Technologies for Development of the Information Systems: from ERP to e-Government

D. Litan, A. Apostu, L. Copcea (Teohari) and M. Teohari

***Abstract*—**Although some ERP (Enterprise Resource Planning) software products are available on the market, they do not entirely fit each and every type of business. That is the main reason why companies prefer to develop their own ERP type of systems, thus eliminating "barriers" such as long time and high costs required to develop the software, compared with the shorter time and lower costs of just purchasing the ERP system and customizing it to the targeted business. However, in the short run, lower cost will eventually result into losses in the long run, due to that, one the one hand, the system is not perfectly adapted to business specifics and, on the other hand, customizing an ERP package can influence implementation of the standard benefits of an integrated system. Therefore, many times, the best choice for a company would be to develop an ERP type of system that caters perfectly for the needs of the organization. In this paper, we will proceed to the analysis, presentation and comparison of the current most important information methodologies (for business analysis and for IT projects – people-oriented methodologies) and databases/ data warehouses which are being used in development of the ERP systems. Also, another aspect that we will consider, in our paper is to examine how some of the technologies used in the development of ERP systems can be applied in the development of e-government type systems.

***Keywords*—**data warehouse, e-government, Enterprise Resource Planning, information methodology, query optimization, information system, information technology.

I. Introduction

THE acronym ERP (Enterprise Resource Planning) was first used in 1990 by firm Gartner Group in connection to MRP (Material Requirements Planning, later Manufacturing Resource Planning) and CIM (Computer-Integrated Manufacturing) and since then it has become widely used for streamlining *not only manufacturing process but all other business processes*. ERP packages integrate solutions for different directions such as accounting, contracts, payroll, maintenance and human resources management, attempting to provide technology solutions for all core functions of an enterprise, regardless of its specific business flow, such as non-manufacturing businesses, non-profit organizations and governments.

There are two main areas in which successfully implemented ERP systems prove their results: the first refers to *operational efficiency* and increased *productivity through automated transactions* offering improved support for decision-making. The second aspect is regarding *improved customer service* taking into consideration the market performance. Both types of ERP benefits are generally expected to result in an improved competitive position, low production cost, expectations of revenue growth, ability to compete globally and the desire to re-engineer the business to respond to market challenges.

But, on the one hand the rigidity and the difficulty of adapting the ERP systems to the specific business requirements of the companies, and on the other hand the complexity of the packages, the hardware and software necessary resources for implementing ERP systems can be very expensive, especially if customization is needed, are two of the main causes for failure in ERP systems implementation.

A. ERP Customization versus Configuration

"An effective IT infrastructure can support a business vision and strategy; a poor, decentralized one can break a company." The process of ERP customization requires strong working knowledge about the existing system and also regarding the innovations in ERP. As customizing is an integral part of ERP solutions one may say that the success of an ERP implementation depends directly on the rate of customization. This process poses a serious challenge on the time and the funds allocated, a successful management depending on balancing them and making both sides meet.

A recently published report (the 2010 Gartner FEI Technology Study: The CFO's Perspective on ERP [1]) presents statistics about ERP implementation, organizational needs and system configuration. ERP solution modification using configuration versus customization is an important topic as"customization has become the dreaded <<c-word>> in the enterprise software community, primarily due to the fact that extensive customization can render systems difficult and/or impossible to upgrade"[2].

Gartner [1] makes the following distinction: "All companies require a certain level of configuration, such as using setup tables to define the code block structure. Configuration within the product is typically not considered customization because it can be upgraded. Customization, however, adds functionality with additional coding."

A similar approach is presented in Focus Research's "A

License to Drive ERP Adoption” [3], article: “configuring a system without breaking its underlying code or compromising its operating panel” is a safer path than customization.

The latest ERP software evolution offers the possibility to customize in order to more closely adhere to the business processes of organizations, this aspect being less expensive than building a completely customized application. A disadvantageous aspect is when upgrading applications or customized functionalities implies errors in processing or system breakdowns that in most cases require completely rewriting the custom code and drive increased development costs.

One of the solutions to the mentioned situation comes from vendors who offer specific templates and “products optimized for very specific micro-verticals” [2] that can be successfully implemented for the major part of midsize organizations.

As the tendency is now towards less customization, ERP vendors provide customization options that imply less modifications on code level. Source [1] provides three steps to ensure companies get the most advantage from their ERP deployments:”letting employees become familiar with the software prior to any modifications; limiting customization to certain areas; and adhering to vendor-established guidelines when doing so”.

The process of customization should necessarily be based on the same set of methodologies implemented for the applications and should be supported by accurate and detailed documentation. This approach should ensure a smooth adaptation to organizations unique business processes, industry requirements and corporate needs.

An aspect that cannot be neglected is that customization increases implementation costs and requires important resources for ERP packages upgrades. In this case, either configuring a system without modifying its underlying code or reconsidering the business practices in order to adapt to the existing functionalities are suitable for a successful and less expensive upgrade of the application. For example, some companies may find it useful to upgrade their “age-old supply-chain-management processes” to suit a brand new ERP system.

Paper [5] proposes a solution architecture for an ERP Integrated system for store management and back office applications passing through all stock movements available in the standard ERP functions and in customizing(with a lot of management parameters to be customized store by store) that can be adapted to other different business situations. Two important aspects are the size of the implementation given by the large number of stores considered in the network that implies computation challenges for the application developed within the “ERP Central Component (ECC)” and for “the simulation model used for comparing forecasting algorithms” and the potentially overload that can be generated in the system control due to the large number of items that need to be taken care of.

The chart below (source: the Gartner report [1]) synthesizes the connection between the level of configuration and code modifications that explains the reason why enterprise software buyers choose not to configure their solutions beyond more than minimal level.

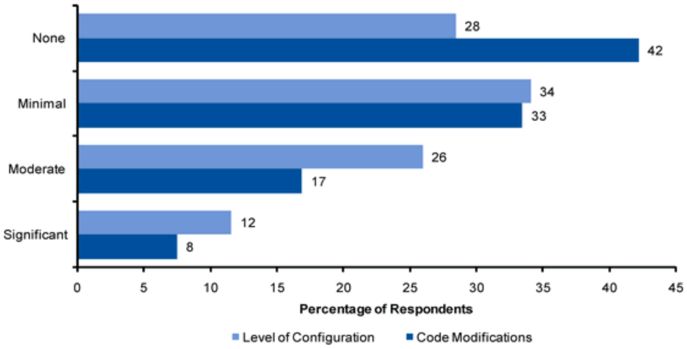


Fig.1 - Chart: Configurations Levels and Code Modifications; source: [1]

Gartner confirms “a fair amount of product configuration is usually necessary to satisfy buyers with complex requirements. Buyers who search for packages that can do everything out of the box aren’t likely to find the list of compromises they confront will be long and expensive.”

B. The ERP Systems Development

As we said above, customizing an ERP package can prove to be expensive and resource consuming and can influence the implementation of the standard benefits of an integrated system, therefore a number of companies prefer to develop their own ERP systems.

The organization which plans to develop and “implement ERP project, usually employs multiple experts from different sections in selection process”[6] because the development and implementation of an ERP system requires first a *business analysis* for establishing the flow of activities and specific documents, the necessities of the customer and the needed customization for a normal development of the business flow; second, *the methods and models* for developing, implementing and maintaining the ERP system are discussed in order to integrate the standard and the customized functionalities and to provide a unitary solution.

Next we will proceed to the analysis, presentation and comparison of the current most important *information methodologies* (for business analysis and for IT projects – people-oriented methodologies) and *databases/ data warehouses* which are being used in development of the ERP systems. Also, another aspect that we will consider, in our paper is *to examine how some of the technologies used in the development of ERP systems can be applied in the development of e-government type systems*.

II. Methodologies for Development of the Information Systems – “Organization – Oriented” Methodologies

“The increased exigencies on the quality of information systems, their importance in improving the performances of an organization have required that well structured software development and design strategies be defined and used, which should have priority to any activity related to information programs developing."[7] Thus "information modeling, as a part of requirement management process, is mainly dealing with capturing user needs and understanding system complexity."[8] But, despite this, the currently used analysis and design methodologies "can not handle the uncertain situations, communications with users or superior management development"[9], that is why in practice, the methodologies are partially used, as a result of their deficiencies.

The methodologies are used to indicate the development manner of the design and analysis process of the information systems, "establishing:

- components of the information system execution process (stages, sub-stages, activities, operations) and their contents;
- running (execution) flow of the components; methods, techniques, procedures, instruments, utilized norms and standards" [10].

Moreover, the execution methodologies of the information systems comprise:

- "approaching method of the systems, in order to clarify the relationship between the variations of the system and its dynamism;
- data formalization and editing processes;
- instruments for documentation conception, execution and drafting;
- project performance method and the specific actions of each stage (life cycle);
- defining of the working method, analysts and designers' role and the relationship between them;
- project management methods (planning, programming, follow-up)."[10]

In time, the information methodologies have been classified according to various criteria as follows:

a) *according their degree of generality*, the information methodologies fall into the following categories:

- *general methodologies*: "developed to allow the accomplishment of information systems from different years and of variable complexities" [11]. Examples of such methodologies: SSADM (Structured System Analysis and Design Methodology), MERISE (Méthode d'Etude et de Realization, Informatique pour les Systém d'Entreprise), OMT (Object Modeling Technique), RUP (Rational Unified Process);
- *framework methodologies*: "comprising elements which are applicable only to some software products: Selection and Implementation of Integrated Packaged Software (SIIPS). It has implementation accelerators for ORACLE and SAP"[12];
- *specialized methodologies*: "developed and optimized for the implementation of a single software product: AIM (for Oracle E-Business Suite), PQIS (for SunSystems), Extract (for Exact), Signature (for Scala), ASAP (for SAP)"[11].

b) *according to the approaching methods*, the information methodologies fall into the following categories:

- *methodologies with structural approaches* "whose working is based on dividing the system into subsystems according to the system functions (the functional approach) or according to the data (data based approach)"[12]. Examples of such methodologies: Structured Analysis and Design Information System (STRADIS), Information Engineering (IE), Structured System Analysis and Design Methodology (SSADM), Méthode d`Etude et de Realization, Informatique pour les Systém d`Entreprise (MERISE), Information System Work and Analysis of Changes (ISAC), Soft System Methodology (SSM), Rapid Application Development (RAD) etc.;
- *methodologies with object-oriented approach:* which enable the setting up of information system using oriented technologies. Examples of such methodologies: Object Oriented Design (OOD), Object Oriented Analysis (OOA), Object Oriented Structured Design (OOSD), Object Modeling Technique (OMT) etc..

"Taking into account all these methodologies, a standard as to symbols, notation, types of diagrams, models etc. called UML was drawn up."[14] The use of UML is performed with the help of RUP, as a general process for the object-oriented development of information products. „The UML gives us the ability to model, in a single language, the business, application, database, and architecture of the system. By having one single language, everybody involved can communicate their thoughts, ideas, and requirements"[15]

c) *the information methodologies, according to the information system life cycle*, fall into the following categories:

- *cascade model methodologies;*
- *spiral model methodologies;*
- *incremental model methodologies;*
- *evolving model methodologies;*
- *composite model methodologies (cycles V and X).*

These methodologies follow the information system life cycle.

d) *the information methodologies, according to the structure of the information methodologies processes*, fall into the following categories: "mono-process methodologies (IBM/ICI), multi-process methodologies with weakly connected processes and multi-process methodologies with interconnected processes (AIM methodology)"[12].

Although it seems unbelievable, in the "software world" only a low percentage of the projects on information systems development are successfully completed. „After two decades of this problem reoccurring, one of the leading causes for the high failure rate is still poor process modeling (requirements' specification). Therefore both researchers and practitioners recognize the importance of business process modeling in understanding and designing accurate software systems."[16]

A. Comparative Approach of Some Information Methodologies

Development of big systems, such as ERP type systems or e-government type portals require the usage of information methodology (for analysis and development) early at the beginning of the development cycle of the new system. Though, “conceptual modeling is central to information systems development”[17], in practice, this stage is most of the times skipped in development of small projects. This is the main reason why currently a great deal of the projects end up in a failure or with clients’ requirement for modifications of the final products.

Because “many methodologies have been developed to improve business processes and to investigate how well current business processes have achieved their goals”[18], before taking a decision regarding the best methodology to use, we hereby propose an analysis and comparison of the main information methodologies.

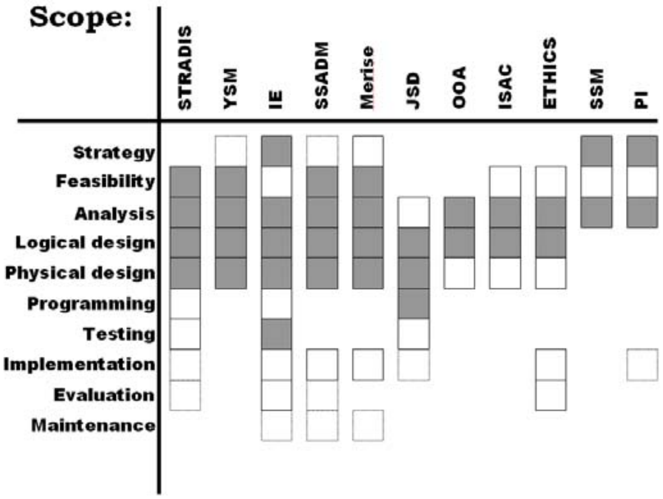


Fig. 2 - Purpose of the information methodologies; source: [19]

	SSM	IE	STRADIS	YSM	SSADM	MERISE	RUP
Objectives	aims at much more than developing an IT system	have clear objectives to develop computerized information systems					
Domain	address general planning, organization, and strategy of information and systems in the organization		are classified as specific problem-solving methodologies				
Target	more applicable in human activity ‘messy’ situations	designed for large systems	general-purpose, DFDs not suitable for management information systems or web-based systems	designed for large systems			general-purpose, not very useful for small systems
Model	DFDs (play a less significant role than in STRADIS)	integrate both processes and data	uses primarily DFDs	DFDs (play a less significant role than in STRADIS)	integrate both processes and data		
Techniques	does not heavily use techniques and tools	explicitly suggests that the techniques are not a fundamental part of the methodology	is largely described in terms of its techniques	specify techniques and view them as important for the methodology		ranges from simple drawing tools to those which support the whole development process, including prototyping, project management and code generation	specify techniques and view them as important for the methodology
Product	comes only with some academic papers	some online specs	some online specs	some online specs	comes with a large set of manuals	has a range of books, and online specs	has a range of books, and online specs
Practice	• academic origin • both business and technical people	• commercial origin • professional technical developers				• academic origin • professional technical developers	

Table I - Comparative Development Methodologies
DFDs = Data Flow Diagrams

As we can see in Table I and Fig. 2, we can easily notice that each methodology has both advantages and disadvantages:

- *SSADM* (The Structured Systems Analysis and Design Method): "It is a detailed method, covering almost all elements of the information system"[9];
- *UML* (The Unified Modeling Language): "It is an expressive modeling language, covering all development elements of the information system"[9];
- *SSM* (Soft Systems Methodology): "Whereas most of the methodologies deal only with soft issues, SSM covers the hard system, as well. SSM supports the activities and processing, creating a conceptual representation pattern of the source activities"[9];
- *MERISE* (Methode d'Etude et Realisation Informatique par le Sous – Ensemble representatif): this methodology allows:

- "closeness to the information system and to the ideal structure of a database;

- a three level description of the information system;

- utilization of precise, simple and exact representation *formalism* for data description. (*Formalism*, as used above, means a group of definitions and rules, combined with a group of diagrams and/or tables). This formalism is internationally regulated by ISO standard under ENTITY-ASSOCIATION name;

- detailed description at conceptual level, allowing the creation of an information system independent of the organization of the firm and choice of the automation technique;

- the visual representation used in the conceptual pattern facilitates the dialogue among all partners involved in the creation of the information system."[20]

- *STRADIS* (Structured Analysis, Design and Implementation of Information Systems): this methodology is based upon the top-down functional decomposition method and usage of Data Flow Diagrams;
- *YSM* (Yourdon Systems Method): this methodology is similar to STRADIS methodology, "its approach being focused upon the data structures importance"[9].

Yet, if we are to draw a line a provide a "verdict" concerning the choice of one of the methodologies in order to develop an information system, this would prove impossible to do, without taking into consideration the type of the information system that is to be developed, the size of the future system, the time allotted for the entire project (see Table I – utilization of some methodologies require more time than others) or even the members of the teams or their competencies in charge with the future information system (see Fig. 2).

B. Life Cycle of an Information System – from ERP Type Information Systems to e-Government Systems

The big sized information systems, mandatorily requires the utilization of information methodologies for analysis and design stages. ERP applications (Enterprise Resource Planning) with CRM (Customer Relationship Management) and/or SCM (Supply Chain Management) integrated modules are considered by us big sized information systems. If we are to consider the future, as well, e-government portals can included into the same category of big sized information systems. *e-Romania* project wishes to be such a portal. The interest fields, part of e-Romania portal are: e-Health, e-Environment, e-Transport, e-SMEs, e-Agriculture, e-Justice, e-Education, e-Culture, e-Church, e-Tourism, e-Association, e-Sport, e-Citizen, e-Civil Servants, e-Statistics. The fields of an e-government portal can be "approached" in a similar way with ERP modules.

Irrespective of the used methodology, for the analysis and designing of the information system, this has a life cycle and a development one.

Normally, *the life cycle of an information system* spans between the date when the decision to develop the new system was taken and the date when that system was decommissioned. As expected, the life cycle of an information system also includes *the development cycle of the information system*. The development cycle of the information system spans between the date when the decision to develop the new system was taken and the date when that system was commissioned.

Next, we will present the life cycle – *cascade pattern* of an information system, exemplified on an *e-government information system type which enables the introduction of online applications by citizens/firms and submitted to the public administration, to which the system belongs,* highlighting the main differences that may appear, compared to a "classic" business system.

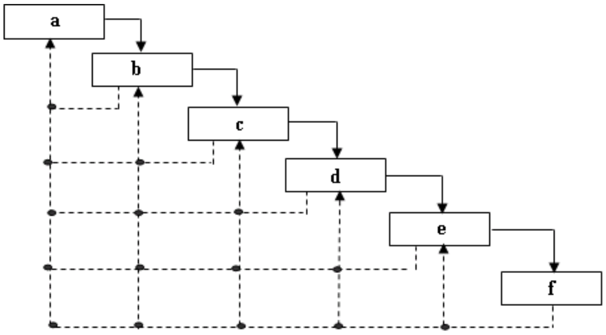


Fig. 3 – Life cycle of an information system: cascade pattern; source: [20]

Legend for Fig. 3:
a = Defining of the requirements;
b = Analysis;
c = Design;
d = Implementation;
e = Testing;
f = Utilization and maintenance.

As it can be seen in Fig. 3, the life cycle of an information system – cascade pattern, begins with stage that deals with *defining of the requirements*. Whereas in the information systems for business, stage dealing with defining of requirements starts from the existing system (in activity), with the purpose to define its problems and identify new opportunities, when dealing with e-government systems, the situation is "slightly" different, because an "older" e-government system does not sometimes exist and until its development and implementation, the information flow exchange between citizens and state or between companies and state has taken place/ is taking place by traditional methods: physical presence at wicket, telephone, fax etc. Therefore, the requirements of defining stage for e-government systems, besides the requirements the future system must meet - which have been clearly mentioned by the public institution representatives, most of the time comprise:

- discussions with the public administration employees, directly involved in the activity of the institution for which the e-government system is to be developed;
- direct tracing /surveillance for a brief time period of the public institution activity method, performed by analysts;
- consultation of documents (both those in physical format – paper, as well as the electronic ones, e.g. Excel type) related to the public institution activity, and these documents are to be later part of the automated information system.

The *analysis* stage of the system.

After establishing all the requirements of the future information system, the analysis part of this stage will be performed by the means of a standard set of diagrams and notices, which are part of an informational methodology. This stage comprises:

- functional analysis of the system: making of the information concerning system inputs and outputs, data storage and archiving methods, system efficiency and complexity etc;
- operational analysis of the system: "detailed characterization of input data (type and dimension), of the current data from files (each one's length and status, grouping type, treating-processing method), of data positioning in the output documents in order to prepare the necessary processes to obtain them in the desired manner, inspection of the type and characteristics of the utilized peripherals, assessment of necessary amount of memory".[7]

The *designing* stage of the system envisages the construction of the new system architecture, as well as the logical designing (creation of the logical solutions) and the physical ones (specification of the new system components) of the information system.

The activities performed for the designing of the system are:

- *Establishing the system architecture;*

In the case of e-government information system, the system architecture must allow both the execution of the requests of the citizens/firms, as well as the administration/solving of the requests filed by the public administration employees.

- *Designing of processes;*

When designing the processes (in the case of e-government information systems), first, the "illustration" of the information flow between citizens/firms and public administration shall be considered, as well as the statuses experienced by each request submitted to the public administration (example of statuses:
- "Wait"- the request was issued by the citizen/firm and it has not been so far opened by any public administration employee;
- "Open" – the request has been opened by a public administration employee, the request has been submitted to the employee by the system and it is pending for solving;
- "Closed" – the request has been closed by an public administration employee, after its solving).

- *Database/files' designing;*
- *Inputs' designing;*

The inputs' designing (in the case of e-government information systems) involves the information that will be introduced into the system by citizens/firms, for example, the online requests submitted by citizens/firms on the one hand and the information introduced by public administration employees, in order to solve the received requests, on the other hand.

- *Outputs' designing;*

The outputs' designing (in the case of e-government information systems) involves deciding upon the export method of the information out of the system: electronic files, printed paper etc.

- *Designing of the users interface;*

Designing of the users interface involves the execution of the video formats for each information system module: the module that is to be used by citizens/firms and the module for public institution employees.

- *Designing of programs.*

This activity deals with the description of the programs required for the envisaged processing within the system.

The *implementation* stage of the system comprise both the development of the information system using the programming technologies, as well as the installation of the completed system on the servers of the public administration for which it has been developed.

Testing. Because e-government information systems have a "very recent" history, both in Romania and worldwide, unlike the business oriented information systems (in the latter case, when the new information system was launched, the users were already familiar with the automation of the information flow), in this case the testing will primarily involve system promotion among population, and based on the feedback received from the population, it will be decided whether the new e-government system has been developed properly or not.

The last stage of am information system development cycle is *utilization and maintenance.* Normally, the newly developed and implemented business systems work simultaneously for a while with the old systems. But, in the case of e-government system, the “old” system does not usually exist, so it is desirable that for a period of time after the implementation of e-government system, this should work simultaneously with the traditional means which allow information exchange (physical presence at wicket, telephone, fax etc.). As for the maintenance of the e-government system, this activity shall be performed by IT Department specialized personnel of the public institution, which owns the system.

An e-government information system that meets the above-mentioned requirements was developed for research purposes and it can be accessed online at the following address: http://www.egovernment-system.eu. Within this system, the public administration was considered to be the town hall of a Romanian town, Europe.

III. Methodologies for Development of the Information Systems - “People-Oriented” or IT Project Methodologies

Carrying out a project in the field of IT involves, on the one hand, teamwork (teams made up by: project manager, analysts, programmers, testers and after the finalization of the software product this is left “in the hands” of the application and database administers) and, on the other hand, the one who takes part, mostly in an active way, in carrying out the project is the client himself.

As „current hyper-competitive business environment requires business organizations to quickly build as well as dismantle dynamic collaboration relationships among various participating work systems, both internally and externally, to respond to fast-changing market needs”[21], the development of a project in the field of IT (Information Technology) is allotted both a term for developing each stage and a budget, so the possible failure which is spotted only at the end of the project, could have serious repercussions on the whole team or on the company that dealt with the project.

In order to prevent this kind of situations, the “people-oriented” methodologies have been developed. Agile belongs to this category. „Agile software development is a group of software development methodologies based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams.”[22] In other words, “Agile ensures the delivery on time of the projects within the quality limits, optimizing the team result by increasing the productivity, the degree of planning and by optimizing the structures within the organization”.[23].

Other methodologies belonging to Agile category are „XP (eXtreme Programming), SCRUM**,** DSDM**,** Crystal**,** Feature Driven Development**,** Lean Software Development (to mention only a few). They all use the basic principle of the Agile philosophy, but they implement it differently. ”[23].

The main features of Agile methodologies are:

- Agile methodologies include, first of all, iterations (see Fig. 4) which last from one to four weeks and which have a minimum planning. Each iteration involves working on planning, analysis, design, development/programming, testing and validation/acceptance by the client of the part/module from the project that was developed. Thus, going through all the stages made up of small “portions” of the project, any failure risk of the project is minimized and any possible modification occurred after the initial requirements can be “integrated” rapidly.

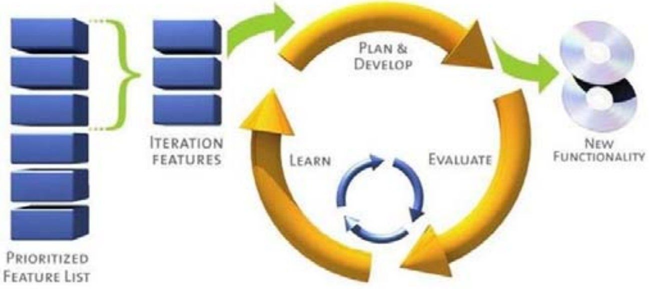


Fig. 4 – Representation of an Agile iteration; source: [24]

- the teams which use Agile are usually multifunctional and able to organize themselves, without the help of the hierarchical managers;
- Agile encourages the “face-to-face” communication among the members of the team. The members of the team usually work on the same premises, in the same office. If the members of the team do not work on the same premises they communicate with each other via videoconferences, e-mail, phone, etc. A team is normally made up of 5-9 persons. The members of the Agile teams usually have short daily extra-meetings besides the meetings held at the end of an iteration to inform the others on what they did the previous day and what they are currently doing or what obstacles they face;
- as we have mentioned above, more often than not, the client/his representative is also considered part of the team. When using Agile, this rule is observed;
- something which differentiates Agile from the “classic” working methodologies is the fact that, in the case of Agile, the written documentation is very scarce and only the absolutely necessary things will be written down;
- the specific tools used for improving the quality of the projects within Agile are: continuous integration, automated or xUnit test, pair programming, test driven development, design patterns, domain-driven design, code refactoring, etc..

The best known Agile methodologies are: Scrum and XP (Extreme Programming), whose features will be presented below.

A. *Agile Methodologies for IT Projects - SCRUM*

“Scrum is an agile framework for completing complex projects.”[25] “Scrum is an iterative and incremental process for software development in the cases when the requirements change rapidly”.[26] At the end of each iteration, the team will make a software product “with partial set of functionalities but which can be delivered to the client”[26].

The main characteristics/features of the Scum methodology are:

- the iterations last for 30 days and are called sprints, and each iteration is supposed to meet a number of pre-set requirements (see Fig. 5);

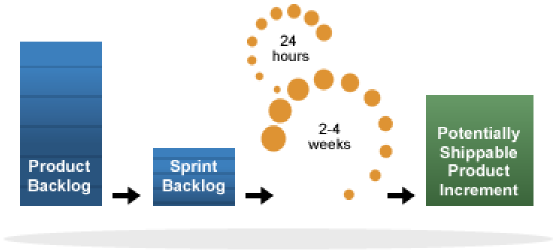


Fig. 5 - The Scrum framework; source:[25]

Note: Within the Scrum methodology, "Product Backlog" refers to: the software product requirements prioritized by the client, and "Sprint Backlog" refers to: the list of requirements which will be implemented in an iteration/sprint.

- establishing a daily meeting between the team members (6-8 members) where they discuss what has been accomplished, what is going to be accomplished and the possible obstacles that may appear while carrying out the project;
- at the beginning of every sprint a meeting to plan it is held where they also discuss the tasks which they failed to accomplish or which were not properly accomplished in the previous sprint in order to integrate them into the new sprint. At the same time, they schedule the activities that are going to be carried out in the next spring and the order in which they will be performed.;
- at the end of each sprint will take place a meeting with all the members of the team to discuss about the sprint which is over;
- the correspondent of the Project manager is the ScrumMaster whose "role is to eliminate the problems that prevent the team from attaining the sprint target" [26].

B. Agile Methodologies for IT Projects - XP (Extreme Programming)

„Extreme Programming (XP) is a software development methodology which is intended to improve software quality and responsiveness to changing customer requirements."[27]. In other words, Extreme Programming is „a discipline of software development that follows a specific structure that is designed to simplify and expedite the process of developing new software. XP teams design software for specific functionalities without adding any functionalities that are not specifically requested that may slow down the process, keeping the development course simple through systematic and regular testing and design improvements."[28]

This methodology is successfully used for the development of small and medium-sized software applications (the teams being made up of 3-10 programmers). „XP improves a software project in four essential ways; communication, simplicity, feedback, and courage:

- It stresses the role of *teamwork* with open and honest communication;
- It requires the skill to develop *simple programs* at top speed;
- It stresses the importance of concrete and rapid *feedback* (working with customers and doing continuous testing);
- It requires *courage* to throw code away and start again if necessary."[29]

Just like Scrum methodology, Agile is an iterative methodology. Each iteration lasts for maximum 4 weeks and is called story. The stages of an iteration are presented in Fig. 6.

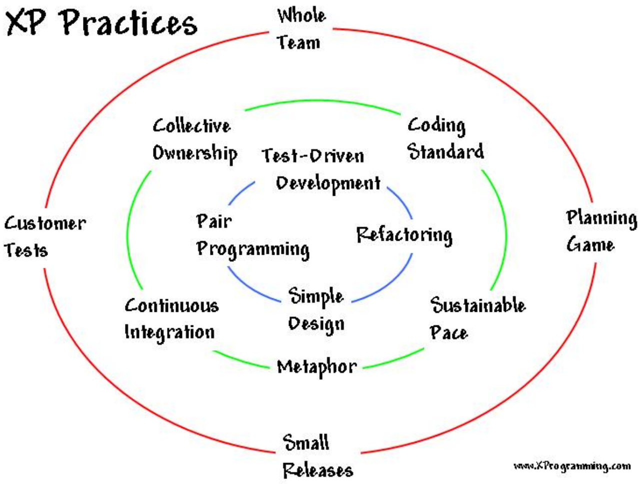


Fig. 6 – The practices and the main "cycles" of XP; source: [30]

As the project which is going to be developed is divided into iterations, the elements of an iteration (Fig. 6) are:

- *project planning (planning game):* each iteration will start with the „*planning game*" (or project planning). During this stage, the most important requirements of the project related to their cost are established.;
- *small releases*: „The software is developed in small stages that are updated frequently, typically every two weeks."[28];
- *whole team*: unlike the "classic" teams which develop software products, the XP team also includes the *client;*
- *metaphor:* „The metaphor describes the broad sweep of the project (describes the project size), while stories are used to describe individual features."[31], or, in other words, this part of the iteration „is a simple evocative description of how the program works"[30];
- *collective ownership:* this principle takes into account the fact that it must be established from the very beginning which programmer will be assigned each written code line;
- *coding standard:* this principle supposes that the whole team of programmers will write the code in the same way, so that, if a problem arises, this could be easily solved not necessarily by the programmer/programmers who wrote

those code lines but also by any programmer/ programmers from the team;

- *continuous integration:* as the code is being written, the programmers from the XP teams will update the rest of the team members with the new. This will be done several times daily.;
- *sustainable pace:* „The concept is that programmers or software developers should not work more than 40 hour weeks, and if there is overtime one week, that the next week should not include more overtime. Since the development cycles are short cycles of continuous integration, and full development (release) cycles are more frequent, the projects in XP do not follow the typical crunch time that other projects require (requiring overtime).”[27];
- *simple design:* this part of the iteration has in view to include only the code necessary to reach the desired results communicated by the client in each step of the process;
- *test-driven development:* in this part the functionality of the code “pieces” (classes, methods) are being tested; „Within XP, unit tests are written before the eventual code is coded. This approach is intended to stimulate the programmer to think about conditions in which his or her code could fail.”[27];
- *refactoring*: this part supposes improving and correcting what has been developed in the current iteration;
- *pair programming*: the code of the software product which is to be developed is written by “teams” made of two programmers working on the same computer; this increases the efficiency and decreases the number of errors in the code.

At present many companies use the Agile methodologies completely or only partially, or combinations of these methodologies not being aware that they are using the Agile methodology. The main aim of these methodologies is to decrease the risk of a project failure.

C. Agile Methodologies for Developing e-Government Type Information Systems

E-government type information systems represent a relatively new “field”, originating at the beginning of the 1990s. Ever since a series of projects have been started both at Romanian and world level: “Governments around the world are implementing innovative e-Government systems and services.”[32], research work, international conferences with the presentation of the results obtained, etc, all aiming to computerize the activity of public administrations as well as the interaction between citizens/companies and the state. Despite all these, in 2010 Romania, most of the citizens are still paying their taxes at the front desk and the contact with local administrations is still performed by traditional means: the physical presence at the front desk, telephone, fax, and, in the happiest cases, the e-government information systems are used by the state to provide its citizens with information. If we were to particularize_e-government systems to e-voting or e-democracy we would find out, at least at Romanian level, that they are almost inexistent.

And yet, where is the mistake in the development of these e-government systems so that the main clients (citizens and companies) use them to such a small extent, at least at the level of Romania? Why does this happen today when we live the “technological age”, “the Internet age”, when the Internet has become accessible to almost anyone, almost anywhere in the world? Obviously, the information systems developed so far are reliable and efficient from a technical point of view. The “problem” of these systems does not lie here. The real “problem” lies in the fact that that they do not really meet the citizens’ needs and they do not manage to fit like a “hand in glove” the middle-aged population’s way of thinking and understanding. We should also take into account the fact that this segment of population is les acquainted with the new technologies than younger people.

And yet, what should we do? As we talk about a “field” which is still “at the beginning”, as we still do not have a “recipe” for success as far as the e-government systems are concerned, *the use of Agile technologies for the development of these systems may prove to be a potential source of success for the projects started in the “field” of e-government*. As we mentioned above, the Agile technologies involve, firstly, working with iterations, which means dividing a big-sized project into smaller ones and, in each small project, the team which deals with the development will go through all the common stages of an independent project: collection of requirements, analysis, design, development, testing and implementation. What we *suggest* is selecting an e-government project which is to be developed and dividing it into iterations so that to each iteration will correspond a module or functionality of the project finally resulting into an e-government mini-system which could be put into practice. With the aid of the feedback from the citizens/companies, the teams dealing with the development of the whole project would be able to know how to continue the development of the system, where was the error in the first iteration and what should further be improved so that the final system may meet the citizens/companies’ needs and expectations. What is more, at the end of the project, the whole system may be utilized by the population of all ages as it is desirable that the use of the web-governmental sites become one of the fastest increasing activities in “the world of Internet”, “giving rise to e-government”[33].

IV. Information Systems Using Data Warehouses

A data warehouse is a relational database that is designed for query and analysis. “The data warehouse is, in fact, a database used for decision-making, totally separate from the operational database of a company.”[34] It usually contains historical data derived from transaction data, and can include data from multiple sources.

A data warehouse environment can include an extraction, transportation, transformation, and loading (ETL) solution, analysis, reporting, data mining capabilities, other applications that manage the process of gathering data, transforming it into useful information to business users.

A classic approach to providing a single version of the truth, where multiple transactional applications exist, is to build a data warehouse. This is a common practice where the goal is to store and analyze years of transactional history and where data quality in source systems is a known issue. However, alternative integration strategies are sometimes used where only recent transactions are needed for business intelligence.

Oracle Warehouse Builder (OWB) is an infrastructure builders' tool most often used during the design and deployment of data warehouses. The data warehouse design might include schema in third normal form, star schema, OLAP cubes, or as a hybrid schema of multiple types. OWB provides an interface to also define the source to target data extraction, transformation and load (ETL) mappings, generate the ETL script, coordinate workflows, and perform metadata management. OWB-generated scripts can pull data from other Oracle databases, other relational sources accessible via ODBC or Oracle Transparent Gateways, and from flat files with fixed width or delimited columns. Integrators are available to enable ease in building extractions from source tables in the Oracle E-Business Suite, PeopleSoft, or SAP applications. The generated scripts can be scheduled using Oracle Enterprise Manager or other popular schedulers and can leverage Oracle Workflow.

If you decide that data warehouses offered as applications (i.e. Oracle/PeopleSoft EPM, Oracle/Siebel Business Analytics Application) are not a good match for your business you may determine it is better to build a custom data warehouse.

After the data warehouse is up and working we would go to try to obtain a better data warehouse performance and how other way than by optimizing.

One of the most common causes of poor performance is poor application or data warehouse design. Such designs should provide optimal storage and maintainability and access to the underlying data in a flexible and highly performing manner. To achieve these design goals, the architect and administrator leverage modeling approaches such as star schema and hybrid designs and utilize data warehousing features such as bitmap indexes, partitioning, and materialized views.

A. *Data Warehouse Performance - Query Rewrite*

When base tables contain large amounts of data, it is expensive and time consuming (queries can take minutes or even hours) to compute the required aggregates or to compute joins between these tables. Because materialized views contain already precomputed aggregates and joins, Oracle Database employs a powerful process called query rewrite to quickly answer the query using materialized views.

The benefit of creating and maintaining materialized views is the ability to take advantage of query rewrite, which transforms a SQL statement expressed in terms of tables or views into a statement accessing one or more materialized views that are defined on the detail tables. The transformation is transparent to the end user or application, requiring no intervention and no reference to the materialized view in the SQL statement. Materialized views can be added or dropped without invalidating the SQL in the application code. A query undergoes several checks to determine whether it is a candidate for query rewrite. If the query fails any of the checks, the query is applied to the detail tables and can be costly in terms of response time and processing power.

The optimizer uses two different methods to recognize when to rewrite a query in terms of a materialized view. The first method is based on matching the SQL text of the query with the SQL text of the materialized view definition. If the first method fails, the optimizer uses the more general method in which it compares joins, selections, data columns, grouping columns, and aggregate functions between the query and materialized views.

a) Cost-Based Query Rewrite Process

Query rewrite is available only with the cost-based optimizer but Oracle optimizes the input query with and without rewrite and selects the least costly alternative.

If the rewrite logic has a choice between multiple materialized views to rewrite a query block, it selects the one that can result in reading the least amount of data.

After a materialized view has been picked for a rewrite, the optimizer performs the rewrite and then tests whether the rewritten query can be rewritten further with another materialized view and the process continues until no further rewrites are possible.

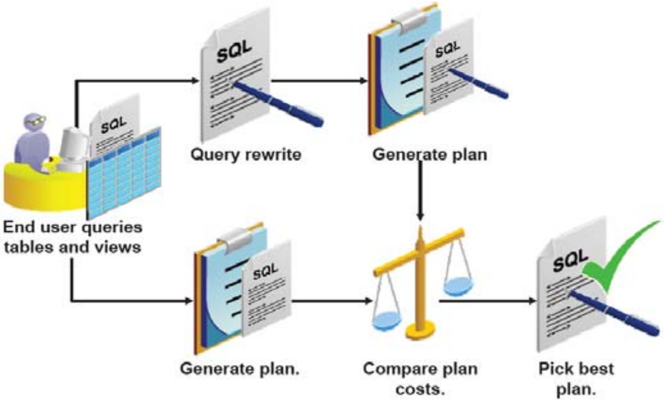


Fig.7- Cost-Based Query Rewrite Process; source: [35]

Query rewrite operates on queries and subqueries in the following types of SQL statements: SELECT, CREATE TABLE … AS SELECT, INSERT INTO … SELECT, DELETE and UPDATE. It also operates on subqueries in set operators: UNION, UNION ALL, INTERSECT, and MINUS.

There are several conditions required for Oracle to rewrite a query and the most important are:

- Query rewrite must be enabled for the session.
- A materialized view must be enabled for query rewrite.
- The rewrite integrity level should allow the use of the materialized view.

There are three Oracle initialization parameters that affect query rewrite: OPTIMIZER_MODE, QUERY_REWRITE_ENABLED and QUERY_REWRITE_INTEGRITY.

OPTIMIZER_MODE can have the values: first_rows_n(the optimizer uses a cost-based approach and optimizes with a goal of best response time to return the first n rows), first_rows (the optimizer uses a mix of costs and heuristics to find a best plan for fast delivery of the first few rows) and all_rows (the optimizer uses a cost-based approach for all SQL statements in the session and optimizes with a goal of best throughput).

QUERY_REWRITE_ENABLED can have the value of TRUE (enabling the query rewrite feature of the optimizer), FALSE or FORCE (enables the query rewrite feature of the optimizer and directs the optimizer to rewrite queries using materialized views even when the estimated query cost of the unwritten query is lower).

QUERY_REWRITE_INTEGRITY determines the integrity level and can have three values: ENFORCED(enables query rewrites only if the Oracle server can guarantee consistency), TRUSTED and STALE_TOLERATED(uses materialized views that are valid but that can contain stale data).

There are several query rewrite hints that can help when written in a query to instruct the optimizer to rewrite or not a query by using materialized views, they are: REWRITE, NOREWRITE and REWRITE_OR_ERROR.

Because query rewrite occurs transparently, it is not always evident that it has taken place. There are three ways to confirm that the query rewrite has occurred, using: EXPLAIN PLAN or DBMS_MVIEW.EXPLAIN_REWRITE procedure or V$SQL_PLAN.

- Example of using EXPLAIN PLAN with Query Rewrite

```
CREATE MATERIALIZED VIEW mv_sales
ENABLE QUERY REWRITE AS
SELECT t. month, SUM(s.qnt_sold) AS money
FROM sales s, time t
WHERE s.time_id = t.time_id
GROUP BY t.month;

EXPLAIN PLAN FOR
SELECT t.month, SUM(s.qnt_sold)
FROM sales s, time t WHERE s.time_id = t.time_id
GROUP BY t.month;

SELECT OPERATION, OBJECT_NAME FROM
PLAN_TABLE;

EXPLAIN PLAN succeeded.

OPERATION                          OBJECT_NAME
-----------------------------------------------------------------------
SELECT STATEMENT
MAT_VIEW REWRITE ACCESS            mv_sales
```

You can use the EXPLAIN PLAN facility with query rewrites to check whether or not the operation shows MAT_VIEW REWRITE ACCESS. If it does, then query rewrite has occurred.

- Example of using DBMS_MVIEW.EXPLAIN_REWRITE procedure with Query Rewrite

```
DECLARE
qrytext VARCHAR2(2000) :=
'SELECT cust_name, SUM(qnt_sold)
FROM sales s, customer c
WHERE s.cust_id = c.cust_id
GROUP BY cust_name';
BEGIN
DBMS_MVIEW.EXPLAIN_REWRITE(qrytext,'smv','124');
END;

SELECT message FROM rewrite_table
WHERE statement_id = '124' ORDER BY sequence;

MESSAGE
-----------------------------------
QSM-01001: query rewrite not enabled
```

It can be difficult to understand why a query did not rewrite. The rules that govern query rewrite eligibility are complex, involving various factors such as constraints, dimensions, query rewrite integrity modes, freshness of the materialized views, and the types of queries themselves.
In addition, you may want to know why query rewrite chose one particular materialized view instead of another.

You can use the DBMS_MVIEW.EXPLAIN_REWRITE procedure to determine whether a query can be rewritten and, if not, why not. You can obtain the output from DBMS_MVIEW.EXPLAIN_REWRITE in two ways. The first is to use a table, while the second is to create a VARRAY. Above we presented an example for using an output table and the output from the query of REWRITE_TABLE, the reason is that query rewrite is not enabled.

b) Query Rewrite Methods

The optimizer uses a number of different methods to rewrite a query, they are divided into two categories: text and general rewrite.

General Query Rewrite Methods

When the text comparison test fails, the Oracle optimizer performs a series of generalized checks as data relationships on which it can depend (primary key and foreign key), NOT NULL and they are very important for query rewrite because they tell what type of result is produced by joins, grouping, or aggregation of data.

Text Match Rewrite Methods

The optimizer uses different methods to rewrite a query. The simplest case occurs when the result stored in a materialized view exactly matches what is requested by a query (the number of queries eligible for this type of query rewrite is minimal). The Oracle optimizer makes this type of determination by comparing the text of the query with the text of the materialized view definition.

For the text match check, the optimizer uses two methods:

- *Full text match:* In full text match, the entire text of a query is compared with the entire text of a materialized view definition (that is, the entire SELECT expression).
- *Partial text match:* When full text match fails, the optimizer attempts a partial text match. In this method, the text starting with the FROM clause of a query is compared with the text starting with the FROM clause of a materialized view definition.

Example, if the query requests AVG(expr), the materialized view's SELECT list must contain either AVG(expr), or SUM(expr) and COUNT(expr).

Expression Matching Query Rewrite

This type of query rewrite is based on matching an expression that appears in a query with an equivalent expression that appears in the SELECT list of a materialized view. Because a materialized view stores precomputed results of an expression, any query that is rewritten to use such a materialized view benefits through the performance improvement achieved by obviating the need for expression computation.

- Data Warehouse Performance –Example of using full text matching with Query Rewrite

```
CREATE MATERIALIZED VIEW sales_by_year_mv
(age_interval, qnt_sold)
ENABLE QUERY REWRITE AS
SELECT TO_CHAR( (2000-c.cust_year_)/10-0.5,999),
SUM(s.qnt_sold)
FROM sales s, customer c
WHERE s.cust_id = c.cust_id
GROUP BY TO_CHAR((2000-c.cust_year)/10-0.5,999);
```

The sales_by_year_mv materialized view includes a full expression that can be used by rewritten queries.

```
SELECT TO_CHAR((2000-c.cust_year_)/10-0.5,999),
SUM(s.qnt_sold)
FROM sales s, customer c
WHERE s.cust_id = c.cust_id
GROUP BY TO_CHAR(
(2000-c.cust_year)/10-0.5,999);
```

Query is rewritten

```
SELECT age_interval, qnt_sold
FROM sales_by_year_mv;
```

c) Partition Change Tracking (PCT) and Query Rewrite

The Oracle server uses metadata to maintain staleness at a finer granularity. When a certain partition of one detail table is updated, only specific sections of the materialized view are marked stale. This feature is often called Partition Change Tracking (PCT). The materialized view must have information that can identify the partition of the table corresponding to a particular row or group of the materialized view. The simplest scenario is when the partitioning key of the table is available in the SELECT (and GROUP BY) list of the materialized view. Query rewrite can use a materialized view in ENFORCED (or TRUSTED) mode provided that the rows from the materialized view used to answer the query are known to be FRESH.

The fresh rows in the materialized view are identified by adding selection predicates to the materialized view's WHERE clause (in the data dictionary). A query is rewritten with this materialized view if its answer is contained within this (restricted) materialized view. Instead of the partitioning key, a partition marker may be present in the SELECT (and GROUP BY) list of the materialized view. The Oracle server can use the materialized view to rewrite queries that require data from only certain partitions (identifiable by the partition marker), for instance, queries that reference a partition-extended table name or queries that have a predicate specifying ranges of the partitioning keys containing entire partitions.

- Data Warehouse Performance – Example of using Partition Change Tracking with Query Rewrite

```
CREATE TABLE sales_2 (time_id DATE NOT NULL,
prod_id NUMBER NOT NULL,
cust_id NUMBER NOT NULL,
channel_id CHAR(1) NOT NULL,
promo_id NUMBER NOT NULL,
quanty_sold NUMBER NOT NULL,
amount_sold NUMBER NOT NULL)
PARTITION BY RANGE (time_id)
( PARTITION q1 VALUES LESS THAN ('1-APR-2010'),
PARTITION q2 VALUES LESS THAN ('1-JUL-2010'),
PARTITION q3 VALUES LESS THAN ('1-OCT-2010'),
PARTITION q4 VALUES LESS THAN ('1-JAN-2011')
);
```

Create the SP_MV materialized view:

```
CREATE MATERIALIZED VIEW sp_mv
ENABLE QUERY REWRITE AS
SELECT time_id, prod_name, SUM(qnt_sold) AS qnts
FROM product p, sale s
WHERE s.prod_id = p.prod_id
GROUP BY time_id, prod_name;
```

Insert data into the third partition of the SALE table:

```
INSERT . . .('1-JUL-2010' <= time_id < '1-OCT-2010'). .
```

The materialized view query reflects the fresh rows:

```
. . .
WHERE s.prod_id = s.store_id AND
time_id < '1-JUL-2010' OR time_id >= '1-OCT-2010'
```

The materialized view named sp_mv that references the two tables, SALE and PRODUCT. Note that sp_mv includes the partition key column from the SALE table (TIME_ID) in both its SELECT and GROUP BY lists. This enables PCT on the SALE table for the sp_mv materialized view.

If new data for SALE are inserted into its third partition, the Oracle server (during a rewrite) can modify the materialized view's definition to reflect the fresh rows.

The following query:

```
SELECT time_id, SUM(qnt_sold) AS qnts
FROM product p, sale s
WHERE s.prod_id = p.prod_id AND
time_id < '1-JUL-2010'
GROUP BY time_id;
```

Can be rewritten as follows:

```
SELECT time_id, SUM(qnts)
FROM sp_mv
WHERE time_id < '1-JUL-2010'
GROUP BY time_id;
```

- Recommendations for creating indexes for materialized views

After a query is rewritten, query execution may need to access any subset of the materialized view key columns and may need to join and aggregate over a subset of those columns. Query execution usually performs best if a single-column bitmap index is defined on each materialized view key column.

For materialized views that contain only joins using fast refresh, the recommendation is to create indexes on columns that contain row IDs to improve refresh performance.

d) Query Rewrite Enhancement to Support Queries Containing Inline Views

Oracle Warehouse supports query rewrite with inline views in two circumstances:

- When the text from the inline views in the materialized view *exactly matches* the text in the request query, or
- When the request query contains inline views that are *equivalent* to the inline views in the materialized view.

- Data Warehouse Performance – Example of using materialized view that contains an inline view with Query Rewrite

```
CREATE MATERIALIZED VIEW SUM_SALES_MV
ENABLE QUERY REWRITE AS
SELECT mv_iv.prod_id, mv_iv.cust_id,
sum(mv_iv.amount_sold) sum_amount_sold
FROM (SELECT products.prod_id, cust_id, amount_sold
FROM sale, product
WHERE sales.prod_id = products.prod_id) MV_IV
GROUP BY mv_iv.prod_id, mv_iv.cust_id;
```

The text of the Inline View matches exactly the text of the materialized view; therefore, the *Query1* below is rewritten with the materialized view.

```
SELECT iv.prod_id, iv.cust_id,
SUM(iv.amount_sold) sum_amount_sold
FROM (SELECT products.prod_id, cust_id, amount_sold
FROM sale, product
WHERE sales.prod_id = products.prod.id) IV
GROUP BY iv.prod_id, iv.cust_id;
```

The text of the Inline View doesn't match the text of the materialized view however, they are equivalent.

Query2

```
SELECT iv.prod_id, iv.cust_id,
SUM(iv.amount_sold) sum_amount_sold
FROM (SELECT products.prod_id, cust_id, amount_sold
FROM product, sale
WHERE sales.prod_id = products.prod_id) IV
GROUP BY iv.prod_id, iv.cust_id;
```

Query1 and Query2 are both first transformed as follows:

```
SELECT prod_id, cust_id, sum(amount_sold)
FROM MV_IV
GROUP BY prod_id, cust_id;
```

Query1 and Query2 are both at the end re-written as follows:

```
SELECT prod_id, cust_id, sum_amount_sold
FROM SUM_SALES_MV;
```

e) Query Rewrite Using Remote Tables

Query rewrite can reference remote objects using MVs which reference the remote objects. This reduces or eliminates the data from network (network round trips, which are costly operations).

- *Data Warehouse Performance – Example of using remote tables with Query Rewrite*

The materialized view is present at the local site, but it references tables that are found at the remote site. The query in the second example below contains tables that are found at a single remote site. Even though the query references remote tables, it is rewritten using the materialized view as shown in the third example below.

```
CREATE MATERIALIZED VIEW
sum_sales_prod_week_mv
ENABLE QUERY REWRITE AS
SELECT p.prod_id, t.week_ending_day, s.cust_id,
SUM(s.amount_sold) AS sum_amount_sold
FROM sales@remotedbl s,products@remotedbl p,
times@remotedbl t
WHERE s.time_id=t.time_id AND s.prod_id=p.prod_id
GROUP BY p.prod_id, t.week_ending_day, s.cust_id;
```

```
SELECT p.prod_id, t.week_ending_day, s.cust_id,
SUM(s.amount_sold) AS sum_amount_sold
FROM sales@remotedbl s, products@remotedbl p,
times@remotedbl t
WHERE s.time_id=t.time_id AND s.prod_id=p.prod_id
GROUP BY p.prod_id, t.week_ending_day, s.cust_id;

SELECT prod_id, week_ending_day, cust_id,
sum_amount_sold
FROM sum_sales_prod_week_mv;
```

B. Data Warehouse – A Technology Adopted by the Government Institutions, too

Typically, for a large company, using Data Warehouse technology may even prove be the essential factor that makes the difference between the company's success and its failure. That is because, first of all, a large company is normally processing large volumes of data each day, and Data Warehouse technology is known for its capacity to allow very speedy data processing (usually in the form of reports).

State institutions, too, are "processing" every day very large amounts of data; nowadays, with the constant development of new information technologies and of the Internet, "e-Government portals will be increasingly used by the citizens of many countries to access a set of services" [40]. Moreover, as "electronic governments (e-governments) are fast emerging replacing functions performed by traditional governments"[41], we may expect that, for example, once e-government systems shall have become the one and only way available for citizens to pay their taxes and duties (at least in Romania), the volume of data and information stored on electronic support will increase significantly in a very short while. It should be added also that the front-office of the information system (system interface with citizens) is going to be doubled by the back-office module (system interface with the civil servant who processes, manages and administers payments). Therefore, the automation alone of the tax payment flow at city level, for example, will generate a huge volume of information. It is also worth mentioning that, alike managers in big companies, the managers within state institutions will need statistics and reports in their decision making. The reports will have to provide information aggregated in both tabular and graphical format as quickly as possible. However, ERP systems currently employed by companies have already successfully enabled the automation of some similar processes, yet in other fields of activity. This way, a successful development and implementation of ERP systems within companies may also become a solution in the case of e-government systems, there where no answer has been found yet for e-government systems to the problems encountered.

V. Conclusion

The major challenge during the implementation of an ERP is finding the appropriate balance between the standard set of functionalities and the necessary customization for adapting the system to the customer's needs. Problems with ERP systems are mainly due to the fact that customization of the ERP software is limited. The process of re-engineering of business flows in order to ensure compatibility with the standard version of the ERP applications may lead to a loss of competitive advantage. Therefore, companies prefer to develop their own ERP systems, using the latest information technologies to achieve a more efficient system. The success of the future information system might, however, be jeopardized, where, for example, steps are skipped along the normal course of action to be followed for system development (e.g. due to lack of time, the stages of analysis and design implemented by means of information methodologies will most often than not be reduced in practice to a simple track keeping of claims raised by customers), or where corporate IT experts do not take the best decisions when implementing the system (e.g. they choose database technology to obtain reports containing data aggregated for very long periods of time, instead of opting for data warehouses).

With this paper we believe "we have made a step forward", analyzing and presenting evidence of the benefits and importance of technologies that can be used successfully by companies in developing ERP systems, technologies which, though increasingly "popular" in recent, not all IT professionals are familiar with. We are referring here to data warehousing technology and information methodologies, in particular to the "people – oriented" one, such as Agile, for example.

Also, this paper has "pointed out" the importance of the opportunity to "learn" from the "experience" of developing ERP systems and to apply the knowledge acquired and the technologies tested on ERP systems in creating e-government types of systems.

References

[1] 2010 Gartner FEI technology study: the CFO's perspective on ERP [Online]. Available: http://www.gartner.com/DisplayDocument?id=1337813&ref=g_sitelink&ref=g_SiteLink

[2] When and why you should customize ERP software [Online]. Available: http://www.microsoft.com/midsizebusiness/increase-business-value/customizing-erp.mspx

[3] A license to drive ERP adoption [Online]. Available: http://www.focus.com/briefs/erp/license-drive-erp-adoption

[4] F. De Maria, C. Briano, M. Brandolini, E. Briano and R. Revetria, "Integrating back-office & retail store management in ERP through simulation models", in *Proc. of the 4th WSEAS International Conference on Computer Engineering and Applications (CEA '10)*,USA, 2010, pp. 44-52.

[5] Drawing the line between ERP configuration and customization [Online]. Available: http://blog.slingshotsoftware.com/bid/38280/Drawing-the-Line-Between-ERP-Configuration-and-Customization

[6] X. Liao, Y. Li and B. Lu, "A model for selecting an ERP system based on linguistic information processing", *Information Systems*, no. 32, 2007, pp. 1005–1017 (ISI Thomson).

[7] G. Cucui, "Considerations on software engineers and project management of ICT", *Annales Universitatis Apulensis Series Oeconomica (University „1 Decembrie 1918" Alba Iulia, Facultaty of Science)*, vol. 2, no. 8, 2006.

[8] K. Siau and Y. Wang, "Cognitive evaluation of information modeling methods", *Information and Software Technology*, no. 49, 2007, pp. 455–474 (ISI Thomson).

[9] I. Chivu, D. Popescu, The effects of implementing SSADM (Structured System Analysis and Design Methodology) in the management process of European public institutions - British experience in the use of SSADM [Online]. Available: http://www.miteapl.ro/r1/studiu-1.pdf

[10] Information systems and information applications in business administration, University „Lucian Blaga" from Sibiu, the Faculty of Economic Sciences – course notes [Online]. Available: http://webspace.ulbsibiu.ro/florin.martin

[11] Information systems modeling [Online]. Available: http://cndiptfsetic.tvet.ro/materiale/Materiale_de_invatare/AP/MI_Modulul%20X%20-%20Mirela%20Iacob.doc#_Toc113877021

[12] Projection of an information system [Online]. Available: http://www.scritube.com/stiinta/informatica/Proiectarea-unui-sistem-inform61886.php

[13] N. Trigoni, "Comparative development methodologies", Lecture 10, Department of Computer Science and Information Systems, Birkbeck College, University of London [Online]. Available: http://www.dcs.bbk.ac.uk/~niki/CDMlecture10.ppt

[14] M. Guran, *Information systems, information and communication in the modern enterprise infrastructure*, A.G.I.R. Publishing House, Bucharest, Romania, 2008.

[15] E. Song, S. Yin, I. Ray, "Using UML to model relational database operations", *Computer Standards & Interfaces*, no. 29, 2007, pp. 343–354 (ISI Thomson).

[16] J. Barjis, "The importance of business process modeling in software systems design", *Science of Computer Programming*, no. 71, 2008, pp. 73–87 (ISI Thomson).

[17] C. Flender, "Ecological modelling of information systems" in *Proc. of the 20th International Conference on Advanced Information Systems Engineering*, 2008, Montpellier, France.

[18] F. Aburub, M. Odeh, I. Beeson, "Modelling non-functional requirements of business processes", *Information and Software Technology*, no. 49, 2007, pp. 1162–1171 (ISI Thomson).

[19] Information systems development methodologies: making the right choice [Online]. Available: http://www.cems.uwe.ac.uk

[20] Information systems [Online]. Available : http://www.icbe-ct.com/pacuraru

[21] H. Zhang, R. Kishore, R. Sharman, R. Ramesh, "Agile Integration Modeling Language (AIML): A conceptual modeling grammar for agile integrative business information systems", *Decision Support Systems*, no. 44, 2007, pp. 266–284 (ISI Thomson).

[22] Agile software development [Online]. Available: http://en.wikipedia.org/wiki/Agile_software_development

[23] IT Project Management – Agile methodology for project management software [Online]. Available: http://www.bobyte.ro/IT_PROJECT_MANAGEMENT/Metodologia_AGILE_de_Management_Proiecte_Software.htm

[24] Agile Interfaces - PMO Integration [Online]. Available: http://leadinganswers.typepad.com/leading_answers/2007/06/agile-interface.html#tp

[25] Scrum is an innovative approach to getting work done [Online]. Available: http://www.scrumalliance.org/learn_about_scrum

[26] Agile methodologies for IT projects: SCRUM and Extreme Programming (XP) [Online]. Available: http://www.trilex.ro/Metodologii/IT-Agile-XP-SCRUM.htm

[27] Extreme Programming [Online]. Available: http://en.wikipedia.org/wiki/Extreme_Programming

[28] Extreme Programming [Online]. Available: http://www.webopedia.com/TERM/E/Extreme_Programming.html

[29] N. Trigoni, Comparative development methodologies, Lecture 9, Department of Computer Science and Information Systems, Birkbeck College, University of London [Online]. Available: http://www.dcs.bbk.ac.uk/~niki/CDMlecture9.ppt

[30] What is Extreme Programming? [Online]. Available: http://xprogramming.com/book/whatisxp

[31] J. Highsmith (Cutter Consortium), "Extreme Programming – Agile project management advisory service white paper", USA, 2002.

[32] J. Lee, "10 year retrospect on stage models of e-Government: A qualitative meta-synthesis", *Government Information Quarterly*, no. 27, 2010, pp. 220–230 (ISI Thomson).

[33] I. Gallego-Alvarez, L. Rodriguez-Dominguez, I.-M. Garcia-Sanchez, "Are determining factors of municipal e-Government common to a worldwide municipal view? An intra-country comparison", *Government Information Quarterly*, no. 27, 2010, pp. 423-430 (ISI Thomson).

[34] A. Mocanu, D. Litan, S. Olaru and A. Munteanu, "Information systems in the knowledge based economy", *WSEAS TRANSACTIONS on BUSINESS and ECONOMICS*, issue 1, vol. 7, 2010, pp. 11-21.

[35] www.otn.oracle.com

[36] M. Teohari and L. Copcea, "XML authoring tool" in *Proc. of the 4th European Computing Conference (WSEAS-Conference, ECC'10)*, 2010, pp. 143-147.

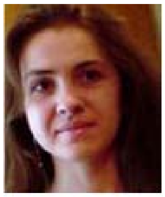
[37] R. Stackowiak, J. Rayman and R. Greenwald, *Oracle data warehousing and business intelligence solutions*, Wiley Publishing, Inc., Canada, 2009.

[38] R. Wrembel and C. Koncilia, *Data warehouses and OLAP: concepts, architectures and solutions*, IRM Press Hershey, Germany, 2009.

[39] http://www.erp.com

[40] K. T. Al-Sarayreh, R. E. Al-Qutaish and M. D. Al-Majali, Incorporating the biometric voice technology into the e-Government systems to enhance the user verification, *WSEAS TRANSACTIONS on COMPUTERS*, issue 5, vol. 7, 2008, pp. 435-452.

[41] M. Alrazooqi and R. de SILVA, An m-Government solution proposal for Dubai government, in *Proc. of the 9th WSEAS International Conference on TELECOMMUNICATIONS and INFORMATICS (TELE-INFO'10)*, 2010, Italy, pp. 95-102.



D. Litan has graduated The Bucharest Academy of Economic Studies (Romania), Faculty of Cybernetics, Statistics and Economic Informatics in 2006. She holds a Master diploma in Databases - Support for business from 2009 and in present she is a Ph.D. Candidate in Economic Informatics with the Doctor's Degree Thesis: The integration of new technologies in the information systems of e-Government.

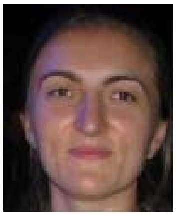
Her research activity can be observed in the following achievements: 10 proceedings, among witch:

- "Ways to Increase the Efficiency of Information Systems", *Proc. of the 10th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Databases (AIKED '11, University of Cambridge)*, February 20-22, 2011, Cambridge, UK;
- "Modern Information Technologies Used in Market Research", *Proc. of the 9th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics (CIMMACS '10)*, December 14-16, 2010, Merida, Venezuela;
- "Business Intelligence and Data warehouse – Technological Support for Decisional Management in Geographical Information Systems", *Proc. of the 3rd International Conference on Communications and Information Technology (CIT'09)*, December 29-31, 2009, Athens, Greece;

and 5 articles published in scientific reviews , among witch:

- "Information systems in the knowledge based economy", *Journal: WSEAS TRANSACTIONS on BUSINESS and ECONOMICS*, issue:1, vol. 7, 2010
- "Modern Database Machines", *Journal: Informatics Economics*, no. 2, 2010.

Her scientific fields of interest include: Databases, Database Management Systems, Programming, Information Systems and Economics.



A. Apostu has graduated The Academy of Economic Studies from Bucharest (Romania), Faculty of Cybernetics, Statistics and Economic Informatics in 2008. She has a Master diploma in Economic Informatics from 2010 and in present she is a Ph.D. Candidate in Economic Informatics with the Doctor's Degree Thesis: "Informatic solution in a distributed environment regarding unitary tracking of prices".

Her scientific fields of interest include: Economics, Databases, Programming, Information Systems, Information Security and Distributed Systems.

L. Copcea (Teohari) has graduated The Bucharest Academy of Economic Studies (Romania), Faculty of Cybernetics, Statistics and Economic Informatics in 2006. She holds a Master diploma in Databases - Support for business from 2008 and in present she is a Ph.D. Candidate in Economic Informatics with the Doctor’s Degree Thesis: Master data management in data warehouses.

Her research activity can be observed in the following achievement: “Proceedings of the 4th European Computing Conference (ECC'10), 2010 (University Politehnica of Bucharest) April 20-22, 2010, Bucharest, Romania.

Her scientific fields of interest include: Data warehouse, Databases, Database Management Systems, High availability solutions, Information Systems and Economics.

M. Teohari has graduated The Bucharest Academy of Economic Studies (Romania), Faculty of Cybernetics, Statistics and Economic Informatics in 2004. He is a Ph.D. Candidate in Economic Informatics with the Doctor’s Degree Thesis: XML technologies for databases.

His research activity can be observed in the following achievement: “Proceedings of the 4th European Computing Conference (ECC'10), 2010 (University Politehnica of Bucharest) April 20-22, 2010, Bucharest, Romania.

His scientific fields of interest include: XML, Databases, Data warehouse, Information Systems and Economics.