUMN executions FORMAT 9999
SELECT sql_text,
       executions
FROM v$sql
WHERE INSTR(sql_text, 'INSERT INTO
bind_variables') > 0
AND INSTR(sql_text, 'EXECUTE') = 0
ORDER BY sql_text;
DROP TABLE bind_variables;
```

The script starts by creating a test table and executing a simple insert statement 10 times, where the insert statement concatenates a value into the string rather than using a bind variable. Next it repeats this process but this time uses a bind

variable rather than concatenating the value into the string. Finally it displays the SQL text parsed by the server and stored in the shared pool, which requires query access on the v\$sql view. The results from the script are displayed below

```
* SQL> @ Script_bind.sql
Table created.
PL/SQL procedure successfully completed.
SQL_TEXT
EXECUTIONS
-----
-----
insert into bind_variables (code) values
('1') 1
insert into bind_variables (code) values
('10') 1
insert into bind_variables (code) values
('2') 1
insert into bind_variables (code) values
('3') 1
insert into bind_variables (code) values
('4') 1
insert into bind_variables (code) values
('5') 1
insert into bind_variables (code) values
('6') 1
insert into bind_variables (code) values
('7') 1
insert into bind_variables (code) values
('8') 1
insert into bind_variables (code) values
('9') 1
insert into bind_variables (code) values
(:b1) 10
11 rows selected.
Table dropped.
```

From this we can see that when bind variables were not used the server parsed and executed each query as a unique statement, whereas the bind variable statement was parsed once and executed 10 times. This clearly demonstrates how applications that do not use bind variables can result in wasted memory in the shared pool, along with increased CPU usage.

5. Conclusion

It is not difficult to create database applications that perform well. The basic rules of thumb are to make a minimum number of round trips to the server and to retrieve precisely the values that you need and no more. These ideas work well because they minimize your most expensive operation, which is disk access. the procedures/functions are stored in the database and are, therefore, executed on the database server which is likely to be more powerful than the clients which in turn means that stored procedures should run faster; the code is stored in a pre-compiled form which means that it is syntactically valid and does not need to be compiled at run-time, thereby saving resources; each user of the stored procedure/function will use exactly the same form of queries which means the queries are reused thereby reducing the parsing overhead and improving the scalability of applications; as the procedures/functions are stored in the database there is no need to transfer the code from the clients to the database server or to transfer intermediate results from the server to the clients. This results in much less network traffic and again improves scalability when using PL/SQL packages, as soon as one object in the package is accessed, the whole package is loaded into memory which makes subsequent access to objects in the package much faster stored procedures/functions can be compiled into “native” machine code making them even faster. there is an overhead involved in switching from SQL to PL/SQL, this may be significant in terms of performance but usually this overhead is outweighed by performance advantages of using PL/SQL more memory may be required when using packages as the whole package is loaded into memory as soon as any object in the package is accessed native compilation can take twice as long as normal compilation

References

- [1] Agrawal, R., et al. "The Claremont Report on Database Research", <http://db.cs.berkeley.edu/claremont/>, May 2008.
- [2] C. Olston, A. Manjhi, C. Garrod, A. Ailamaki, B. Maggs, and T. Mowry. A scalability service for dynamic web applications. In Proc. Conference on Innovative Data Systems Research (CIDR), 2005.
- [3] K. Burleson, Donald "Creating a self-tuning Oracle database"
- [4] John Garmany "Easy Oracle PL/SQL Programming:" Get Started Fast with Working PL/SQL Code .
- [5] K. Amiri, S. Park, R. Tewari, and S. Padmanabhan. DBProxy: A dynamic data cache for Web applications. In Proc. International Conference on Data Engineering, 2003.
- [6] M. Altinel, C. Bornhovd, S.Krishnamurthy, C. Mohan, H. Pirahesh, and B. Reinwald. Cache tables: Paving the way for an adaptive database cache. In Proc. International Conference on Very Large Data Bases, 2003.
- [7] Q. Luo, S. Krishnamurthy, C. Mohan, H. Pirahesh, H. Woo, B. G. Lindsay, and J. F. Naughton. Middle-tier database caching for e-business. In Proc. ACM SIGMOD International Conference on Management of Data, 2002.
- [8] Winter, R, "Why Are Data Warehouses Growing So Fast?",B-eye Network, <http://www.b-eye-network.com/view/7188>,April 2008

Hitesh KUMAR SHARMA, The author is An Assistant Professor in ITM University. He has published 8 research papers in National Journals and 1 research paper in International Journal. Currently He is pursuing his Ph.D. in the area of database tuning.

Ranjit BISWAS, Associate Director, MIRU, Faridabad, Published about 100 research papers in International journals/bulletins of USA/Europe, out of which more than 40 papers are independently published papers and the rest are published jointly with other authors (with Ph.D. scholars). Haryana, INDIA.

Aditya SHASTRI, Ph.D. MIT, Published about 200 research papers in international journals on Graph Theory with applications in Communication, Computer Graphics and Parallel Processing ,Vice Chancellor, Director, Banasthali University, Banasthali, INDIA

Problem Decomposition Method to Compute an Optimal Cover for a Set of Functional Dependencies

Vitalie COTELEA

Academy of Economic Studies of Moldova
61 Banulescu-Bodoni Street, Chisinau, Republic of Moldova
Email: vitalie.cotelea@gmail.com

The paper proposes a problem decomposition method for building optimal cover for a set of functional dependencies to decrease the solving time. At the beginning, the paper includes an overview of the covers of functional dependencies. There are considered definitions and properties of non redundant covers for sets of functional dependencies, reduced and canonical covers as well as equivalence classes of functional dependencies, minimum and optimal covers. Then, a theoretical tool for inference of functional dependencies is proposed, which possesses the uniqueness property. And finally, the set of attributes of the relational schema is divided into equivalence classes of attributes that will serve as the basis for building optimal cover for a set of functional dependencies.

Keywords: Logical Database Design, Functional Dependencies, Optimal Cover, Problem Decomposition

1 Introduction

The A set of functional dependencies may have different structures, and several properties of the database are dictated by structure and quality of the functional dependencies set. In addition, a set of functional dependencies can be modified, simplified, while retaining its qualitative aspect of the deduction system.

The notion of equivalent sets of functional dependencies is a central one in database design. This is because the design process starts from a given schema, modifying it to obtain a schema with desirable qualities, but equivalent with former in terms of integrity constraints both structural and behavioral.

These sets of functional dependencies usually are called covers. If the structure of dependencies set is simple, it is more efficient the checking of the database consistency and the possibility of application of integrity constraints is facilitated.

Further, the computing of optimal cover for a set of functional dependencies is dealt with.

2 Covers for functional dependencies

Definition 1. ([1], p.71) Two sets of functional dependencies F and G over scheme R are equivalent, written $F \equiv G$, if $F^+ = G^+$.

It is said that in case if $F \mid -G$, then F covers G . If sets F and G are equivalent, they are cover one for another.

If $F \equiv G$, that is if $F^+ = G^+$, then any dependence $X \rightarrow Y$ that is implied by F is implied by G . So to check if F and G are equivalent, take any dependence $X \rightarrow Y$ in F and check if $G \mid -X \rightarrow Y$. If some dependence $X \rightarrow Y$ does not belong to G^+ , then $F^+ \neq G^+$. Then, analog, check if any dependence $V \rightarrow W$ in G is derived from F . If all dependencies are derived from these appropriate sets, the sets F and G are equivalent.

Consider by $|F|$ and $\|F\|$ the cardinality of F and number of attributes involved by F (including repeated), respectively. As the complexity of the algorithm to deduct a dependence from a given set of functional dependencies, that is a inference of dependence $X \rightarrow Y$ from the set F , is

$O(\|F\|)$, it is not hard to see that the algorithm for determining whether two sets of functional dependencies F are G are equivalent, will consume a polynomial time relative to the size of the input data, $O(\|G\| \cdot \|F\| + \|F\| \cdot \|G\|)$.

Example 1. The sets $F = \{AB \rightarrow C, AC \rightarrow D, AD \rightarrow B, C \rightarrow B\}$ and $G = \{AD \rightarrow C, AB \rightarrow D, C \rightarrow B\}$ are equivalent, but F is not equivalent to the set $G' = \{AB \rightarrow C, AC \rightarrow D, AD \rightarrow B, AC \rightarrow B\}$.

Definition 2. ([2], p.295) A set F of functional dependencies is non redundant, if $\nexists G$, so that $G \subset F$ (that is, if there is no proper subset G of F) with $G \equiv F$. If such subset exists, then F is redundant. The set F is a non redundant cover for G , if F is cover for G and F is non redundant.

Example 2. Let $G = \{A \rightarrow BC, B \rightarrow C\}$. The set $F = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$ is a cover for the set G , but it is a redundant cover, since $F^1 = \{A \rightarrow B, B \rightarrow C\}$ is a cover for G and $F^1 \subset F$.

There is an alternative definition (procedural) for notion of non redundant cover.

Definition 3. [3] The set F of functional dependencies is non redundant, if there is no functional dependency $X \rightarrow Y$ in F , such that $(F - \{X \rightarrow Y\}) \models X \rightarrow Y$. Otherwise, F is redundant.

This definition forms the basis for the algorithm which computes a non redundant cover. It is worth to mention that the result from the application of the algorithm depends on the order of examining the functional dependencies.

It is not hard to see that a set of functional dependencies may have more than one non redundant cover. The result depends on the order in which functional dependencies are examined for removal. In [4], it is

considered the representation of all non redundant covers of a set of functional dependencies, and in [5] an efficient algorithm for computing a non redundant cover is presented.

If F is a non redundant set of functional dependencies, then it can not be removed any functional dependency from F , without affecting the equivalence of the obtained set with the previous one.

In contrast, functional dependencies in F can be reduced in size by removing some attributes from them.

Definition 4. ([1], p.74) Let F be a set of functional dependencies over scheme R and let $X \rightarrow Y \in F$. Attribute A in R is extraneous in dependency $X \rightarrow Y$ with respect to F , if

$$1. A \in X, F - \{X \rightarrow Y\} \cup \{(X - \{A\}) \rightarrow Y\} \equiv F$$

or

$$2. A \in Y, F - \{X \rightarrow Y\} \cup \{X \rightarrow (Y - \{A\})\} \equiv F.$$

In other words, the attribute A is extraneous in dependency $X \rightarrow Y$, if it can be removed from the left or right side of dependency, without changing the closure of F . The elimination process of extraneous attributes is called, respectively, left reduction or right reduction of dependencies.

Definition 5. ([1], p.74) A set F of functional dependencies is left-reduced (right-reduced), if every functional dependency in F contains no extraneous attributes in the left (right) side. If a set of functional dependencies is left-reduced and right-reduced then it is reduced.

Theorem 1. If a set of functional dependencies is reduced, then it is non redundant.

Proof. The statement is true, because if it is assumed that the set of functional dependencies is not non redundant, then from the set may be removed at least one functional dependency and therefore all the attributes of this dependence are extraneous, which contradicts the claim that the set is reduced.

It's obvious that if a dependency is

redundant, then all its attributes are extraneous. To avoid dependencies of the form $X \rightarrow \emptyset$, it is assumed that the set which must be reduced is non redundant.

It seems that reduced cover can be calculated by finding and removing at random extraneous attributes. But, considering left and right sides of the dependencies in a different order, it can get different results. When the right sides are examined first, after considering the left, it may appear redundant attributes in right sides. So, if the set of dependencies is not non redundant at algorithms entry, the eliminating of extraneous attributes must begin with the left sides.

Definition 6. ([1], p.77) A set of functional dependencies F is canonical, if F is non redundant, left-reduced and every functional dependency in F is of the form $X \rightarrow A$.

Example 3. The set $F = \{A \rightarrow B, A \rightarrow C, B \rightarrow D\}$ is a canonical cover for $G = \{A \rightarrow BC, B \rightarrow D\}$.

Theorem 2. A canonical set of functional dependencies is a reduced set of functional dependencies.

Proof. The validity of this statement follows directly from Definition 6. Since a canonical set of functional dependencies is non redundant and every functional dependency has a single attribute on the right side, it is right-reduced. Since it is also left reduced, it is reduced.

The reverse statement is not true. This is shown in Example 3, where the set G is reduced, but it is not canonical. However, the relationship between reduced and canonical covers can be characterized by the following theorem.

Theorem 3. ([1], p.77) Let F be a reduced cover. If the set G of functional dependencies is formed by splitting each dependency $X \rightarrow A_1 \dots A_n$ in F into $X \rightarrow A_1, \dots, X \rightarrow A_n$, then G is a canonical cover of F . Conversely, if G is a canonical cover, it is reduced. If the set F

is formed by aggregating all dependencies in G with equal left sides into a single functional dependency, then F is also a reduced cover.

Definition 7. ([1], p.78) Let F be a set of functional dependencies over scheme R and let $X, Y \subseteq R$. Two sets of attributes X and Y are equivalent, written $X \leftrightarrow Y$, under the set F , if $F \models X \rightarrow Y$ and $F \models Y \rightarrow X$.

Definition 7 suggests that the set F can be partitioned into equivalence classes. That is, on F can be defined an equivalence relation: dependencies $X \rightarrow Y$ and $V \rightarrow W$ in F belong to a class of equivalence, if and only if $X \leftrightarrow V$ under the set F .

Definition 8. Let F be a set of functional dependencies over scheme R and let $X, Y \subseteq R$. It is defined as the set of equivalence classes of functional dependencies for the set X of attributes with respect to F , denoted $E_F(X)$, the set $E_F(X) = \{V \rightarrow W \mid V \rightarrow W \in F \ \& \ X \leftrightarrow V\}$.

So $E_F(X)$ is the set of functional dependencies in F with left sides equivalent to X with respect to F .

Let \overline{E}_F be the set $\overline{E}_F = \{E_F(X) \mid X \subseteq R \ \& \ E_F(X) \neq \emptyset\}$. In

other words, \overline{E}_F is the set of all nonempty equivalence classes, in which the set F of functional dependencies is partitioned.

The next lemma shows the correlation between structures of two equivalent and non redundant sets of functional dependencies.

Lemma 1. ([1], p.78) Let F and G be equivalent, non redundant sets of functional dependencies over scheme R . Let $X \rightarrow Y$ be a functional dependency in F . There is a functional dependency $V \rightarrow W$ in G with $X \leftrightarrow V$ under F (hence under G).

Lemma above can be paraphrased as follows. In two non redundant covers F and G , for each dependency in F there is a dependency in G with equivalent left sides. Therefore, the equivalent non redundant sets of functional dependencies have the same number of equivalence classes.

Definition 9. ([1], p.79) A set F of functional dependencies is minimum if F has as few functional dependencies as any equivalent set G of functional dependencies, that is

$$\forall G \quad F \equiv G \Rightarrow |F| \leq |G|.$$

Theorem 4. Let F be a minimum set of functional dependencies. Then F is a non redundant set of functional dependencies.

Proof. The statement is true, because if it is assumed that the set F is not non redundant, then it can be removed from at least one functional dependency and therefore there will be a cover with fewer functional dependencies, which contradicts the assumption that it is minimum.

It is obvious that the reverse statement is not correct.

Consider by $PS_F(X)$ the set of left sides of the dependencies forming equivalence class $E_F(X)$, that is:

$$PS_F(X) = \{V \mid V \rightarrow W \in E_F(X)\}.$$

Then there is:

Lemma 2. ([1], p.81) Let F be a non redundant set of functional dependencies. Pick X , a left side of some functional dependency in F and any Y equivalent to X (that is $X \leftrightarrow Y$ under F). There exists a set Z in $PS_F(X)$ such that $(F - E_F(X)) \models Y \rightarrow Z$.

Lemma 3. [6] Let F and G be equivalent, non redundant sets of functional dependencies over scheme R . Let X be a left side of some functional dependency in F and any Y such that $X \leftrightarrow Y$ under F . If

$$Y \rightarrow Z \in (F - E_F(X))^+ \quad , \quad \text{then}$$

$$Y \rightarrow Z \in (G - E_G(X))^+ .$$

Theorem 5. [6] A non redundant set F of functional dependencies is a minimum set, if and only if there are no distinct functional dependencies $X \rightarrow Y$ and $V \rightarrow W$ in any equivalence class $E_F(X)$ such that $X \rightarrow V \in (F - E_F(X))^+$.

Corollary 1. If F and G are equivalent, minimum sets of functional dependencies, then the corresponding equivalence classes contain the same number of functional dependencies.

Corollary 2. If F and G are equivalent and minimum sets of functional dependencies, then for each left side $X_j \in PS_F(X)$ there is a single left side V_k in $PS_G(X)$ such that $X_j \rightarrow V_k \in (F - E_F(X))^+$ and $V_k \rightarrow X_j \in (F - E_F(X))^+$.

Proposition 1. The existence of the bijection indicated in Corollary 2, allows substitution of some left sides of a minimum set of functional dependencies by the corresponding left sides of another minimum cover, which does not affect the equivalence of minimal sets. In addition, the new set of functional dependencies will continue to be minimal.

The above theorem states that if a non redundant set G has two dependencies $X \rightarrow Y$ and $V \rightarrow W$, such that $X \leftrightarrow V$

and $(G - E_G(X)) \models X \rightarrow V$, then G is not minimum set of functional dependencies. These two dependencies can be substituted with other functional dependency $V \rightarrow YW$. Consequently, it is obtained an equivalent set of functional dependencies with one dependency less.

The algorithm to minimize a set of functional dependencies is based on this process.

A set F of functional dependencies can be evaluated taking into account the number

of attribute symbols (including repeated) involved by the functional dependencies in F . For example, the set $F = \{AB \rightarrow C, C \rightarrow B\}$ consists of five attribute symbols, that is $\|F\| = 5$.

Definition 10. ([1], p.86) A set F of functional dependencies is optimal if there is no equivalent set G of functional dependencies with fewer attribute symbols than F , that is

$$\forall G \ F \equiv G \Rightarrow \|F\| \leq \|G\|$$

Theorem 6. ([1], p.86) An optimal set of functional dependencies is reduced and minimum.

Example 4. The set $F = \{ABC \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$ is not an optimal set of functional dependencies, because the set $G = \{AD \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$ consists of fewer symbols than F and $F \equiv G$. It should be noted that the set G is optimal.

Unfortunately, there is not known any algorithm of polynomial complexity that would build an optimal cover for a given set of functional dependencies. This problem belongs to the class of NP-complete problems.

A size reduction technique for solving this problem is proposed below.

Definition 11. Let $X \rightarrow Y \in F$ be a functional dependency. The set X of attributes is a determinant for the set Y of attributes, if no proper subset X' of set X exists such that $X' \rightarrow Y \in F^+$.

3 An inference model of functional dependencies

To prove several assertions about functional dependency inference, a model called Maximal derivation is proposed. This model deducts in a linear mode the set of attributes functionally dependent (under a set of functional dependencies) for a given set of attributes.

It has the uniqueness property and it is very easy to use in demonstrating claims

about functional dependencies structures. In general, this model is not something else than a sequence of sets of attributes, which are built iteratively, involving for their construction groups of functional dependencies with left sides included in the previous set.

Since the maximal derivation is a sequence of sets of attributes, there can be built its reduced version, called simply – derivation, which effectively applies in the inference of functional dependencies. Some properties of the proposed model and its inference ability have been proven. It is equivalent to applying the inference model of dependencies with Armstrong axioms.

In [7] is presented a model called *maximal derivation* (the name is taken from [8]). The construction concept is based on the algorithm which computes the closure of the set of attributes under the set of dependencies, as described in [5].

Definition 12. Let F be a set of functional dependencies over set R of attributes and let $X \subseteq R$. *Maximal derivation* of the set of attributes X under the set F of dependencies is a sequence of sets of attributes $\langle X_0, X_1, \dots, X_n \rangle$, so that:

- (1). $X_0 = X$;
- (2). $X_i = X_{i-1} \cup Z$, $i = \overline{1, n}$, where $Z = \bigcup_j W_j$ for all dependencies $V_j \rightarrow W_j \in F$ which satisfies $V_j \subseteq X_{i-1}$ and $W_j \not\subseteq X_{i-1}$;
- (3). Nothing else from R is a member of X_i .

Before we show that maximal derivation is a powerful derivation tool for functional dependencies, two of its properties are considered.

Lemma 4. [7] If $X \subseteq Y$ and sequences $\langle X_0, X_1, \dots, X_n \rangle$, $\langle Y_0, Y_1, \dots, Y_m \rangle$ are maximal derivations of the sets X and Y , respectively, under F , then for any X_i

exists a set Y_j such that $X_i \subseteq Y_j$ and $j \leq i$.

This property tells us that if the set of attributes is larger, then the terms of maximal derivation converge faster and they are closer to the beginning of the maximal derivation.

Lemma 5. [7] If $\langle X_0, X_1, \dots, X_n \rangle$ is the maximal derivation of the set X under the set F of functional dependencies, then $X \rightarrow X_i \in F^+$, $i = \overline{0n}$.

The property represented by this lemma states that any term of maximal derivation is functionally determined by the set of attributes on which this derivation is built. Based on these two properties the next theorem will be proven:

Theorem 7. [7] Let $\langle X_0, X_1, \dots, X_n \rangle$ be the maximal derivation of the set X under the set F of functional dependencies. Then $X \rightarrow Y \in F^+$ if and only if $Y \subseteq X_n$.

This theorem actually proves that applying the maximal derivation for the deduction of functional dependencies from a given set of dependencies is equivalent to applying Armstrong's axioms for the dependencies deduction process, because this theorem's proof is based only on the inference of these rules. But unlike other derivation instruments, the deduction using maximal derivation is unique, i.e. there are no two different maximal derivations for the deduction of a functional dependency from a given set of dependencies.

Due to the fact that Armstrong rules are sound and complete, the maximal derivation has the same properties. In addition, the derivation is a deterministic process and not a nondeterministic one as is the case of deduction using rules of inference.

Definition 13. [7] Let $X \rightarrow Y \in F^+$ and $\langle X_0, X_1, \dots, X_n \rangle$ be the maximal derivation of the set X under F . Let X_i be the first element which contains the set Y . Then the subsequence

$\langle X_0, X_1, \dots, X_i \rangle$ is considered to be the *derivation* (not necessarily the maximal one) of the functional dependency $X \rightarrow Y$ under F .

From Theorem 7 and Definition 13 follows

Corollary 3. [7] $X \rightarrow Y \in F^+$ then and only then when the derivation of $X \rightarrow Y$ under F exists.

Corollary 4. [7] If $X \rightarrow Y \in F^+$ and the dependency $V \rightarrow W \in F$ is used for computing the derivation of the $X \rightarrow Y$ under F , then $X \rightarrow V \in F^+$.

The correctness of this statement logically follows from the Lemma 5 and the reflexivity and transitivity rules.

It is obvious that the last element, X_n , in maximal derivation is nothing else but X^+ . And Theorem 7 says that $X \rightarrow Y$ follows logically from F , if $Y \subseteq X^+$. So, the maximal derivation serves as a theoretical model for algorithm to building the closure of a set X of attributes under a set F of functional dependencies.

The uniqueness of derivation is explained by the fact that every step of the algorithm to create the next term of maximal derivation involves all dependencies that satisfy condition (2), respectively.

4 Redundant and non redundant equivalence classes of attributes

In this section, we introduce the notion of contribution graph for a set of functional dependencies and condensed graph of the contribution graph. Also, it is presented that strongly connected components of a contribution graph divide the set of attributes of relational schema into equivalence classes of attributes and a strict partial order can be defined over the nodes of condensed graph.

Mapping of functional dependencies inference in contribution graph is examined and there are introduced concepts of redundant equivalence class and non redundant equivalence class of

attributes.

Given a set F of functional dependencies on the set R of attributes, that are part of the relation scheme $Sch(R, F)$, a contribution graph is drawn, in order to represent F .

Definition 14. Contribution graph $G = (S, E)$ of set F is a graph that:

- for $\forall A \in R$ there exists in S a vertex labeled with attribute A ;
- for $\forall X \rightarrow Y \in F$ and for $\forall A \in X$ and $\forall B \in Y$ there exists in E an edge $a = (A, B)$, that is directed from vertex A to vertex B .

Example 5. If $F = \{ABC \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$ and $R = \{A, B, C, D, E\}$, then the contribution graph of set F of dependencies is presented in Figure 1.

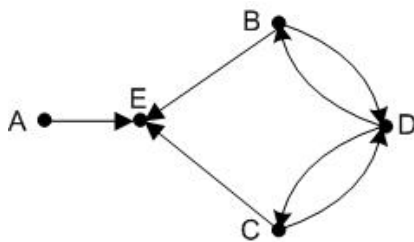


Fig. 1. A contribution graph for set F

Two vertices $A, B \in S$ are strongly connected, if and only if there exists in graph G a path from A to B and backwards, from B to A . It is obvious that the relation of strong connectivity is an equivalence relation. So, there is a partition of set of vertices S into pairwise disjoint subsets. That is, $S = \bigcup_{i=1}^n S_i$ and all vertices in $S_i, i = \overline{1, n}$, are strongly connected, and every two vertices from different subsets are not strongly connected.

In accordance with this partition, sub-graphs $G_i = (S_i, E_i), i = \overline{1, n}$ are called strongly connected components [9] of the graph G , where E_i represents the set of edges that connect pairs of vertices in S_i .

Example 6. The set of vertices of the graph represented in Figure 1 are split into three equivalence classes $S_1 = \{A\}, S_2 = \{B, C, D\}$ and $S_3 = \{E\}$.

The concept of the condensed graph of a contribution graph is introduced:

Definition 15. Let G^* be the condensed graph of the graph G . Set of vertices of graph G^* represents set $\{G_1, \dots, G_n\}$ of all strongly connected components of graph G and there is an edge from vertex G_i to vertex G_j of graph G^* , if there exists in G at least one edge that connects one vertex from component G_i to one vertex from component G_j .

Obviously the graph G^* is an acyclic one.

Example 7. The condensed graph of graph from Figure 1 has three vertices and two edges, as shown in Figure 2.



Fig. 2. Condensed graph of the graph from Figure 1

Over the set of vertices of graph G^* a strict partial order is defined. Vertex G_i precedes vertex G_j , if G_j is accessible from G_i . Now, the equivalence classes S_1, \dots, S_n will be sorted based on the corresponding order graph's G^* vertices.

Lemma 6. If $X \rightarrow Y \in F^+$ and X is a determinant of set Y under F , then for every attribute $A \in (X - Y)$ there is an attribute $B \in Y$ so that in the contribution graph G there exists a path from vertex A to vertex B and for every attribute $B \in (Y - X)$ there exists in X an attribute A , from which the vertex B can be reached.

Proof. Let attribute $B \in (Y - X)$ and let the subset X' of set X be determinant for

B under F . Because $X' \rightarrow B \in F^+$, according to Definition 13, there is a derivation $H = \langle X'_0, X'_1, \dots, X'_m \rangle$ for dependency $X' \rightarrow B$ under F . Then, based on Corollary 4, there exists a sequence of dependencies $V_1 \rightarrow W_1, \dots, V_q \rightarrow W_q$ in F , where $A \in V_1$, $B \in W_q$ and $W_i \cap V_{i-1} \neq \emptyset$, for $i = \overline{1, q-1}$. Contribution graph has a structure, such that for every dependency $V_j \rightarrow W_j$ in F , from each vertex labeled with an attribute in V_j an edge leaves to every vertex labeled with an attribute in W_j . So, there exists a path from every vertex $A \in X'$ to vertex B .

It must be mentioned that, if \overline{X} is considered the union of all determinant of attributes in $Y - X$, then $\overline{X} \cup X \cap Y = X$. Indeed, if we suppose that the set $\overline{X} \cup X \cap Y$ is a proper subset of set X , this will contradict the supposition that X is a determinant for Y under F .

Corollary 5. If reduced dependency $V \rightarrow W$ is used non redundantly in building the derivation H for dependency $X \rightarrow Y$ under F , then in contribution graph G there exists a path from every vertex labeled with an attribute in V to every vertex labeled with an attribute in Y . The following theorem shows a correlation between non redundant equivalence classes of attributes and the left and right sides of a left-reduced dependency.

Theorem 8. Let $X \rightarrow Y \in F^+$, where X is a determinant for Y under F and $X, Y \subseteq T_1 \cup \dots \cup T_m$. For a T_j , where $j = \overline{1, m}$, the following takes place: if $Y \cap T_j \neq \emptyset$, then $X \cap T_j \neq \emptyset$.

Proof. The soundness of this statement is proven by contradiction: let $Y \cap T_j \neq \emptyset$,

but $X \cap T_j = \emptyset$. Evidently that $X \subseteq T_1 \cup \dots \cup T_{j-1} \cup T_{j+1} \cup \dots \cup T_m$ and $X \rightarrow (Y \cap T_j) \in F^+$. Let X' , where $X' \subseteq X$, is determinant for $Y \cap T_j$ under F . According to Lemma 6, on the contribution graph of the set F of dependencies, from every vertex labeled with an attribute in X' there exists a path to a vertex labeled with an attribute in $Y \cap T_j$. Thereby, $X' \subseteq T_1 \cup \dots \cup T_{j-1}$. But, in this case, T_j is redundant. Therefore, $X \cap T_j \neq \emptyset$. Then, more so $X \cap T_j \neq \emptyset$ takes place. A contradiction has been reached.

Below, there are shown a series of features related to the determinants and the sets of redundant and non redundant equivalence classes of attributes.

Theorem 9. If X is a determinant under F of set $S_1 \cup \dots \cup S_j$, where $j = \overline{1, n}$, then $X \subseteq S_1 \cup \dots \cup S_j$.

Proof. Let $X \not\subseteq S_1 \cup \dots \cup S_j$. Then there exists an equivalence class S_t , where $t = \overline{j, n}$, such that $X \cap S_t \neq \emptyset$. By Lemma 6, in the contribution graph G , from every attribute $A \in X \cap S_t$ there is a path towards B , where $B \in S_1 \cup \dots \cup S_j$. But this fact contradicts the supposition that sets S_1, \dots, S_j precede the set S_t .

Corollary 6. If X is a determinant of set $S_1 \cup \dots \cup S_n$ under F , then $X \cap S_1 \neq \emptyset$.

Proof. Indeed, for every attribute B in S_1 or $B \in X$, or, according to Lemma 6, there is in X an attribute A from which vertex B is accessible in contribution graph G . But then A is also a member of equivalence class S_1 .

Definition 16. Equivalence class S_j is called non redundant, if and only if for

every attribute A in S_j , the expression $(\bigcup_{i=1}^n S_i - S_j) \rightarrow A \notin F^+$ holds.

Considering Lemma 6, it can be concluded that set S_j is non redundant, if and only if for every attribute A in S_j , the expression $(\bigcup_{i=1}^{j-1} S_i) \rightarrow A \notin F^+$ holds.

From the ordered sequence of sets S_1, \dots, S_n a sequence of ordered non redundant sets can be built T_1, \dots, T_n , where $T_1 = S_1$ and $T_j = S_j - (\bigcup_{i=1}^{j-1} T_i)_F^+$ for $j = \overline{2, n}$. As a result of this process, some sets T_j can become empty. These empty sets can be excluded from the sequence and a sequence of nonempty sets T_1, \dots, T_m will be obtained, keeping the precedence of prior sets.

Proposition 2. $T_1 = S_1$.

Proposition 3.

$$(T_1 \cup \dots \cup T_m) \rightarrow (S_1 \cup \dots \cup S_n) \in F^+$$

Example 8. Sequence of equivalence classes of attributes $S_1 = \{A\}$, $S_2 = \{B, C, D\}$ and $S_3 = \{E\}$ turns into the following sequence of non redundant equivalence classes of attributes: $T_1 = \{A\}$, $T_2 = \{B, C, D\}$.

Lemma 7. If X is a determinant under F of set $S_1 \cup \dots \cup S_n$, then Z , where $Z = X \cap (S_1 \cup \dots \cup S_j)$ and $j = \overline{1, n}$, is a determinant for $S_1 \cup \dots \cup S_j$ under F .

Proof. According to Theorem 9, the expression $X \subseteq S_1 \cup \dots \cup S_n$ takes place. First it will be shown that $Z \rightarrow (S_1 \cup \dots \cup S_j) \in F^+$. Lets suppose the contrary: $Z \rightarrow (S_1 \cup \dots \cup S_j) \notin F^+$. Then there exists a set Z' , where $Z' \subseteq X$, which is a determinant of set $S_1 \cup \dots \cup S_j$ and

$Z' \cap (\bigcup_{i=j+1}^n S_i) \neq \emptyset$. Considering Lemma 6, there is a path from every vertex labeled with A in $Z' \cap (\bigcup_{i=j+1}^n S_i)$ that leads to a vertex B in $\bigcup_{i=1}^j S_i$. A contradiction has been encountered. Therefore, $Z \rightarrow (S_1 \cup \dots \cup S_j) \in F^+$.

To complete the proof of this lemma, it will be shown that Z is a determinant under F of set $S_1 \cup \dots \cup S_j$. Indeed, if it is considered that Z is not a determinant of F under F , then there must exist in Z an attribute A , such that $(Z - \{A\}) \rightarrow (S_1 \cup \dots \cup S_j) \in F^+$. But then $(Z - \{A\}) \rightarrow Z \in F^+$ takes place, fact that implies $(X - \{A\}) \rightarrow X \in F^+$. So, a contradiction has been encountered, that X is a determinant of set $S_1 \cup \dots \cup S_n$ under X .

Theorem 10. If set X of attributes is a determinant of set $S_1 \cup \dots \cup S_n$, then $X \subseteq T_1 \cup \dots \cup T_m$.

Proof. Let S_j be the first set of attributes that doesn't coincide with T_j and assume that there is an attribute A in X , such that $A \in S_j$ and $A \notin T_j$. Lemma 7 implies that $(X \cap (S_1 \cup \dots \cup S_j)) \rightarrow (S_1 \cup \dots \cup S_j) \in F^+$.

Since $A \notin T_j$, then $(X \cap (S_1 \cup \dots \cup S_j)) \rightarrow A \in F^+$.

So $(X - \{A\}) \rightarrow X \in F^+$, thus X is not a determinant of set $S_1 \cup \dots \cup S_n$ under F . Appealing to Theorem 10, Lemma 7 can be paraphrased for non redundant equivalence classes of attributes.

Lemma 8. If X is a determinant under F of set $T_1 \cup \dots \cup T_m$, then Z , where $Z = X \cap (T_1 \cup \dots \cup T_j)$ and $j = \overline{1, m}$, is a determinant for $T_1 \cup \dots \cup T_j$ under F .

5 Calculation of determinants using a scheme decomposition method

In this section, there are proposed theoretical tools that can be the basis of relational schemes decomposition algorithm for computing determinants related to the scheme, rather all independent components from which all determinants of scheme can be built. The scheme is partitioned in subschema to solve the determinants searching problem for each subschema separately. Then, for each subschema will be found determinants with the fewest attributes (including repeated). The groups of attributes from the set of functional dependencies that are determinants in some subschema are substituted with the shortest determinants. This happens only for equivalence classes of functional dependencies containing the determinants in the left or right sides as subsets.

Below there are considered the functional dependencies and non redundant equivalence classes of attributes. It is examined how the determinants of a non redundant equivalence class of attributes in relation to a set of functional dependencies in the projection of this set of dependencies on the attributes of non redundant equivalence class of attributes are reflected.

The next theorem presents a property of non redundant equivalence classes of attributes if the set of functional dependencies, on which these classes are built, is reduced.

Theorem 11. If the dependency $V \rightarrow W \in F$ is reduced and $W \cap T_j \neq \emptyset$, then $V \cap T_j \neq \emptyset$.

Proof. Appealing to the definition of contribution graph, from each vertex labeled with an attribute in V an edge leaves to every vertex labeled with an attribute in W . Then $V \cap T_i = \emptyset$ for $i = \overline{j+1, m}$. Assuming that $V \cap T_j = \emptyset$,

then there is $(T_1 \cup \dots \cup T_{j-1}) \rightarrow V \in F^+$. From this it follows that $(T_1 \cup \dots \cup T_{j-1}) \rightarrow W \in F^+$. But in this case, T_j is redundant, fact that contradicts the nature of this set of attributes.

Proposition 4. Let $X \rightarrow Y \in F^+$ be a functional dependency. If $Y \subseteq T_1 \cup \dots \cup T_j$ and dependency $V \rightarrow W \in F$ is non redundantly used in derivation of dependency $X \rightarrow Y$ under F , then $V \cap T_i = \emptyset$ for $i = \overline{j+1, m}$.

Veracity of this statement is based directly on the Corollary 5.

Definition 17. Let F be a set of functional dependencies over the set R of attributes. Projection of the set F of dependencies, labeled $\pi_Z(F)$, on a set Z of attributes, where $Z \subseteq R$, is the set of functional dependencies defined by the expression $\pi_Z(F) = \{(X \cap Z) \rightarrow (Y \cap Z) \mid X \rightarrow Y \in F \ \& \ (X \cap Z) \neq \emptyset \ \& \ (Y \cap Z) \neq \emptyset\}$.

Then the following statement is true:

Lemma 9. If X , where $X \subseteq T_1 \cup \dots \cup T_j$, is a determinant of the set T_j under F , then $(X \cap T_j) \rightarrow T_j \in \pi_{T_j}^+(F)$.

Proof. According to Theorem 8, $X \cap T_j \neq \emptyset$. Since $X \rightarrow T_j \in F^+$, following Corollary 3, there is a derivation $H = \langle X_0, X_1, \dots, X_n \rangle$ for dependency $X \rightarrow T_j$ under F .

Let $H' = \langle Z_0, Z_1, \dots, Z_m \rangle$ be the maximal derivation for $X \cap T_j$ under $\pi_{T_j}(F)$. Given the Lemma 7, to prove the lemma, it suffices to show that $T_j \subseteq Z_m$.

Indeed, given that $T_j \subseteq X_n$, then either $X = T_j$, or T_j is formed in H from dependencies which contain attributes of T_j in their right side. In the first case,

$X \cap T_j = T_j$ and the dependency $(X \cap T_j) \rightarrow T_j$ is deduced from any set of dependencies. In the second case, taking account of Theorem 11, the dependencies used in H , that have some attributes of T_j in their right sides, have also attributes of T_j in the left sides. Thus, if H has used the dependency $V \rightarrow W$ in F and $W \cap T_j \neq \emptyset$, then H' has used the dependency $(V \cap T_j) \rightarrow (W \cap T_j)$ in $\pi_{T_j}(F)$. Therefore, $T_j \subseteq Z_m$.

Further, it is established the relationship between determinants of scheme and its subschema determinants obtained via projections.

Theorem 12. Let $Sch(\cup_{i=1}^n S_i, F)$ be a database schema. The set X , where $X \subseteq T_1 \cup \dots \cup T_m$, is a determinant for $T_1 \cup \dots \cup T_m$ under F , if and only if $X \cap T_1, \dots, X \cap T_m$ are determinants for T_1, \dots, T_m under $\pi_{T_1}(F), \dots, \pi_{T_m}(F)$, respectively.

Proof. Necessity. Let the set X be a determinant for $T_1 \cup \dots \cup T_m$ under F . It will be shown that $X \cap T_1, \dots, X \cap T_m$ are determinants for T_1, \dots, T_m under $\pi_{T_1}(F), \dots, \pi_{T_m}(F)$, respectively.

Will be proved this by applying mathematical induction on the number of non redundant equivalence classes, i , where $i = \overline{1, m}$. Let $i = 1$. Taking into account Lemma 8, $X \cap T_1$ is a determinant for T_1 under F . According to Lemma 9, $X \cap T_1$ is a determinant for T_1 in relation to $\pi_{T_1}(F)$.

It is assumed now that the assertion is fair for $T_1 \cup \dots \cup T_{k-1}$, namely, for a number of classes less than k and will demonstrate

that the affirmation is also true for a number of classes equal to k .

Since $X \cap (T_1 \cup \dots \cup T_k) \rightarrow (T_1 \cup \dots \cup T_k) \in F^+$, where, according to Lemma 8, $X \cap (T_1 \cup \dots \cup T_k)$ is a determinant for $(T_1 \cup \dots \cup T_k)$ under F , then $X \cap (T_1 \cup \dots \cup T_k) \rightarrow T_k \in F^+$. If it is assumed that for an attribute A , where $A \in X \cap T_k$, the expression $(X \cap (T_1 \cup \dots \cup T_k) \setminus \{A\}) \rightarrow T_k \in F^+$ holds, then, on the basis that $X \cap (T_1 \cup \dots \cup T_i)$ is determinant for $(T_1 \cup \dots \cup T_i)$, where $i = \overline{1, k-1}$, it follows that $(X \cap (T_1 \cup \dots \cup T_k) \setminus \{A\}) \rightarrow (X \cap (T_1 \cup \dots \cup T_k)) \in F^+$ takes place. But in this case, $X \cap (T_1 \cup \dots \cup T_k)$ will not be determinant for $(T_1 \cup \dots \cup T_k)$. Thus, there is $X' \subseteq X$, that $X \cap T_k \subseteq X'$ and X' is determinant for T_k under F . Whence, $X \cap T_k = X' \cap T_k$, is determinant for T_k under F and, according to Lemma 9, $X \cap T_k$ is determinant for T_k under $\pi_{T_k}(F)$.

Sufficiency. Now it will be proven that if the $X \cap T_1, \dots, X \cap T_m$ are determinants for T_1, \dots, T_m under $\pi_{T_1}(F), \dots, \pi_{T_m}(F)$, respectively, then the set Z , where $Z = X \cap T_1 \cup \dots \cup X \cap T_m$, is determinant for $T_1 \cup \dots \cup T_m$ under F .

It is obvious that $Z \rightarrow (T_1 \cup \dots \cup T_m) \in F^+$, where $Z = X \cap T_1 \cup \dots \cup X \cap T_m$. Assuming that for at least one attribute A in $X \cap T_i$ the expression $(Z - \{A\}) \rightarrow (T_1 \cup \dots \cup T_m) \in F^+$ holds, then, by virtue of Lemma 9, $X \cap T_i$ will not be

determinant for T_i under $\pi_{T_i}(F)$.
Consequently, $Z = X \cap T_1 \cup \dots \cup X \cap T_m$,
is determinant for $T_1 \cup \dots \cup T_m$ under F .
The theorem is proved.

Let $Sch(\bigcup_{i=1}^n S_i, F)$ be a relational schema,
and let $T_1 \cup \dots \cup T_m$ be the set of non
redundant equivalence classes of attributes
built on the set F of dependencies. Thus,
each determinant X of the set $T_1 \cup \dots \cup T_m$
under F consists of the union of
determinants for the set T_i (one from each
non redundant equivalence class of
attributes) under $\pi_{T_i}(F)$, where $i = \overline{1, m}$.
The problem of calculating the sets of
attributes that can be substituted with other
equivalent sets of attributes of smallest
cardinality for each equivalence class in
which the set F of dependencies of the
scheme $Sch(\bigcup_{i=1}^n S_i, F)$ is partitioned,
consists in finding all the determinants for
each T_i .

It should be noted that the set of
dependencies $\pi_{T_i}(F)$ may not be
minimum, even if F is minimum.
Moreover, it may be neither non
redundant.

Example 9. If $F = \{C \rightarrow B, B \rightarrow C, AB \rightarrow D, AD \rightarrow B\}$, then there are two non redundant classes of attributes $T_1 = \{A\}$ and $T_2 = \{B, C, D\}$. Projection on class T_2 of the set F will be $\pi_{T_2}(F) = \{C \rightarrow B, B \rightarrow C, B \rightarrow D, D \rightarrow B\}$. Although the set F of functional dependencies is minimum, the set $\pi_{T_2}(F)$ is not minimum, because there is an equivalent set of dependencies $F' = \{C \rightarrow B, B \rightarrow CD, D \rightarrow B\}$ with fewer dependencies.

Example 10. If $F = \{C \rightarrow B, B \rightarrow C, AB \rightarrow D, AD \rightarrow C\}$, also there are two

non redundant classes $T_1 = \{A\}$ and $T_2 = \{B, C, D\}$. In this case the projection $\pi_{T_2}(F) \equiv \{C \rightarrow B, B \rightarrow C, B \rightarrow D, D \rightarrow C\}$ is not non redundant, because the dependency $B \rightarrow C$ is redundant in $\pi_{T_2}(F)$. In other words, $\pi_{T_2}(F) \equiv \{C \rightarrow B, B \rightarrow D, D \rightarrow C\}$. Therefore, before computing the determinants of each set T_i it is useful to minimize the set $\pi_{T_i}(F)$ of functional dependencies.

It is to mention that partitioning the set of attributes in classes of equivalence, essentially reduces dimensions of problem of computing the optimal cover. For this, it suffices to consider the set T_i , for example, which may include the determinant with minimum cardinality. The set T_i contains all attributes involved in the determinants for T_i under $\pi_{T_i}(F)$. These will form all the determinants for $T_1 \cup \dots \cup T_m$ under F . In addition, left side of every dependency in $\pi_{T_i}(F)$, in essence, represents a determinant for T_i . In case $\pi_{T_i}(F) = \emptyset$, there is only one determinant, T_i itself. The following is an integrator example.

Example 11. Let $F = \{ABC \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$ be a minimum and reduced set of functional dependencies. Any set of functional dependencies can be minimized and reduced in polynomial time [1]. It is necessary to build an optimal cover of this set of dependencies.

The set F of functional dependencies is divided into equivalence classes of dependencies. Obviously, that on the equivalence classes of functional dependencies can be defined a strict partial order. Let $Attr(F_i)$ denote the set of attributes involved by dependencies of equivalence class F_i . The equivalence class

F_i precedes the equivalence class F_j if $Attr(F_j)^+ \subset Attr(F_i)^+$. In the considered example, the set of functional dependencies is divided into two equivalence classes $F = F_1 \cup F_2$, where $F_1 = \{ABC \rightarrow E\}$, and $F_2 = \{BC \rightarrow D, D \rightarrow BC\}$.

The contribution graph for the set of dependencies F has the form represented in Figure 1. As noted already above, the set of vertices of the graph in Figure 1 is divided into three equivalence classes of attributes $S_1 = \{A\}$, $S_2 = \{B, C, D\}$ and $S_3 = \{E\}$, and are reduced to the following sequence of non redundant equivalent classes of attributes $T_1 = \{A\}$, $T_2 = \{B, C, D\}$.

The set F of functional dependencies, below, is projected on the sets of attributes T_1 and T_2 , resulting in the following sets of functional

dependencies $\pi_{T_1}(F) = \emptyset$, $\pi_{T_2}(F) = \{BC \rightarrow D, D \rightarrow BC\}$. Thus, for the non redundant classes of attributes there were obtained the following sets of determinants $\{A\}$, $\{D, BC\}$, respectively.

Now the groups of attributes that are determinants and part of dependencies in F are substituted by those with the smallest length. Substitutions occur in the equivalence classes of dependencies which precede corresponding class that has generated the determinant. Therefore, the set of attributes BC of dependencies that are part of the equivalence class F_1 (there is only one dependency) is substituted by determinant D . Thus optimal cover is obtained as $F = \{AD \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$.

6 Conclusions

It is known that various types of covers

provide specific properties to database scheme. Referring to the problem of building optimal covers, it was found that it is the strictest structure of functional dependencies regarding the constituent elements.

Because the task of obtaining the optimal cover is classified as NP-complete problem, a way of achieving a solution, in acceptable time, is to apply a decomposition method to divide the original in smaller problems that could be solved, and then to combine particular solutions in order to construct the initial problem solution.

It should be mentioned that the proposed method does not change the complexity of the problem. Its nature continues to be NP-complete. However, it can be solved in such a way that it will reduce the time needed to impose constraints on database content and to reduce the time required to execute the algorithm for computing the closures.

References

- [1] D. Maier, "The theory of relational database", Computer Science Press, 1983, 637 p.
- [2] S. Sumathi, S. Esakkirajan, "Fundamentals of Relational Database Management Systems", 2007, Springer-Verlag Berlin and Heidelberg GmbH & Co. K., 776 p.
- [3] J. Paredaens, "About Functional Dependencies in a Database Structure and their Coverings", Phillips MBLE Lab. Report 342, 1977.
- [4] E. A. Lewis, L.C. Sekino, P. D. Ting, "A Canonical Representation for the Relational Schema and Logical Data Independence", *IEEE Computer Software and Applications Conf. (COMPSAC '77)*, 1977, pp.276-280.
- [5] C. Beeri, Bernstein, A. Philip, "Computational problems related to the design of normal form relational schemas", *ACM Trans. Database Syst.*, 1979, V.4, N 1, pp.30-59.

- [6] В. Котеля, Караниколов Аурел, “Минимальное покрытие в схеме базы данных”, *Мат.исслед.*, вып 87, Штиинца, Кишинев, 1986, с.49-61.
- [7] V. Cotelea, “An inference model for functional dependencies in database schemas”, *Meridian Ingineresc*, N.3, 2009, pp.89-92.
- [8] J. D. Ullman, “On Kent’s “Consequences of assuming a universal relation””, *ACM Trans. Database Syst.*, 1983, V.8, N.4, p.637-643.
- [9] S. Even, “Graph Algorithms”, Computer Science press, 1979, 250 p.



Vitalie COTELEA is Associate Professor at Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies of Moldova. He is the author and co-author over 90 scientific works, including one monograph and more than 10 books. His work focuses on Databases and Information Systems Design and Declarative Programming. He has graduated the Faculty of Mathematics and Cybernetics in 1974 of State University of Moldova, Chisinau. He holds a PhD diploma in Computer Science from 1988 of Kiev State University, Ukraine.

Business Intelligence using Software Agents

Ana-Ramona BOLOGA

Academy of Economic Studies, Bucharest

Razvan BOLOGA

Academy of Economic Studies, Bucharest

ramona.bologa@ie.ase.ro, razvanbologa@ase.ro

This paper presents some ideas about business intelligence today and the importance of developing real time business solutions. The authors make an exploration of links between business intelligence and artificial intelligence and focuses specifically on the implementation of software agents-based systems in business intelligence. There are briefly presented some of the few solutions proposed so far that use software agents properties for the benefit of business intelligence. The authors then propose some basic ideas for developing real-time agent-based software system for business intelligence in supply chain management, using Case Base Reasoning Agents.

Keywords: Business intelligence, Agent-Based Systems, Supply Chain Management, Case-Base Reasoning

1 Business intelligent nowadays. Real-time business intelligence

Business intelligence applications are not a new trend any more, but they have become a must during the last decade as a basic tool used by the modern management. Business intelligence is the result of the natural evolution in time of decision support systems and expert systems, systems that aimed at replacing humans in the decision making process or, at least, at offering solutions to the issues they are concerned of. Gartner's definition [Gartner, 1989] introduces business intelligence as an umbrella term that includes humans, processes and applications/tools used or information management, access and analysis in order to improve decisions and increase business performances.

In fact, business intelligence software solutions are applications that help organizations to know themselves and better understand themselves. The main objectives of a business intelligence system are:

- Accessing and integrating data from various data sources and

storing it, usually into a data warehouse,

- Analyzing data in order to transform it into information, and then into knowledge;
- Presenting information using an easy to understand and use graphic interface.

Our day's business intelligence applications focus on the way they can ensure the useful, correct and in-time information, usually taken from disparate data sources of the organization. The main concern is that of crossing the « information gap » between huge amounts of data available to the decision factor and the useful information, presented in a suggestive manner, that should support the decision making process. So, they insist very much on data collection stage and on presenting information in a friendly way, in form of reports and charts in order to give an image as clear as possible on the current organization state and they are less a real decision tool.

On the other side, most of BI platforms/applications need a specialist to run statistical reports or data mining process and to make the setup for the reports that are

accessed by various business users. Business users usually do not have real time access to data, as they work on a historical data warehouse, so they have no control and cannot choose between the various available data sources having different quality that are integrated and loaded into data warehouse through the ETL process (Extract, Transform and Load).

As business environment is continuously changing, the use of historical data stored in data warehouses can lead to results that are not any more applicable to the new situation, and the delay between the moment of the analysis and the moment of moving decision into action can decisively affect the opportunity and the effectiveness of the resulted actions. This is the reason why companies speak more and more about real time business intelligence or even real time enterprise.

Real time business intelligence makes a comparison between present business events and historical patterns in order to automatically detect problems and opportunities. The result should be the initiation of corrective actions or the adjusting of business rules for process optimization. When talking about business intelligence there are three types of latency: data latency (for data collecting and storage), analysis latency (for analyzing data and obtaining actionable information) and action latency (for reacting to changes and get action to change the business processes). The latest two are not considered by traditional business intelligence as they are dominated by manual processes. [15] Business intelligence in real time is event-driven and uses event stream processing techniques in order to allow event analysis without being transformed and stored in the database first. These in-memory techniques present the advantage of quick monitoring many events and data latency is reduced to milliseconds.

Here are some examples of domains that really need real time business

intelligence, as have various latency levels and data persistence requirements:

- supply chain optimization;
- call center management;
- quality management in manufacturing;
- global shipment and delivery monitoring;
- fraud detection in financial companies;
- real time marketing etc.

It is supposed that modern companies have achieved the transition to a performing management style, to an entrepreneurial culture that incorporates business intelligence elements. The next step is the usage of business intelligence system that can offer solutions to problems and can make decisions starting from the existing information. An organization can include intelligent behavior in its base functions by using business intelligence. And the current method to add intelligent behavior is to include artificial intelligence methods and techniques.

So, we can conclude that most of today business intelligence systems are passive systems. The main problems can be structured on three levels [1]:

a. Analytics level – most business intelligence solutions need an expert or an analyst to run or to setup them, and that fragments the information flow and leads to important delays.

b. Data integration level – data integration has critical importance. The stage of data preparation has a particular importance, even if the result presentation stage has bigger impact and impresses the beneficiary more. If some real time analysis is needed, then there are two possibilities: or the data warehouse is continuously fed from various real/time data sources, or the business intelligence system has direct access to operational data sources through a data integration layer. The data integration layer is a unified data layer that offers a common metadata structure and unifies data access by creating a unique virtual view.

c. Operational level – the information

flow is interrupted by annual interventions. This level has two main functions: business activity monitoring (by using dashboards and charts) and real-time process tuning and change

(implying that Bi tools should be automatically linked to business processes and they adjust and lead the process parameters).

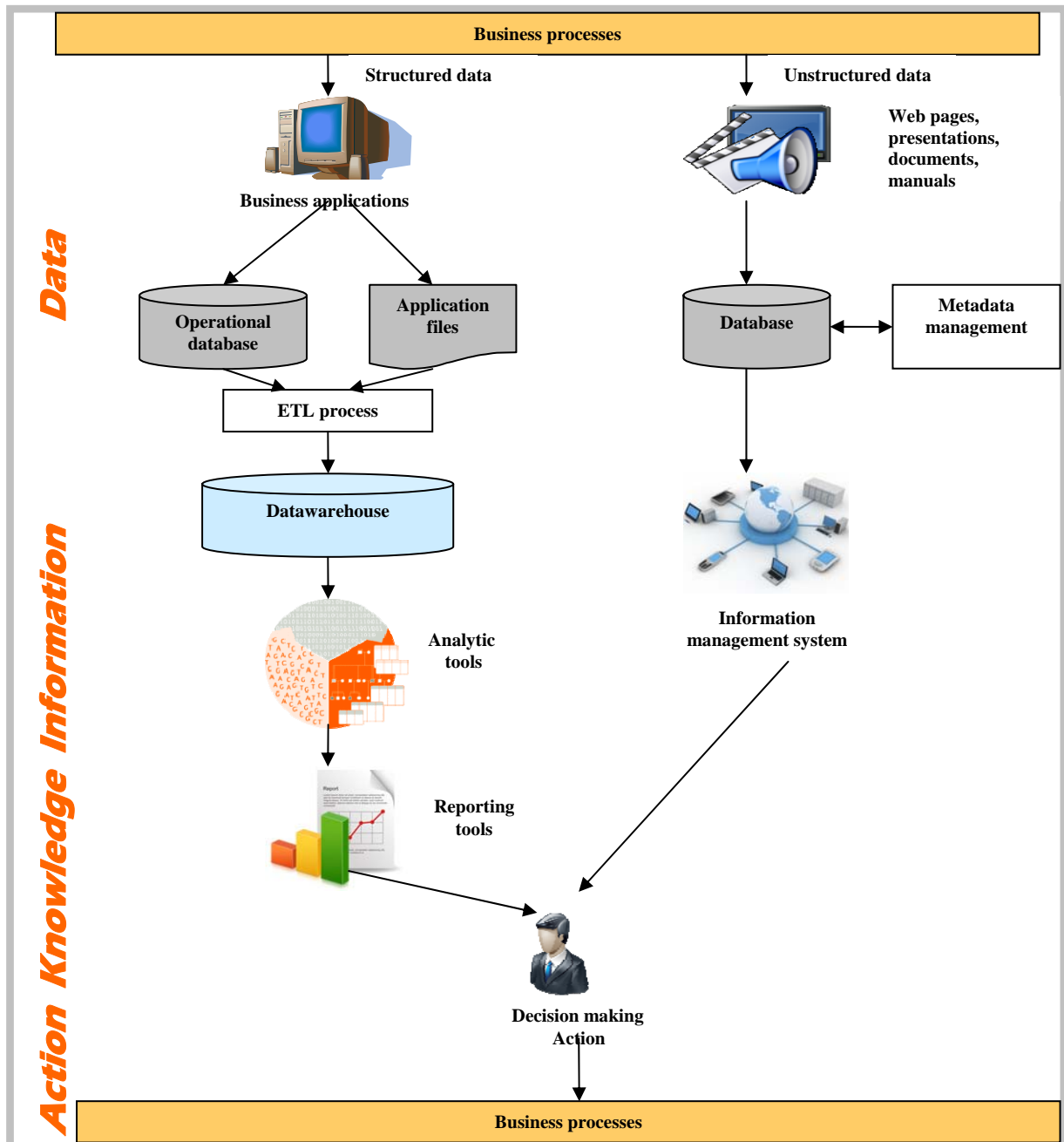


Figure 1. Business intelligence today

2 Software agents and multi-agent systems

Artificial intelligence has been used in economic applications since 80's, generating a lot of interest, but not imposing itself at that moment. During

the last years, artificial intelligence made fast progresses, and the success of neural networks and expert systems conducted to the final acceptance of artificial intelligence in the corporate environment. Neural networks, for example, are already

considered a dedicated method for pattern recognition, especially used for image data type and for complex data sources, being included as base technique by most of data mining tools on the market (e.g. The data mining solution offered by SAS, IBM, Oracle, SPSS all include neural nets as a modeling option).

Usually, artificial intelligence is used for solving complex problems or as a decision support tool (using neural networks or expert systems). For real-time business applications, artificial intelligence includes several techniques that could also be used for improving business intelligence applications:

- Data mining and automatic learning – using neural networks, decision trees, and support vector machines, random forests;
- Evolutionary computation – using genetic algorithms, evolutionary strategies, swarm intelligence;
- Other techniques, like software agents or fuzzy logic.

Our purpose is to investigate how multi-agent systems and software agents have been applied in BI and how can their potential be used for business intelligence systems improvement.

Software agent technology was one of the fields that experienced the fastest development during the last years. The economic potential of agent based systems was identified at the beginning of 90's, so, nowadays there is an extraordinary explosion of agent-based applications or multi-agent based systems developed for a diversity of fields: e-commerce, supply chain management, resource allocation, intelligent production, industrial control, information finding and filtering, collaborative work, mobile commerce, decision support, simulations, production planning and control.

The researches that approach various aspects of software agent technology and its applications have registered very fast progresses and it promises to be only the

beginning. It is possible that in a close future every software application will include agents or agent communication capabilities. Software agents can play multiple roles in economic applications: taking over repetitive tasks, customizing interaction information, user notification when important events occur, user behavior learning, context-based user assistance, remote task execution etc. The obvious generated advantages of using software agents are: reducing the information overload and the amount of work by using software agents instead of humans, cutting down transactional costs, customized services, off-line work mode that is vital for distance working agents, restricted by technical limitations.

The definition of the agent term is very controversial, as it was used in many other computer branches besides artificial intelligence. In artificial intelligence we distinguish two main approaches: the first one is based on the agent notion as assigning of a behavioral identity to a software component, the second one is based on describing the attributes the agent possesses. A representative definition is given by Pattie Maes [7]: "autonomous agents are computational systems that can inhabit a complex, constantly changing environment, sense what is going on, and act independently to accomplish a specified set of tasks or achieve certain goals".

Intelligent agents are defined by IBM [14] as "software entities that carry out some set of operations on behalf of a user or another program, with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of user's goals and desires". Wooldridge and Jennings [13] consider that an intelligent agent is a hardware or a software agent that has the following features:

- Autonomy – the agents operates as an independent process, with no direct human intervention and has control over its actions and internal state;
- Reactivity – the agent perceives his environment (physical world, Internet, a

collection of agents or even a user through a graphical interface) and promptly answers to the perceived environment changes.

- Pro-activity – the agent not only that reacts to its environment changes, but it is also capable of manifesting a goal oriented behavior, by taking the initiative.

- Social abilities – the agent interacts with other agents (including humans) using an agent communication language.

In the context of intelligent agents, there are other two interesting properties to be mentioned:

- Self-analysis – the agent is capable of analyzing and explaining its behavior and of detecting its errors or its success;

- Learning, adaptation and improvement by interaction with its environment.

A multi-agent system can be defined as a loosely-coupled network of entities that work together to solve a problem that cannot be solved by an individual agent. These entities can show self-organization and complex behavior, even if the individual agent's strategies are simple.

The utility of the agent's special features depend on the way the agents are applied, as they are recommended for specific types and fields of applications. Contributory factors are:

- Complex real problems;
- Open, distributed system modeling;
- Limited capacity of a single agent to solve real-world problems;
- Data control and expertise distribution;
- Old software packages reuse and their integration into new systems;
- Agent-based application modeling by organizing the system environment as agent societies that cooperate and compete in solving problems;
- Need of reusing network distributed resources and expertise;

- Increase of system performance: calculation speed, extensibility, reliability, flexibility, maintenance, response quality.

3 Multi-agent systems applied in Business Intelligence

The specialty literature is not very generous when it comes to agent-based business intelligence applications.

Erik Thompson, specialist at Hyperion Solution Corporation defined in 2002 five potential impact fields for intelligent agent on traditional analytic systems [12]:

- i. Agents should contribute to BI solution evolution from application-oriented solution to process-centric solutions and to offering a single access point to distributed information. Agents-based systems must have access to a self-description of the individual modules and the user's personal agent can interrogate for specific information all library agents reasonable for various data sets.

- ii. There is necessary a software-user dialog so that it can learn what the user's wishes are and can anticipate his future desires. The active usage of agents gives BI solutions a more customization and intelligence besides the options and preferences offered by the application.

- iii. Intelligent agent can offer customized analytical assistance for high level business processes, observing them, learning about them and interacting with uses. Agent-based applications and a BPM (Business Process Management) platform could be involved in encoding horizontal analytic knowledge and domain specific knowledge.

- iv. For most Bi applications, the server side is very important because of the changes that occur in the loading patterns and it has a high number of physical settings. Intelligent agents for physical optimization can evaluate their own physical organization and, if necessary, interact with a system administrator before reorganizing data

In the following paragraphs we will shortly

present some of the rare approaches existing in the specialty literature.

In [8] it is presented a model of automatic reporting base on push agents as the next step in the evolution of reporting models, following the end-of-period reporting and the dynamic query reporting. The main characteristic of those three models are presented in Table 1:

Type of reporting	End of period	Dynamic query	Push/pull agents
Timeliness of data	Stale	Near real time	Near real time
Potential for managerial filtering	Yes	No	No
Delivery of choice	No	Not typically	Yes
Alerts	No	Not typically	Yes
Silo issues	No	Possibility	No

Table 1: A comparison of reporting models

In order to develop such a model, a company should define some key performance indicators (KPI) it wishes to monitor against to their planned level. When detecting changes in these indicators, software agents should automatically decide if they should make automatic recommendations/ warnings for corrective actions, sending messages to the appropriate manager through the chosen communication channel (email, voicemail, phone message etc). The main benefits underlined by the authors in using the proposed agent-based topology for business process modeling are:

- Prompt reporting;
- Automatic measurement of planned objectives;
- Unfiltered information;
- User preferences based delivery.

In [3], there are presented three potential domains for applying agent based systems in business intelligence and it is proposed an intelligent agent-based BI system for customer credit ranking in a bank. The three domains are:

1. Intelligent acquisition – for collecting unstructured data from internal distributed agents that are capable o recognizing information in semi-structured documents and autonomously search possible information sources.

2. Intelligent modeling – for creating simulation environment models in order to predict future model states, using agent-based systems, data manipulation and intelligent rules.

3. Intelligent delivery – proactive information delivery of selected information using appropriate channels. There are used brokerage agents that identify information importance and correlation and autonomously decide on the delivery channel.

In [2] information software agents are proposed for collecting public data that are freely available on Web in order to support the decision-making process. Web information agent is capable of collecting information and organizes it as a local database, process that involves four steps: accessing data from real-time data sources or from historical data sources (achieved data that use an indexing system), collecting data, and verifying and correcting records consistency, preparing data and storing data locally. The steps are repeated iteratively until all the received data are stored. More than that, the agent ca monitor data changes and can update these changes daily. Such an agent was used for collecting data about stolen cars from the Croatian Ministry of Internal Affairs Web site. Insurance companies could benefit from the collected data in order to estimate risk more realistically.

In [4] a dynamic based infrastructure is developed. It is Java based, platform neutral, light-weight and extendable. It is Java based platform neutral, light-weight and extensible. It is different from other infrastructures and from the client-server model as it supports dynamic change of agent's behavior. Problem solving capabilities are obtained by dynamically loading java classes representing data,

knowledge and application programs. It was created for facilitating an easier design and implementation of autonomous, cooperative, adaptive and mobile software agents.

This infrastructure has been prototyped for developing event-driven BI applications and there were implemented several examples of such applications. The infrastructure has proved to be very efficient facilitating solution development, implementation and adoption.

4 Business intelligence in Supply Chain Management using software agents

Our approach is different from the other ones, as we intend to add real-time business intelligence features to an existing agent-based system for supply chain management. Supply chain management is one of the fields that have successfully applied agent technology, because its automation requires a distributed point-to-point architecture that must also have some special properties [5]: disintermediation – direct association between users and their software, dynamic composability and execution – the software infrastructure must enable resource discovery and composition at runtime, interaction – specific interaction patterns must be explicitly represented and reasoned with, as specific interactions might be unknown until runtime, error tolerance and exploitation – the system should be able to anticipate and compensate for errors.

Supply chain is a network of facilities for obtaining raw materials, conversion into intermediate goods and final products and then delivers the products customers through a distribution system [6].

Supply Chain Management aims at managing the flow of information, materials, services and money for an activity so as to maximize the efficiency of the process. Development of Internet and communication standards offers great

opportunities to connect the supply chain of suppliers and customers in a vast network, and thus optimize costs and opportunities for everyone involved.

In recent years there are many architectures for intelligent agents based on supply chain management, the set of activities efferent supply chain is modeled using agents that interact and communicate with each other and the environment to fulfill their individual tasks, but also considering the objectives of the entire system.

Activities like planning, programming, negotiation, multiple interactions make multi-agent system particularly appropriate for this area.

Some examples of complex activities, which usually involved several software agents as:

- Getting orders from customers, negotiating terms and conditions or other special requirements;
- Coordination of suppliers, production and distribution so as to obtain optimum results and maximum satisfaction of all parties involved.
- Planning and implementation of resource transport so as to avoid any delay and to minimize costs and transport times.
- Planning and re-planning activities considering the time of delivery, available resources, interdependencies between activities, delays and other specific criteria.
- Inventory management and material requirements planning starting from the expected demand or the planned production, etc.

For each of these activities, the need for real time decision is obvious. For example, agents should consider and decide the selection of suppliers need to review the activity of suppliers that has previously worked with the company, to analyze the frequency of delays or lacks of common stock, long-term trend of price increases, all that compared to other suppliers and making calls to the specific multidimensional operations (drill-down, roll up, slice / dice). Another example is the analysis of a specific

product sales activity compared to other products, taking into account various criteria (geographic, time, clients etc), tracking the trend for profits generated by it, and identify the causes that led to current situation. All these could help in accepting orders from customers, and negotiating with them and in deciding above the optimal production mix.

We propose a multi-agent system for managing supply chains based on Case Based Reasoning agents. Case Base Reasoning (CBR) solves new problems by retrieving previous solutions to similar or close problems that are stored in a database of cases. Therefore, CBR is based on previous experiences and patterns of previous experiences, similarly with human thinking. Each case

is described by a set of features or attribute-value pairs. Case retrieval algorithm is often a simple K-Nearest neighbor algorithm.

CBR is typically used when there isn't any model or method to match exactly to the problem requirements proposed by the user. The model or method is constructed by applying reasoning on similar cases. When a problem is solved, the solved case (scenario and solution) is retained by updating the case database. Therefore, CBR is not a special method of reasoning, but is also a paradigm of machine learning which allows the accumulation of lifelong learning through experience.[11].

Figure 2 represents a generalized CBR cycle, including four main steps: Retrieve, Reuse, Revise, and Retain.

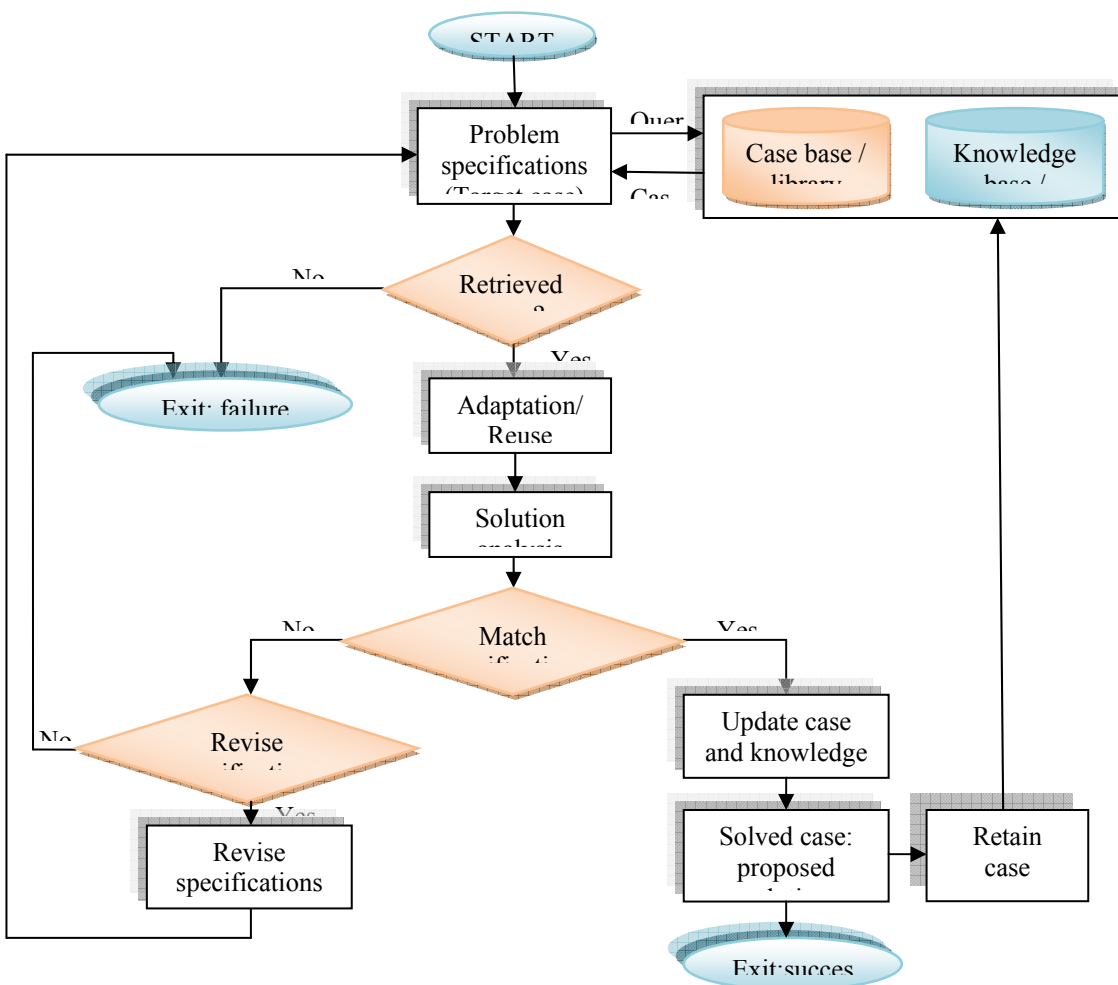


Figure 2. Case-based reasoning process

The proposed multi-agent system should be formed of three intelligent agent-based subsystems (Figure 3):

a. Extracting and uploading data management sub-system – It could be two parallel systems designed to ensure that decisions in short time: a classic one that actually store data in a traditional data warehouse and a second data virtualization system that accesses data directly from multiple data sources. This agent-based subsystem should decide which of the systems to be used and should implement specific mechanisms for extracting and loading data. The decision of adopting a virtualized solution should be carefully analyzed. BI projects usually involve complex multidimensional analysis and a great importance is given to data cleaning and consolidation. The number and dimension of data sources, the quality of the raw data and the analysis requirements should also be considered. In many cases, the best integration solution is a combination of virtual and physical approaches, keeping the physical data warehouse in order to benefit from its features and applying virtualization for cutting costs and getting quicker results for data source access, for data mart elimination, for prototyping new data marts or data warehouses, for federating multiple physical consolidated sources and so on.

b. Data mining sub-system – In SCM there are always discrepancies between supply and demand, and DM can help predict the degree of incertitude in the SCM. DM can thus be considered an essential tool in understanding the behavior of customers, in pricing, in promotion planning and product development. For example, for manufacturers, DM can make predictions of supply uncertainty, predictions of process uncertainty due machine failure, poor operation and maintenance plans, predictions of supply uncertainty using various criteria like article, distributor,

location, forecasting of future trends of supply, etc. [DM1]

As we know, DM techniques for complex environments should be very dynamic, because changes in the system can affect the overall system performance. Due to the need for data mining on distributed sources it appeared distributed data mining. MAS often solve complex applications that require distributed problem solving because they often have distributed and proactive and reactive properties that are very useful for these cases. On the other hand, in many applications, individual and collective agent behavior depends on observed data from distributed data sources. Therefore, combining distributed data mining with MAS data-driven applications is very attractive [9]. Agents can be taught through case base reasoning to automate the processes of data mining: to configure, set, test results and if results are unsatisfactory resume.

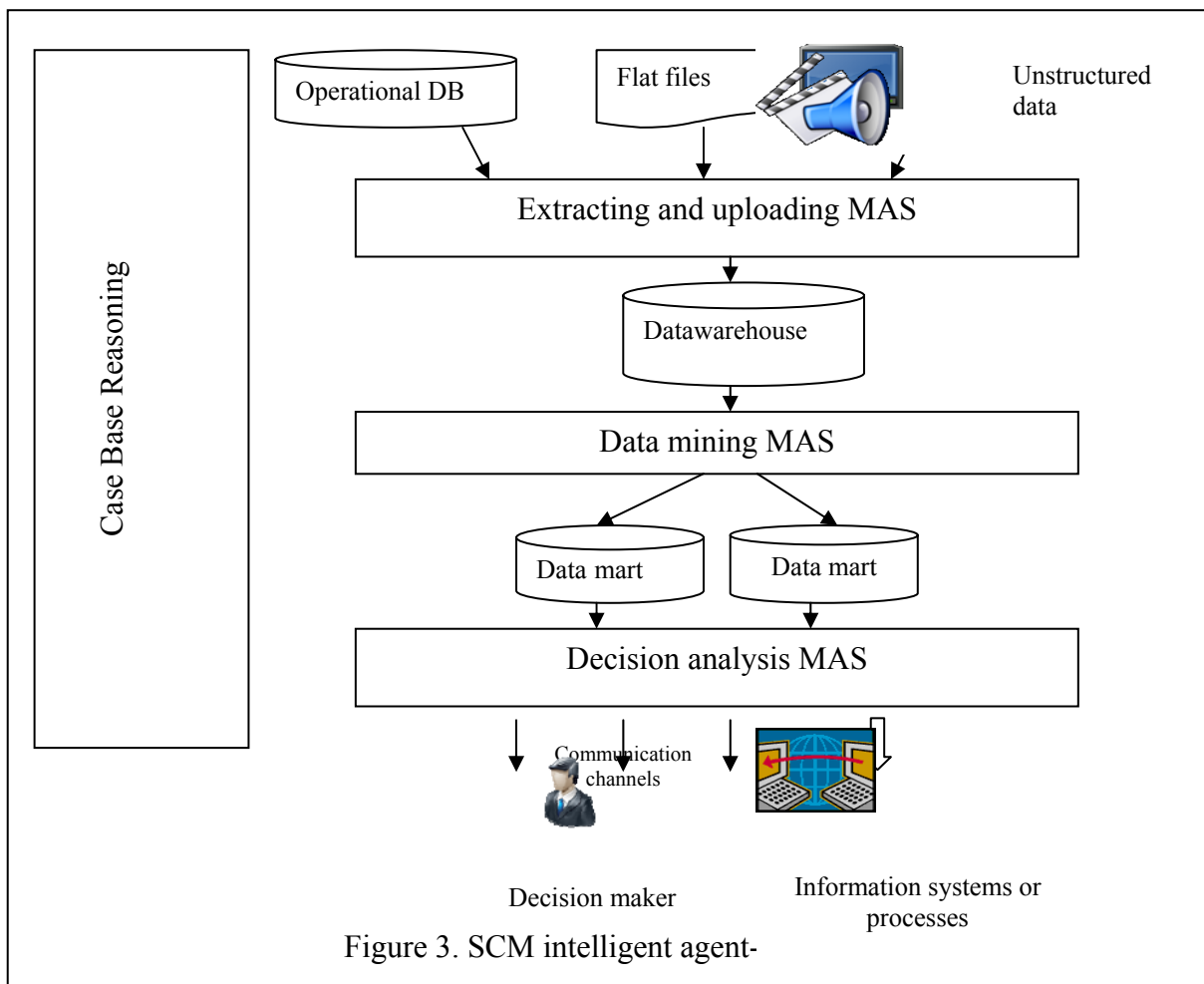
c. Decision analysis sub-system – This sub-system involves the use of agents that rely on CBR for each case stored in the cases database consisting of a set of analytical operations, aimed at finding a solution to a scenario of a problem. For the description of the scenario and solution to the problem there will be used some metrics or indicators such as KPI (key performance indicators). Some examples of KPI are: functional costs (storage, transportation etc), channel costs (institutional, retail, government, etc.). A KPI triggers a response that can be automated using CBR., seeking a match in the case base based on the KPI's similarity and retrieving those cases that are most similar to the current scenario. Any similar case will be a sequence of analytical operations. [1]

The results of the analysis would be presented by assistant agents to interested persons using appropriate communication channels. A second possibility would be automatic triggering of corrective actions in the chain of automated processes.

The analysis results could be transformed directly into action, if necessary. Software

agents could act directly on processes, but involves a good modeling and

automation of business processes, which is very rare, at this time.



5 Conclusions

The integration of business intelligence and software agents can provide solutions to some business intelligence problems. For example, business intelligence solutions often face difficulties when working on distributed data sources, possibly located in unstructured documents.

To highlight the potential of software agents we started from the specific case of using agents with very good results for Supply Chain Management. It was proposed a system based on software agents for implementation in real time (or as close to real time) of business intelligence in SCM, using CBR agents. It was proposed an intelligent agent-based solution based on three subsystems, emphasizing the benefits of

an agent-oriented solution.

It follows that in a future paper this architecture to be developed and detailed to the level of functional prototype.

References

- [1] B. Azvine, Z. Cui, D.D Nauck, B. Majeed, "Real time business intelligence for the adaptive enterprise", *The 8th IEEE International Conference on ECommerce Technology and The 3rd IEEE International Conference on Enterprise Computing ECommerce and EServices CECEEE06*, 2006, p. 29;
- [2] M.P. Bach, N. Vlahovic, B. Knezevic, "Public Data Retrieval with Software Agents for Business Intelligence", *Proceedings of the 5th WSEAS Int. Conf. on Applied Informatics and*

- Communications*, Malta, September 15-17, 2005, pp.215-220;
- [3] S. Bobeck, I. Perko, "Intelligent agent based business intelligence", *Current Developments in Technology-Assisted Education*, FORMATEX, pp.1047–1051, 2006;
- [4] Q. Chen, P. Chundi, U. Dayal, M. Hsu, "Dynamic Software Agents for Business Intelligence Applications", *Proceedings of the Second International Conference on Autonomous Agents*, 1998, pp.453-454;
- [5] M. N. Huhns, L. M. Stephens, "Automating Supply Chains", *IEEE Internet Computing*, July-August 2001, pp. 90-93.
- [6] H.L. Lee, C. Billington, "The Evolution of Supply-Chain-Management Models and Practice at Hewlett-Packard", *Interfaces* 25: 5 September-October, 1995, pp.42-63;
- [7] P. Maes, "Artificial Life Meets Entertainment: Life like Autonomous Agents", *Communications of the ACM*, 38, 11, pp. 108-114, 1995;
- [8] M.S. Raisinghani, J.H. Nugent, "Intelligent agents for competitive advantage", book chapter in *Business Intelligence in the Digital Economy: Opportunities, Limitations and Risks*, Idea Group Publishing, 2004;
- [9] V.S. Rao, "Multi Agent-Based Distributed Data Mining: An Overview", *International Journal of Reviews in Computing*, © 2009-2010 IJRIC & LLS, pp. 83-92;
- [10] J. Sebestyénová, "Case-based Reasoning in Agent-based Decision Support System", *Acta Polytechnica Hungarica*, Vol. 4, No. 1, 2007, pp.127-138;
- [11] M.K. Singh, M.S. Raisinghani, "Data mining for supply chain management in complex networks: concepts, methodology and application", published in *Developing quality complex database systems*, IGI Publishing Hershey, PA, USA, 2001, ISBN:1-878289-88-8 ;
- [12] E. Thomsen, "Agents uncovered (intelligent agents)", *Intelligent Enterprise*, September 17, 2002;
- [13] M. Wooldridge, N.R. Jennings, "Agent Theories, Architectures and Languages: A Survey", *Intelligent Agents*, Wooldridge și Jennings, Berlin, Springer-Verlag, 1995;
- [14] [http://activist.gpl.ibm.com:81/White Paper/ptc2.htm](http://activist.gpl.ibm.com:81/WhitePaper/ptc2.htm);
- [15] http://en.wikipedia.org/wiki/Real-time_business_intelligence.



Ana-Ramona BOLOGA (born in 1976) is lecturer at the Academy of Economic Science from Bucharest, Economic Informatics Department. Her PhD paper was entitled “Software Agents Technology in Business Environment”. Her fields of interest are: integrated information systems, information system analysis and design methodologies, and software agents.



Razvan BOLOGA (born 1976) is lecturer at the Academy of Economic Studies in Bucharest Romania. He is part of the Computer Science department and his fields of interest include information systems, knowledge management and software ecosystems. Mr. Bologa has attended over 15 conferences presenting the results of his research.

Applications of Spatial Data Using Business Analytics Tools

Anca Ioana ANDREESCU, Anda VELICANU, Daniela MITROESCU
 Economic Informatics and Cybernetics Department, Academy of Economic Studies,
 Bucharest, ROMANIA
 anca.andreescu@ase.ro, anda.velicanu@ie.ase.ro, mitroescu_daniela@yahoo.com

This paper addresses the possibilities of using spatial data in business analytics tools, with emphasis on SAS software. Various kinds of map data sets containing spatial data are presented and discussed. Examples of map charts illustrating macroeconomic parameters demonstrate the application of spatial data for the creation of map charts in SAS Enterprise Guise. Extended features of map charts are being exemplified by producing charts via SAS programming procedures.

Keywords: *spatial data, business analytics, SAS system, map charts*

1 Introduction

Historically speaking, since the '70, methods of statistical calculation and data processing have begun to change with the introduction of integrated statistical packages. They allow users to perform one or more statistical procedures on a single set of data stored in a file format standard. Processing and data management have been hardly started and cumbersome and mapping packages were missing from the standard system [1]. Given the growing necessity of representing geographical distributed information we will discuss in the paper how analytics tools deal with geographic data and allow graphical descriptions based on this data.

2 Spatial data structure

The spatial data (geographical data) emerged from the need to work and process geographical data like points, lines, polygons, three-dimensional geometric objects representing real world objects. Spatial locating this data can be bi-or tri-dimensional, depending on its type and the information available. Such a two-dimensional level can indicate the latitude and longitude coordinates or x and y coordinates of a given finite area and a three-dimensional level can specify latitude, longitude and height / altitude or the x, y and z coordinates in a given finite area. Examples of spatial objects are

cities, buildings, rivers, touristic points of interest etc. This article will mainly deal with issues regarding bi-dimensional spatial databases.

Spatial objects consist of points, lines, surfaces, volumes and objects of higher dimensions that are frequently used in computer aided design applications, cartography, GIS, etc. They are described by spatial attributes (length, configuration, perimeter, area, volume, etc.) and by non-spatial (common) attributes (the creating date, ownership, membership of a superior structure, etc.). The values of spatial attributes of these objects represent spatial data.

Objects in the real world can be divided into two categories: discrete objects (a house) and continuous fields (rainfall or altitude).

Geographic data represent real world objects (roads, soil, elevation) by storing data with two or three measurable dimensions relative to a Cartesian coordinate system or latitude-longitudinal system.

Spatial data can be divided into point data and regional data. The point data is completely characterized by its location in a multidimensional space. It can come directly from measurements or can be rendered in order to be more easily stored and retrieved.

Regional data is characterized by location (a fixed point in a region) and destination (a line in 2D or a 3D surface).

Spatial data are also those of a certain rank (e.g. "Cities in Europe"), approaches data (e.g. "Three close lakes in Romania"), junction data (e.g. "Pairs of cities in Romania located at 50 km of each other").

Spatial data structure is specific for storing, querying and processing the geometric information. This allows us to provide the necessary information for a database management system (DBMS) so that it can follow the spatial data model. In general, a geographic data structure must allow the representation of three components:

- data on the geography of objects (points, lines, polygons);
- data on objects' attributes (name, description);
- images (used usually as background maps or as symbolic representation of objects – e.g. a green cross for pharmacies, a triangle for campsites, a tree for forests etc.).

Raster and vector data models based on spatial data representation are considered to be *dual* by [2].

The two main models used for storing such data - raster and vector - are detailed in [3]. Raster model is based on partitioning the areas and the vector model works with points, lines and

polygons that describe the geometry of a spatial object. [4] identifies five levels of significance for the representation of real world entities in data structures that can be implemented in the computer:

- entities level;
- objects level;
- conceptual level;
- raster and vector level;
- data structures level (implementation level).

The difference between entity and object, shown by [4], is that entities can be seen as phenomena that have no clear physical boundary, while the objects are defined by dimensional attributes (length, width, volume).

At a conceptual level, the objects acquire attributes such as distance, coordinate system, and topology. Raster or vector level determines how objects are represented as spatial data. The implementation level depends on the DBMS through which spatial data is stored according to available data structure and the specialized spatial database (SDB).

Table 1 presents a comparative analysis of both data types, in terms of characteristics such as storage, operations' implementation, compatibility with relational databases, updating and analysis capabilities.

Table 1. Comparative analysis of raster and vector data

Data	Storing space	Implementing overlapping operations	Viewing capabilities	Compatibility with relational databases	Updating	Analysis capabilities
Raster	Big	Easy	Images	Little	Difficult	Reduced
Vector	Small	Difficult	Graphics	Large	Easy	Many

In the table above it can be noticed that using raster data has advantages like easy implementation of overlapping operations, methods of rendering images, while using vector data requires less storage space, methods of rendering graphics, large compatibility with relational databases, ease to update and many analysis facilities.

As illustrated in paper [5], creating a professional looking Map Chart in SAS Enterprise Guide is a simple process that has gotten easier with each release of the software. As long as data that can be joined to a map dataset, high quality maps can be created. It only requires one task and two datasets in the simplest form.

The *coordinate system*, also called spatial reference system, is another important notion when talking about spatial data. The coordinate system is a mean of allocating the coordinates to a location and establishing relationships between sets of such coordinates. This allows interpreting a set of coordinates as a representation of a location in the real world. Each spatial data has an associated coordinate system. The coordinate system can be:

- with a geo-reference (geodesic, depending on a certain representation of Earth);
- without a geo-reference (Cartesian, not depending on a certain representation of Earth).

If the coordinate system is geodesic, then it has an associated default unit of measurement (e.g. meters), but can have automated transformation of data into other units (e.g. miles). Spatial data can be associated with Cartesian, geodetic (geographical), projected or local coordinate system.

Cartesian coordinates are those which measure the position of a point along the axes that are perpendicular in the origin, representing two-dimensional or three-dimensional space. If a coordinate system is not explicitly associated with geometry, it is assumed that the geometry is in Cartesian coordinate system.

Geodetic coordinates are longitude and latitude, closely related to spherical polar coordinates and are defined relative to a certain geodesic data.

Projected coordinates are plane Cartesian resulting from performing a mathematical mapping from a point on the Earth's surface onto a plane. There are many such mathematical mappings.

Local coordinates are Cartesians in a non-geodetic coordinate system. They are often used for CAD (Computer Aided Design).

When performing operations on spatial data a Cartesian model or a curvilinear pattern must be used, according to the

coordinate system associated with the data.

In SAS Enterprise Guide the coordinate system can be defined after the spatial data is imported from a database or a data file (xls, html etc.).

For systems that store and process spatial data in spatial databases it results in a number of advantages, primarily due to their specialization (dedicated system). Thus, the general advantages of working with geographic data are shown in [6]:

- precision in representing the data and processing it;
- creating and using specific libraries (information structuring, using standards, interchanging information);
- possibility to update information;
- quick access to information (due to flexible organizing criteria and to fast working speed of the equipment);
- easily, precise and unlimited reproducing of graphic material (paper copies can be obtained quickly and also spreadsheets and scale drawings can be done considering the user's choice);
- adaptability to evolution of informatics technologies.

If data to be used is not yet in digital format, there are several techniques by which spatial information can be obtained.

Thus, maps can be digitized or drawn with the mouse for collecting the coordinates of various items. Scanning with electronic devices can also help convert the lines and points on a map into digital data.

Entering data into the system is requesting the largest amount of time resource. Each occurrence of objects in a map and spatial relations between them must be specified.

Editing the automatically retrieved information can also be difficult. Electronic scanners record the map patterns with the same accuracy with which they retrieve important elements from the map. For example, such a pattern can lead to connecting two lines that should not meet. Such unwanted information be edited or removed from the data file.

3 Producing map graphs with SAS Enterprise Guide

Being one of the largest software producers for statistics and business analytics in the world, SAS paid particular attention to the representation of spatial data in its solutions. SAS Enterprise Guide is the Graphical User Interface (GUI) that helps to exploitation and dynamic publication in a Microsoft Windows client application. SAS interface is the preferred solution for business analysts, programmers and statisticians.

SAS means analysis and data management, a series of procedures and programming capabilities. An important feature offered by SAS statistical processing of cartographic data based on geographically and able to juggle with these statistics to obtain cartographic works. Map coordinates are stored in tables provided to streamline the process. So, in order to create a map, the user needs tables for storing the owned information that will be implemented through a mapping process. Further, the imported data are joined (intersection) in Enterprise Guide with the coordinates provided by the system, and this will generate the final mapping product. SAS offers users the possibility to choose the type of map produced from a variety of devices widely available, simple maps, maps with parameter chosen by the user, the possibility to redefine the geographical regions, the possibility to choose maps colors [7].

The following paragraphs will present the process steps necessary to create a cartographic work. The analyzed data are actual values of macroeconomic indicators, studied for the countries of the international economic organizations, the Organization for Economic Co-operation and Development (OECD).

The first step is to import tables containing data. Different formats of tables can be imported, including xls, mdb, csv and html. In the example, we

have chosen to import tables in mdb format. To create a parameter map with all the values we need in a single table of indicators including Gross Domestic Product (GDP), Net Domestic Product (NDP) and UNEMPLOYMENT, so the tables imported have been joined. To join two or more tables Query Builder tool have to be used and one can choose the type of join (manually or automatically).

To create a map using SAS Enterprise Guide 4.2, we need to get the geographic coordinates owned by the system in the *map* library. A join was made between the imported table, which contains indicators for each country, and the table with the coordinates of the system. Next, a map is created for the data we get by processing imported tables. As settings we can choose the type of the map (Choropleth, Riser and Prism), data sources, appearance, titles and other properties. To set up a map with a parameter that can be chosen at runtime, a variable that will contain the list of indicators was created. Creating a map in SAS Enterprise Guide 4.3 is even easier than in version 4.2 of this product. Briefly, the key element to create a Map Chart is to making sure your data and the map file have a commonly named variable on which to join. As mentioned before, earlier versions required to use the Query Builder task to join your data to the shape or map file. In 4.3, this is done automatically. In order to create a basic Map Chart, the user must only provide data with a geographic variable and an analysis variable [5].

In figure 1 is presented a two-dimensional (*choropleth*) map chart which indicates levels of magnitude or response levels of the corresponding response variable by filling map areas with different colors and patterns. For each OECD country, one level of the Gross Domestic Product for the year 2008 is described using various shades of the same color. An example of a 3D map chart can be found if figure 2. This is a *riser* map graph having riser appears to be standing on each geographical area. The height of the riser is proportional to the value of the chart column

for that geographical area. The same information as in figure 1 is represented in this chart.

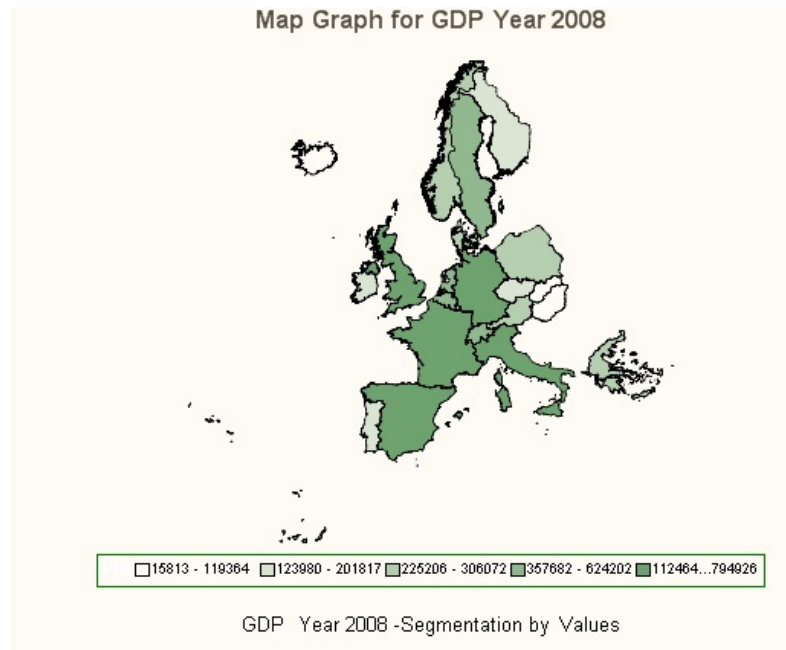


Figure 1. Choropleth map chart

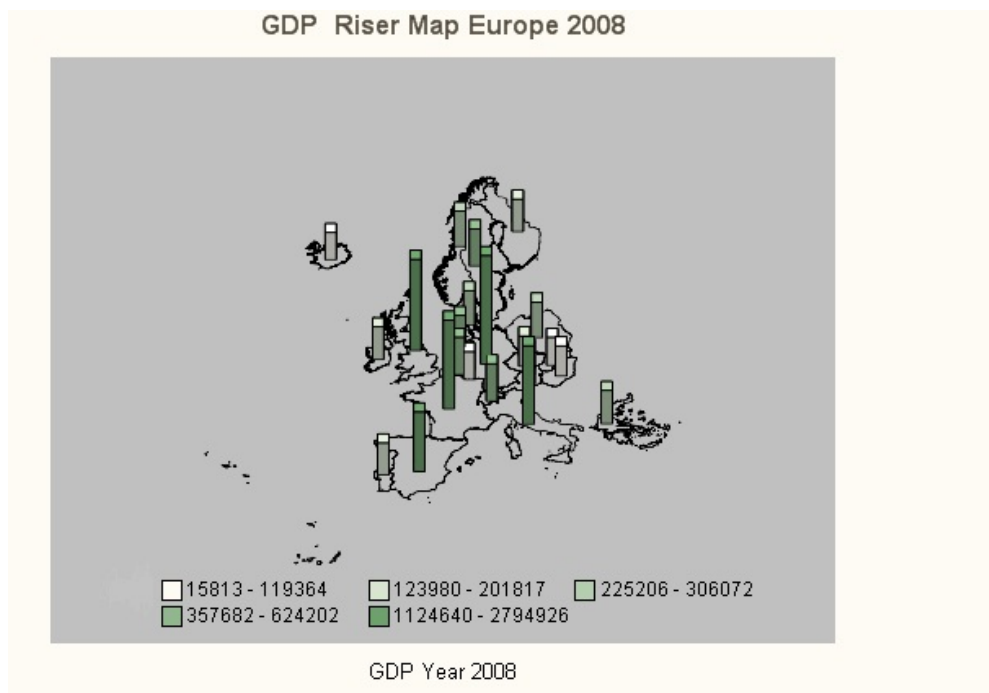


Figure 2. Riser map chart

4 Extended features of map graphs in SAS

Map graphs produced by SAS Enterprise Guide are static and their lone purpose is to display information. New features within SAS/GRAPH allow for the creation of maps with dynamic features

like drill-down areas, pop-ups and animation. This implies writing SAS code for achieving such features.

Starting from the examples illustrated in [7], we demonstrate the use of the ACTIVEX device driver that enables a number of facilities, including the ability to pop-up

information associated with each map area. In this regard, two data sets will be used to create a map graph which will display the population density for all Romanian counties. The first (named *romanian_dens*), containing the response data set, has three variables: ID – identification number for a county, County_name and Density – the density of population per square kilometer in the year 2006, information retrieved from [8]. The second data set (named *Romania*) is the Romanian shape file provided by SAS system and located in the *map* library.

The SAS code from the program below generates a map graph of the Romanian population density in the form of an html type file.

```

*** Section 1 - create a format to
display county name;
data county_fmt (keep=fmtname start
label)
/ view=county_fmt;
retain fmtname 'countyname';
set romanian_dens
rename=(ID=start county_name=label));
run;
proc format cntlin=county_fmt;
run;

*** Section 2 - set graphics and
output options;
goptions reset=all
device=activex
xpixels=700
ypixels=500;
ods listing close;
ods html file = 'c:\romania.html';
title 'ROMANIAN POPULATION
DENSITY By COUNTY, 2006';
footnote j=r 'Pop-ups in Map Graph';

*** Section 3 - define graph
parameters;
proc gmap data=romanian_dens
map=maps.romania;
id ID;
choro density / levels=all nolegend;
format ID countyname.;
label ID = 'COUNTY';
run;

```

The following paragraphs explain how the SAS language features were used in the source code presented above.

In Section 1, the *data* step creates a user format (*county_fmt*) that will be used in

proc gmap command to display information associated with each Romanian county. This will be done starting from the *romanian_dens* data set (specified in the *set* command). The *keep* flag determines whether or not a variable is written to the output data set. The specified variables will be the only variables in your output data set [9]. The *view* option tells SAS to compile, but not to execute, the source program and to store the compiled code in the input DATA step view that is named in the option [10]. *Retain* flag is used to prevent *fmtname* variable from being reset to missing at the top of the data step. This makes it a vital tool for passing information from one observation to another.

An important feature of SAS language is the ability to create user-defined formats starting from a specially constructed SAS data set. This can be done using the *proc format* option *cntlin=* (control in) that permits the user to name the data set (or data view in our case) which will be used. Control input data set require a minimum of three variables, named *fmtname* (the format name), *start* (a single value that will be formatted) and *label* (information to be displayed) [11].

Section 2 has the roles of resetting the graphics options to default values, selecting the JAVA device driver and setting the image size. Also, using the Output Delivery System (*ods*) a html file is generated and characteristics of this output type (title and footnote) are being defined.

Section 3 involves the actual construction of the graph through the *gmap* procedure. The GMAP procedure produces two-dimensional (*choropleth*) or three-dimensional (block, prism, and surface) maps that show variations of a variable value with respect to an area [12]. The only type of map graph that SAS can create using only code, not Enterprise Guide tasks is the surface graph type. Main parameters of *gmap* proc are response data set (*data*), map data set (*map*), variable that determines the color intensity of the geographic areas (*id*) and type of graph (*choro*, in the presented example). In

the resulting map, as it can be seen in Figure 3, information about each area (the value of the ID variable and the value of the variable in the CHORO statement) appear in pop-up box as the

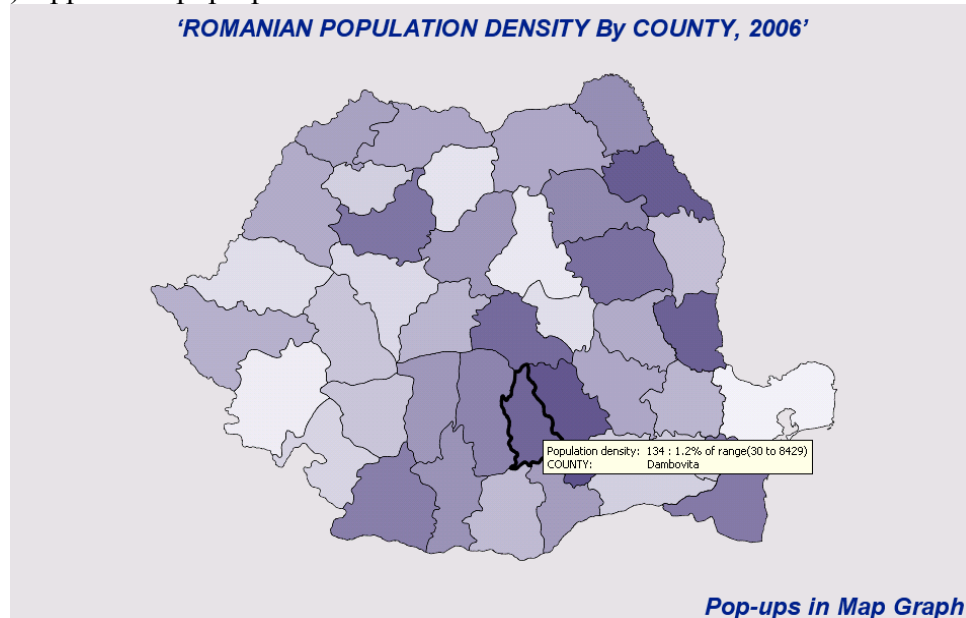


Figure 3. Map chart with pop-up features

5 Conclusions

The possibility to represent spatial data in statistics and analytics software tools is an important characteristic in the context of the new economy, mainly due to globalization and geographic distribution of organizational units in various economic areas. SAS software offers big support in this regard, by letting users to manipulate different kind of spatial data and to create map graphs. Besides the support offered through the map library, additional maps can be found on the SAS Maps Online page available at [13]. Besides this, a connection with professional spatial analysis tools can be made using, for example, SAS Bridge for ESRI to access ArcGIS files.

References

- [1] G. B. Meehan, *Statistical Mapping Capabilities and Potentials of SAS*, Institute for Research in Social Science, University of North Carolina, 1982
- [2] D. Peuquet, *Representations of Geographic Space: Toward a Conceptual Synthesis*, Annals of the Association of American Cartographers, Nr. 78 (3), 1988, pp. 375-394.
- [3] *Geographic information system* http://en.wikipedia.org/wiki/Geographic_information_system#Data_representation
- [4] R. McMaster, L. Barnett, *A spatial-object level organization of transformations for cartographic generalization*, Proceedings of the International Symposium on Computer – Assisted Cartography Auto-Carto 11, 1993, pp. 386-395.
- [5] S. R. Thompson, *Easier than You Think: Creating Maps with SAS Enterprise Guide*, SAS Global Forum 2011, Las Vegas, Nevada, USA, <http://support.sas.com/resources/papers/proceedings11/311-2011.pdf>
- [6] M. Băduț, *Soluție pentru proiectarea si administrarea întreprinderilor - O tripletă de încredere: MicroStation - MicroStation GeoGraphics - PlantSpace*, PC World Romania, nr 7, 1997, pp. 12
- [7] M. Zdeb, *Creating Maps with SAS/GRAPH- Drill Downs, Pop-Ups, and Animation*, SUGI 29 Conference Proceedings, Montréal, Canada, 2004,

- <http://www2.sas.com/proceedings/sugi29/toc.html>
- [8] *Density of Romanian population*, http://enciclopediaromaniei.ro/wiki/Index:Jude%C5%A3e#cite_note-2
- [9] *An Introduction to SAS Data Steps*, <http://www.ssc.wisc.edu/sscc/pubs/4-18.htm>
- [10] DATA Step Views, <http://support.sas.com/documentation/cdl/en/lrcon/62955/HTML/default/viewer.htm#a001278887.htm>
- [11] R. Cody, *Learning SAS by Example: A Programmers Guide*, SAS Institute Inc., Cary, USA, 2007
- [12] SAS GMAP Procedure, <http://support.sas.com/documentation/cdl/en/graphref/63022/HTML/default/viewer.htm#overview-gmap.htm>
- [13] *SAS Maps Online*, <http://support.sas.com/rnd/datavisualization/maponline/>

Anca Ioana ANDREESCU is university lecturer in Economic Informatics and Cybernetics Department, Academy of Economic Studies of Bucharest. She published over 15 articles in journals and magazines in computer science, informatics and business management fields, over 20 papers presented at national and international conferences, symposiums and workshops and she was member in over nine 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*. Her interest domains related to computer science are: business rules approaches, business analytics, requirements engineering and software development methodologies.



Anda VELICANU has graduated the Faculty of Economic Cybernetics, Statistics and Informatics of the Bucharest Academy of Economic Studies, in 2008. She has a PhD in Economic Informatics and since January 2009 she is a Pre-Assistant Lecturer. She teaches Database, Database Management Systems and Software Packages seminars at the Economic Cybernetics, Statistics and Informatics Faculty. Her research activity can be observed in the following achievements: 11 diplomas, 2 scientific awards, 13 proceedings, 7 articles published in scientific reviews, 4 research contracts, 3 books and 1 research grant. She is a member of INFOREC professional association, secretary of the "Database Systems Journal". Her scientific fields of interest include: Databases, Database Management Systems, Spatial Databases, Programming, Information Systems.



Daniela MITROESCU has graduated the Bucharest Academy of Economic Studies, Faculty of Cybernetics, Statistics and Economic Informatics in 2010, with a diploma on statistical databases. She is currently student in second year at the Economic Informatics Master Program in the Academy of Economic Studies and she is preparing a master degree thesis on the representation on geographical data in SAS. Her areas of interest are: Databases, Business Analytics and Programming languages.



Solutions for the Object-Relational Databases Design

Manole VELICANU, Iuliana BOTHA

Academy of Economic Studies, Bucharest

mvelicanu@yahoo.com, iuliana.botha@ie.ase.ro

The need for databases occurs in the moment when takes place an informatics system development. Moreover, databases are an important step in this process. For this reason, this paper deals with object-relational databases implementation as part of informatics systems development. The practical implementation is made on a decision support system (DSS) prototype, which can be applied in the uncertain and unpredictable environments, like the production and the prediction of the wind energy.

Keywords: *Object-Relational Databases, Informatics Systems Development, Unified Modeling Language (UML), Database Design, Database Implementation.*

1 Introduction

The IT&C fast development in recent decades is accompanied by significant changes in economic informatics. For this reason, the engagement of specialists is aimed at creating user-oriented information systems, which respond to requests promptly and accurately. Thus, over time, it was made the transition from computerized systems that address particular administrative problems, to intelligent systems which assist the decisions, based on knowledge modeling and processing.

Currently, organizations are required to store and process increasing quantities of data, requiring recourse to modern information technology, databases, data warehouses, Internet and, more so in these cases, the knowledge bases of intelligent systems.

Developing object-oriented model was due to inability of the relational model to

successfully deal with very large data volumes, of great complexity, encountered most often in new types of computer applications (multimedia, Internet, XML, spatial applications etc.). However, although OODBMS (Object-Oriented DBMS) appear to meet the needs for better software required by the new economy, markets for their use remains relatively low, the reason most often cited being the difficult query with a large consumption of computational resources.

The advantages and especially the limitations of both relational and object-oriented technology, demonstrate that businesses need the capabilities of both data models. As a result of these emerging technologies, it was developed the *object-relational data model*. This model provides an extension of the relational data model while also support the main concepts of the object-oriented one (Figure 1).

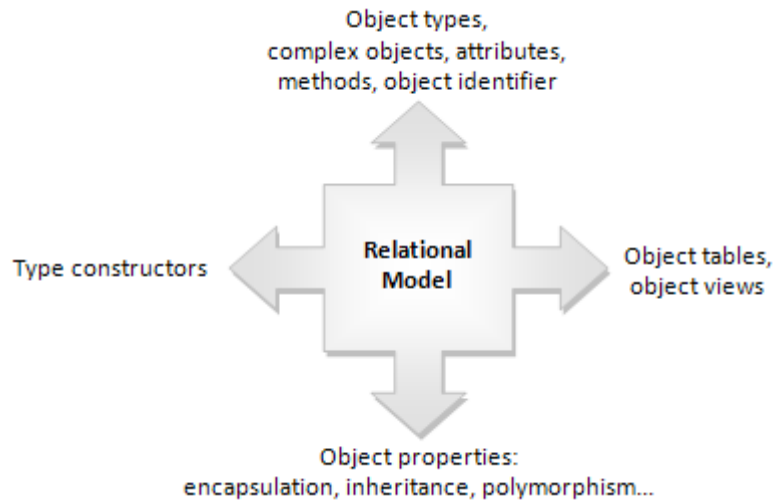


Figure 1 – Extensions of the relational model
(Source: Authors)

While in the 80s the relational databases were crowned as the most widely accepted databases, their limitations lead to a new hybrid database generation, according to the new data model specified above: object-relational databases.

The current technology has created the need for the use and storage of complex object types that were not supported by relational databases. Therefore, the new generation of database systems supports a unified relational and object-oriented data model. The idea is to import, as extensions of the relational model, the object-oriented concepts such as scalability, complex data types (large objects, multimedia data, spatial data, user defined object types etc.) and the fundamental object characteristics like encapsulation, inheritance and polymorphism [5].

2. Design of the object-relational databases

The *object-oriented methods* used for the design of the systems with object-relational databases are based on the concepts of object and classes of objects and allow the use of three different models for designing an object-relational database: the *static model* by which are modeled objects and the relations between them; the *dynamic model* by which are described interactions between objects; the *functional model* by

which are transformed data values using operations and processes.

The advantages of applying object-oriented methods in order to design the object-relational databases can be derived precisely from the facilities of unitary model behavior and data, and capture for the same object static, functional or dynamic characteristics, in terms of interactions with other objects. Thus can be modeled the business needs in interaction with objects.

In conclusion, we find that the benefits of the object-oriented methods in comparison with the structured one, recommend the object-oriented approach in the case of object-relational databases design.

Since object-oriented methodologies and methods have some limitations as well as many differences (in terms of symbols, notations or types of diagrams), it was needed a standard for modeling that can be widely applied in creating new systems or the maintenance of systems.

Thus, in 1997 was developed UML language (Unified Modeling Language), which brings standardization in the representation of symbols, notations, diagrams, and types of models which can be used for modeling a system with object-relational databases. In this way have been removed differences between object-oriented methodologies.

The latest version of UML is 2.3, which defines fifteen types of diagrams, divided into three categories. Thus, eight of them show static aspects, other three present general behaviors, and four represent different aspects of interaction between system elements.

3. UML extensions. Proposing an UML extension for object-relational databases design.

Modeling object-relational databases with object-oriented methods involves the usage of the basic concepts of object-oriented paradigm (classes, objects, encapsulation, aggregation, inheritance, polymorphism, abstraction) in implementing the static (structural) and dynamic model.

As stated in [4] one of the main goals of UML is to provide extensibility and specialization mechanisms by which basic concepts can be extended.

The standard set of UML notations and diagrams can be adapted using the following aspects:

- *Conditions of use.* Object-relational databases are designed especially for complex data storage (such as XML, multimedia, spatial) and must be optimized to achieve a variety of queries on them;
- *The data model used.* Object-relational databases implement the object-relational data model, a hybrid model that combines features of standard data models (relational and object-oriented);
- *Typical operations performed.* Using object-relational databases involves operations with multimedia data, spatial operations, and operations with structured or unstructured XML data.

To cover all modeling situations, the authors of UML [10], but also the OMG organization [12] give the possibility to extend syntax and semantics of UML

language through *stereotypes*, *comments* and *restrictions*.

Defining a collection of stereotypes, comments and restrictions, which extend an existing diagram type, in order to achieve a certain goal, is called a *profile*.

Together, the three mechanisms allow the creation of UML extensions adapted to a specific project. These mechanisms also allow adaptation of UML language to new information technologies, such as advanced object-oriented programming languages and their characteristics, distributed technologies, multidimensional analysis.

It is possible to add new items, to change the existing specifications and even to modify their semantics. Of course, it is important that these extensions to be done in a controlled manner, so that the UML objective remains unchanged, namely modeling information.

The study [3] presents a design methodology for object-relational databases. It defines new UML stereotypes, specific for object-relational databases and proposes a set of rules to transform UML schema in an object-relational one.

The papers [7] and [8] propose extending the set of standard UML stereotypes and restrictions by introducing a corresponding set for multidimensional modeling. For applications frequently used, like Web applications, [9] proposes an UML extension, and the studies [10] and [11] examine some extensions used to model relational databases.

Taking into account previous extensions for database design, [3] explains standard stereotypes of a relational database for each level: conceptual, logical and physical (Table 1).

Table 1 - Stereotypes for relational databases design

(Source: adapted from [3])

Database level	Database element	UML element	Stereotype
Conceptual level	Database	Component	<<Database>>
	Schema	Package	<<Schema>>
	Persistent class	Class	<<Persistent>>
	Multivalued attribute	Attribute	<<MA>>
	Calculated attribute	Attribute	<<DA>>
	Composite attribute	Attribute	<<CA>>
	Identifier	Attribute	<<ID>>
Logical level	Table	Class	<<Table>>
	Virtual table (view)	Class	<<View>>
	Column	Attribute	<<Column>>
	Primary key	Attribute	<<PK>>
	Foreign key	Attribute	<<FK>>
	Entity constraint (NOT NULL)	Attribute	<<NOT NULL>>
	Uniqueness constraint	Attribute	<<Unique>>
	Domain constraint (CHECK)	Constraint	<<Check>>
	Trigger	Constraint	<<Trigger>>
	Stored procedure	Class	<<Stored Procedure>>
Physical level	Tablespace	Component	<<Tablespace>>
	Index	Class	<<Index>>

As stated in [4], an UML extension must contain a short description, the list and description of stereotypes, the comments, the constraints, and also a set of formatting rules which are needed in order to specify if the data model is consistent.



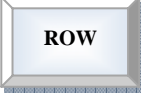

For each stereotype are defined properties, semantics, comments and constraints. Also, each stereotype is identified by specific icons. *To define these stereotypes we propose the template* detailed in Table 2.


In order to use UML as modeling standard for object-relational databases, it should be extended to be applied to the object-


relational data model. It would be possible to represent special types such as collections (ARRAY), nested tables, references (REF).


We realized the UML extension using the CASE product IBM Rational Software Architect that offers the possibility to define custom profiles, which allows extending menus, stereotypes, comments and other restrictions. We applied the proposed extension to highlight the features of the object-relational model in the UML class diagram.


Table 2 - UML extension for object-relational database design
(Source: Authors)

<<UDT>>
- Name: User defined type
- Base class: Class
- Description: The extension defines an object class (an user defined object type)
- Image: 
- Constraints:
- It is used to define the data type of other elements
- Can encapsulate attributes and methods
- Values:
Unspecified
<<REF>>
- Name: Tip REF
- Base class: Attribute
- Description: The extension defines a relationship to another object type
- Image: 
- Constraints:
It is used to refer an object type
- Values:
The pointed object type
<<ROW>>
- Name: Tip ROW
- Base class: Attribute
- Description: The extension defines a composite attribute
- Image: 
- Constraints:
- It is used to refer a composite attribute type, with a given number of elements
- The elements of ROW type can have different data types
- The ROW type has no methods
- Values:
Name of each component and the corresponding data type
<<ARRAY>>
- Name: Tip ARRAY
- Base class: Attribute
- Description: The extension defines a collection data type
- Image: 
- Constraints:
- It is used in order to refer a collection data type, with a finite number of indexed elements
- The elements of ARRAY type can have any data type, except the ARRAY type
- Values:
Number of elements of the collection and the corresponding data types

<<OID>>	
-	Name: OID
-	Base class: Attribute
-	Description: The extension defines an unique identifier for each object type
-	Image: 
-	Constraints: Unspecified
-	Values: Unspecified


<<LOBAttribute>>	
-	Name: LOBAttribute
-	Base class: Attribute
-	Description: The extension defines a multimedia attribute (image, video, audio)
-	Image: 
-	Constraints: Unspecified
-	Values: Unspecified
	Derivation rule Type: LOB Multiplicity: 1


<<SpatialAttribute>>	
-	Name: SpatialAttribute
-	Base class: Attribute
-	Description: The extension defines a spatial (geographic) attribute of the object type
-	Image: 
-	Constraints: Unspecified
-	Values: Unspecified
	Derivation rule Type: Spatial Multiplicity: 1

<<UDTAttribute>>	
-	Name: UDTAttribute
-	Base class: Attribute
-	Description: The extension defines an attribute with the user defined type
-	Image: 
-	Constraints: Unspecified
-	Values: Unspecified
	Derivation rule Type: UDT Multiplicity: 1

<<Inheritance>>
- Name: Inheritance
- Base class: Association-Generalization
- Description: The extension defines a generalization relationship between object types
- Image: Unspecified
- Constraints: Unspecified
- Values: Unspecified

<<Validation rule>>
- Name: Validation rule
- Description: The extension defines the rules of membership of classes to object-relational data model
- Image: Unspecified
- Constraints: Unspecified
- Values: Unspecified

<<Supertype>>
- Name: Supertype
- Base class: Class
- Description: The extension defines a supertype from an inheritance hierarchy
- Image: 
- Constraints: Unspecified
- Values: Unspecified

<<Subtype>>
- Name: Subtype
- Base class: Class
- Description: The extension defines a subtype from an inheritance hierarchy
- Image: 
- Constraints: Unspecified
- Values: Unspecified

The above table shows a general pattern which can be used to indicate the specifications concerning the characteristics of object-relational model in the UML diagram of classes. Using the proposed extension, the classes diagram

will contain references and visual images of elements for the presented data model, such as object types, object identifiers, special attributes, and inheritance relationship.

4. Solution concerning the development flow of the applications with object-relational databases

Analyzing the *top-down* and *bottom-up* strategies for implementing the informatics systems, we conclude that, if using object-relational databases, it is appropriate to choose any of them based on the existing system characteristics.

A top-down approach is appropriate and desirable in a situation where there is not implemented an informatics system or if there are disparate applications which work as a result of local developments made over time, but their preservation is not vital.

If, however, what is already required to be kept in its current form and with current solutions, the approach should be bottom-up. This will try to integrate the existing systems into a new one which will be available throughout the organization.

After the identification and analysis of system requirements and setting its objectives we will decide whether to store some components of the existing system or replace them completely, by choosing an appropriate approach for development.

We consider that it is appropriate to achieve an unified and homogeneous system using top-down development strategy, considering that integrating existing databases it is very possible that they are not built using an object-relational data model and they use different DBMS.

Based on standard steps to be taken into consideration in order to develop database applications, we believe that the sequence of activities must be that shown in Figure 2.

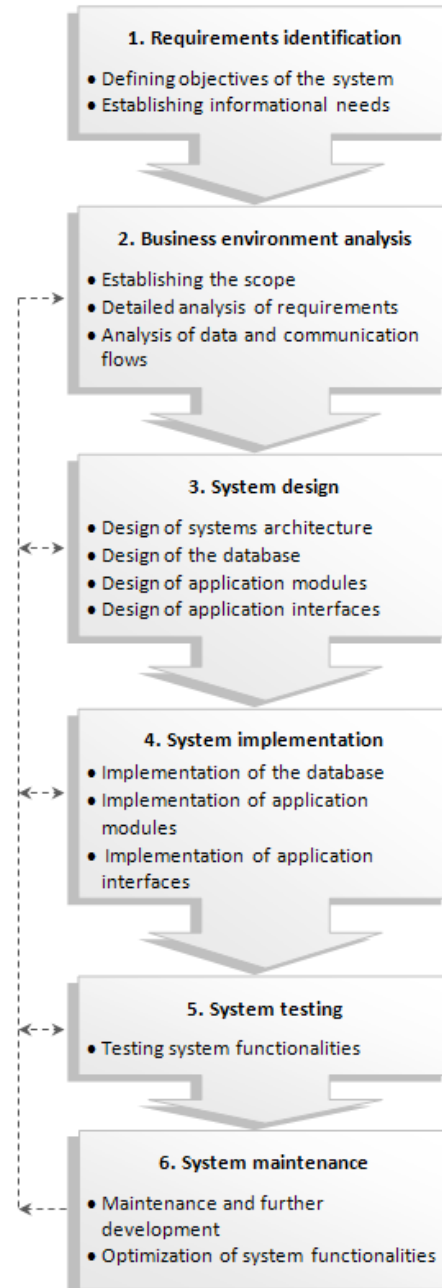


Figure 2 – Development flow of the applications with object-relational databases
(Source: Authors)

1) The first activity to be addressed is to *identify system requirements* that define the system objectives and future information needs for all its users. Must determine how to achieve the objectives and steps that are necessary to go, so there is a rigorous approach and the obtained system to work properly and have flexibility. Also, we need not to forget the study of law in the

fields of activity and the work rules and procedures established by the organization.

2) *Business environment analysis* involves the identification of the environment in which the business operates, the detailed analysis of operational requirements, the analysis of operating rules and procedures, data analysis and communication flow analysis within the organization and outside it.

3) The *system design* involves actually building the new system architecture, database design and corresponding applications design.

Database design is a very important step because it is responsible for ensuring consistency of data stored. A poor database design can lead to compromise data integrity, their redundancy and thus reduced performance.

Paper [13] identifies and describes three levels for designing object-relational databases, as we propose in Figure 2:

- *Conceptual level*, refers to the development of a model independently of any consideration regarding the appearance of data, as result of database analysis and modeling;
- *Logical level*, refers to the development of the object-relational data model, but independent of the chosen type of DBMS and other physical aspects of the model;
- *Physical level*, refers to the effective implementation of object-relational database, including these aspects related to ensure data security, managed through a specific DBMS.

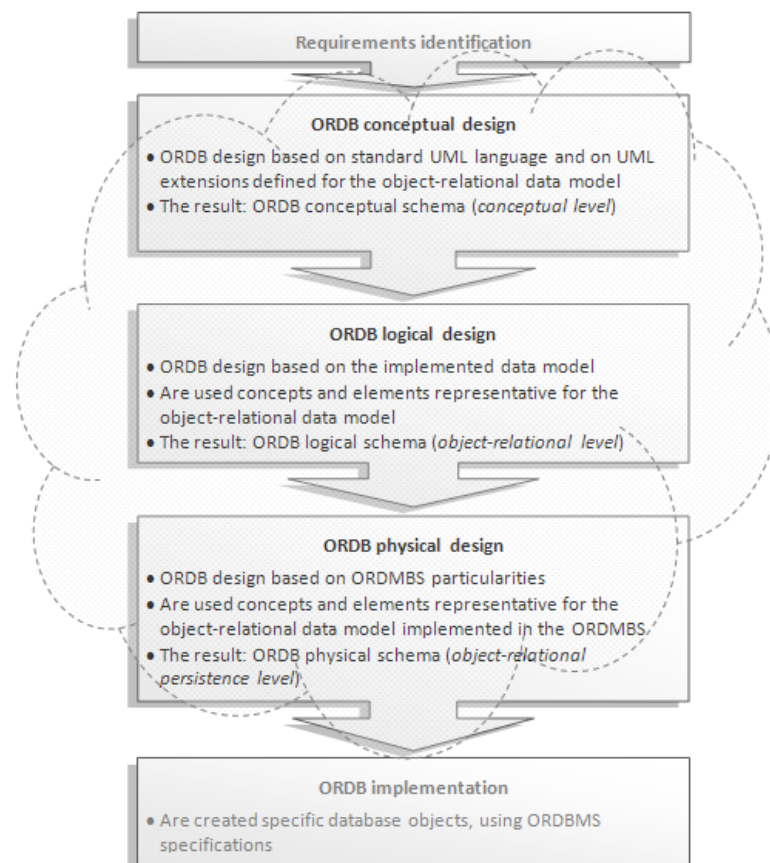


Figure 3 – Object-relational databases design
(Source: Authors)

Figure 3 shows the steps that are required to be made in order to design an object-relational database. In this respect, it is

necessary a process of mapping the model obtained from UML class diagrams to a standard object-relational model,

independent of product, and then the subsequent implementation in a DBMS. Mapping of the object model specific to UML language in an object-relational model that can be implemented in a database requires, according to [5], the implementation of identity, fields, classes, associations and relationships of generalization, aggregation and composition.

The identity problem is treated in [5], where two approaches are specified on the implementation of identity: based on the existence and based on value. The same paper also states aspects of data fields' implementation.

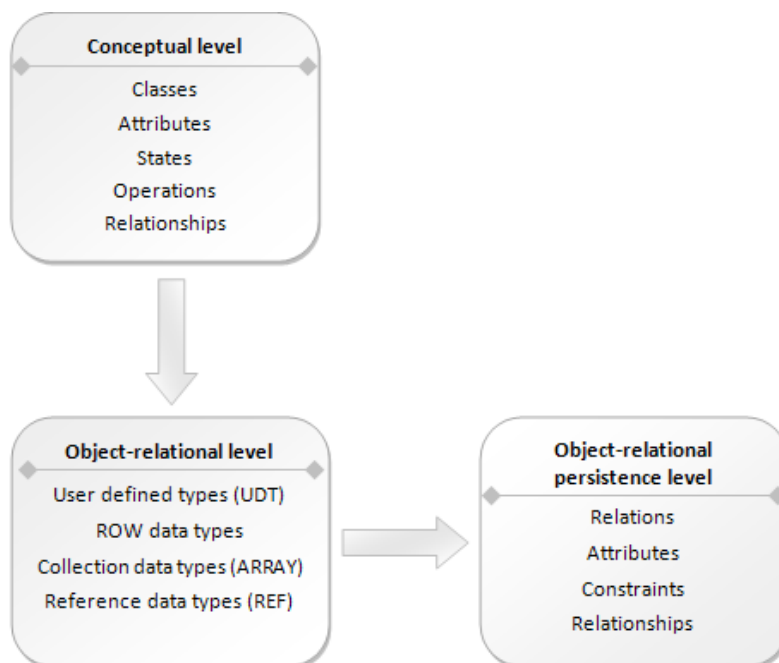


Figure 4 – Levels involved in object-relational mapping
(Source: adapted from [6])

Conceptual level of UML classes (conceptual schema of the database) meets the requirements of the system and focuses on classes, associations, attributes, states, operations. Can be identified different types of associations between classes, such as aggregation, composition, recursive association, hierarchy, class association.

Object-relational level (logical schema of the database) contains the elements proposed by the standard SQL:2003, such as: user-defined types, structured types, references, ROW data types and data

collections. Definitions made at this level do not allow persistent objects or data, until are created the tables that will store them.

According to [1], the transformation of an UML class diagram at object-relational is resumed to the mapping of logical structure of data and their behavior. This is not possible with traditional entity-association model (E-A). Unlike the relational model, the object types support encapsulation and the definition of methods can be explicitly associated with defining the types of objects.

Corresponding to the three stages of designing object-relational databases, [6] identifies three levels of mapping, outlined in Figure 3:

Object-relational persistence level (physical schema of the database) is composed of tables, some of which are created from the previously defined object types. In addition to tables, this level also contains other specific relational elements, such as restrictions, data fields, relationships between tables.

In the practical experiment that we have realized, we used mapping to make the transition from the conceptual level, shown in the UML diagram of classes, at the object-relational one. Implementation is made in Oracle DBMS, by defining object types and data type constructors, corresponding to the object-relational data model. Later, we made the transition to the level of persistence, by creating tables of objects on the types defined in the previous level.

The paper [2] proposes to use UML class diagrams to design the conceptual schema instead of entity-association model (E-A), commonly used for relational databases.

Unlike E-A model, UML has the advantage that it allows the entire system design and makes easier the integration of different views on the system. Design phase is divided into two steps:

- Logical design, standard, independent of any product;
- Specific design, which takes into account a specific product (Oracle, Informix etc.).

The logical design is particularly important in designing object-relational databases, because each product implements a distinct object-relational model. Thus, we can identify an object-relational specification, independent of product, which can be easily determined using the standard SQL:2003.

In terms of graphical representation, based on the stereotypes proposed to extend UML, we can use a number of stereotypes specific for the object-relational model and also for the standard SQL:2003.

The specific design involves specifying the SQL schema in the chosen product. In terms of graphical representation, based on stereotypes proposed to extend UML, we can use a number of stereotypes specific for the object-relational model implemented in the selected product.

Next, from [3], we have indicated some rules to be followed for successful implementation of mapping between specified schemas (Table 3).

Table 3 – Rules used for mapping within schemas
(Source: adapted from [3])

UML	SQL:2003	Oracle 10g
Class	Structured type	Object type
Class extension	Typed table	Object table
Attribute	Attribute	Attribute
Multivalued attribute	ARRAY	VARRAY
Composite attribute	ROW / Attribute with structured type	Attribute with object type
Calculated attribute	Method / Trigger	Method / Trigger
Association		
1:1 association	REF-REF	REF-REF
1:n association	REF-ARRAY	REF-Nested table
n:n association	ARRAY-ARRAY	Nested table - Nested table
Aggregation	ARRAY	Nested table
Generalization	Data type	Child object type created with UNDER clause

In the practical application, we experienced the changes for the three levels of mapping presented. So we transformed the UML classes in Oracle object types and then their extensions in object tables. Associations between UML classes were identified as being of type one-to-many (1:n), which led to the inclusion of a REF type attribute in the type of object that participates in association with multiplicity n . Since some UML classes presented generalization relationships, we have achieved in Oracle the implementation of the concept of inheritance by defining subtypes (each object belongs to a sub-class, is also part of the super-class). We have made the inheritance definition by specifying the UNDER clause in the specifications of each subtype, indicating in this way the supertype.

The result of the design phase is represented by the object-relational database schema, used in the phase pre-implementation.

4) After the design, follows the *system implementation* phase, by which are specified the main components (using the components diagram) and their distribution on existing physical resources (modeled through the deployment diagram).

In this phase, is created the physical schema of the object-relational database, are written programs to define and manipulate objects in an ORDBMS that supports high-level programming languages, standard or proprietary, and also standardized declarative language SQL. Also, is tested the system and are made improvements in its operation.

Practical experimentation uses the design concepts defined above, making the changes necessary to transition from UML modeling to object-relational data model, and then to persistent objects in the database. We have implemented the concepts using the Oracle DBMS and the object extension included in the procedural language PL/SQL.

5) *Testing the system* enables its adjustment so we can decide its proper functioning regardless of the parameters involved. Thus, implementation and testing the system functionalities are phases that complete the system development.

6) However, during system operation will be made *maintenance and further development*, based on additional requests of the beneficiaries or based on changes in the organization activities. In case of failure, we can return to a previous activity and produce a new version of the system.

5. Conclusions

The paper deals with implementation issues concerning the object-relational databases, as part of informatics systems development.

Based on the literature and using the features of the standard modeling language UML, we have proposed an extension of it for modeling an object-relational database. The proposed template explains the standard stereotypes that are specific for the object-relational data model. For each stereotype are indicated properties, semantics, restrictions and comments.

Also, we have proposed a solution for the development flow of the applications with object-relational databases. Based on standard steps to be taken into account in the process of development of a database application, we consider that the sequence of activities must present specific features for each level, as outlined in the paper.

We considered the design as the most important activity by the fact that it is the only one that incorporates features of the object-relational data model. Thus, design is the activity that we detail the most in the paper, specifying a series of rules to be followed for a successful transition from standard design to the specific one.

Acknowledgment

This paper presents some results of the research project PN II, TE Program, Code 332: "Informatics Solutions for decision making support in the uncertain and

unpredictable environments in order to integrate them within a Grid network”, financed within the framework of People research program.

References

- [1] M.Wang, *Using UML for object-relational database systems development: A framework*, Issues in Information Systems, vol.9, no.2, 2008, ISSN 1529-7314
- [2] E.Marcos, B.Vela, J.M.Cavero, P.Cáceres, R.Juan, *Aggregation and Composition in Object – Relational Database Design*, 2001
- [3] E.Marcos, B.Vela, J.M.Cavero, *A Methodological Approach for Object-Relational Database Design using UML*, Software and Systems Modeling Journal, vol.2, no.1, 2003, pp.59–72, ISSN 1619-1374
- [4] A.Bâra, V.Diaconița, I.Lungu, M.Velicanu, *Improving Performance in Integrated DSS with Object Oriented Modeling*, WSEAS Transactions on Computers, no.4, vol.8, 2009, pp.599-609, ISSN: 1109-2750
- [5] C.Strîmbei, *Utilitatea metodologiei UML în proiectarea bazelor de date*, Informatica Economică Journal, no.2(26)/2003, pp.56-62, ISSN 1453-1305
- [6] M.F.Golobisky, A.Vecchietti, *Fundamentals for the Automation of Object-Relational Database Design*, IJCSI International Journal of Computer Science Issues, vol.8, no.2, 2011, ISSN 1694-0814
- [7] S.Luján-Mora, J.Trujillo, I.Y.Song, *Extending UML for Multidimensional Modeling*, vol. Proceedings of the 5th International Conference on The Unified Modeling Language, 2002, pp.290-304, ISBN 3-540-44254-5
- [8] S.Luján-Mora, J.Trujillo, I.Y.Song, *A UML profile for multidimensional modeling in data warehouses*, Journal Data & Knowledge Engineering - Special issue, vol.59, no.3, 2006, pp.725-769, ISSN 0169-023X
- [9] J.Conallen, *Building Web Application with UML*, Addison-Wesley, 2000, ISBN 0-201-61577-0
- [10] G.Booch, J.Rumbaugh, I.Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley, 1998, ISBN 0-201-57168-4
- [11] E.J.Naiburg, R.A.Maksimchuk, *UML for Database Design*, Addison-Wesley, 2001, ISBN 0-201-72163-5
- [12] OMG, *Unified Modeling Language: Infrastructure and Superstructure*, August 2007, <http://www.uml.org/>
- [13] T.Connolly, C.Begg, *Database Systems. A Practical Approach to Design, Implementation and Management*, 4th edition, Addison Wesley, 2004, ISBN 0-321-210-255



Manole VELICANU is a Professor at the Economic Informatics Department at the Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies of Bucharest. He has graduated the Faculty of Economic Cybernetics in 1976, holds a PhD diploma in Economics from 1994 and starting with 2002 he is a PhD coordinator in the field of Economic Informatics. He is the author of 18 books in the domain of economic informatics, 64 published articles (among which 2 articles ISI indexed), 55 scientific papers published in conferences proceedings (among which 5 papers ISI indexed and 7 included in international databases) and 36 scientific papers presented at conferences, but unpublished. He participated (as director or as team member) in more than 40 research projects that have been financed from national research programs. He is a member of INFOREC professional association, a CNCSIS expert evaluator and a MCT expert evaluator for the program *Cercetare de Excelenta - CEEEX* (from 2006). From 2005 he is co-manager of the master program *Databases for Business*

Support. His fields of interest include: Databases, Design of Economic Information Systems, Database Management Systems, Artificial Intelligence, Programming languages.



Iuliana BOTHA is an Assistant Lecturer 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 Cybernetics, Statistics and Economic Informatics in 2006 and the Databases for Business Support master program organized by the Academy of Economic Studies of Bucharest in 2008. Currently, she is a PhD student in the field of Economic Informatics at the Academy of

Economic Studies. She is co-author of 6 books, 13 published articles (3 articles ISI indexed and the other 10 included in international databases), 16 scientific papers published in conferences proceedings (among which 6 paper ISI indexed). She participated as team member in 4 research projects that have been financed from national research programs. From 2007 she is the scientific secretary of the master program *Databases for Business Support* and she is also a member of INFOREC professional association. Her scientific fields of interest include: Databases, Database Management Systems, Design of Economic Information Systems, Business Intelligence, e-Learning Technologies.