The Interaction Of The Web Technologies In Integrated Marketing Information System

Veselina Ivanova Nedeva[1]

Abstract: **The report presents the different approaches of interaction of the information technologies, which now have their place in the integrated marketing information system. They are defined as wall as the script language so as the tasks of IMIS. In this report we have presented a few basic variants of the interaction, which reflect the specific features IMIS.**

Keywords: **Data Warehouse, Data Mining, OLAP, architecture of information system.**

I. INTRODUCTION:

The report presents different approaches of the interaction of the information technologies, which now have their place in the integrated marketing information system. The script language and the tasks of Integrated Marketing Information System (IMIS) define them.

In the process of analysis the main objects and their relations are defined. We explain the reflecting behavior on the main objects. Diagrams of the interaction objects are developed. The determination of the objects and scenarios is connected to information technologies and consumer services.

The specialists, participating in the creation of IMIS, have their place and role in process of the designing and making the information systems. IMIS creates the balance between information necessities and needs on one side and information technologies and necessities on the other side. If the information technologies are not adapted to the condition of the company, they will be inefficient and the money than have been put for their introduction are of no use. On other side Data warehouse (DW), Data Mining (DM) and OLAP (Online Analytical Processing) might be based on these data, thus creating seriously problems for the information security.

II. THE SOURCES OF INFORMATION FOR MARKETING INFORMATION SYSTEM

To comply with the requirements of the users concerning the preparation and the decision making, IMIS must give information from different sources - an operative system, geodemographics systems and historical data from the system from past reporting period. They should support high quality of data in order to give support for the purpose of making decisions and using the possibility of online analysis. The extraction, transformation and consolidation of the data from operative systems require much time and resources. Creating the multi-objective, consolidated and spreader over the territory of the corporation information archives requires coordination of the different data models. Also the data from the operative systems have to be standardized in order to make them stable.

In IMIS the technology DW is used. The obvious advantage of the DW is the quality of the data – it is prematurely “filtered” (the unnecessary and doubled information is removed); it is complete, because it consists of the whole available data from the operative systems of the external sources – GDIS and the succeeded systems; DW is constantly updating; it consists of details and summarized data, which is a good base for study and analysis and shortens the time for their conduct; DW is consolidated over a single star – scheme; it is integrated in a common information data base of IMIS; DW is available on-line for all users.

[1]Veselina Iv.Nedeva, Assist.Prof.Ph.D., Technical college – Yambol, Gr.Ignatiev Str. 38, Yambol 8600, BULGARIA,
e-mail: vnedeva@tk.uni-sz.bg

III. THE ARCHITECTURE OF MARKETING INFORMATION SYSTEM

The architecture and technology that evolved to answer this demand in IMIS was client/server, in the guise of a two-tiered approach. By replacing the file server with a true database server, the network could respond to client requests with just the answer to a query against a relational DBMS. One benefit to this approach, then, is to significantly reduce network traffic. Also, with a real DBMS, true multi-user updating is now easily available to users on the PC LAN.

A. 2-tier architecture

In a 2-tier client/server architecture, SQL are typically used to communicate between the client and server. The server is likely to have support for stored procedures and triggers. These mean that the server can be programmed to implement business rules that are better suited to run on the server than the client, resulting in a much more efficient overall system. 2-tiered client/server approach is a good and economical solution for certain classes of problems.

The 2-tiered client/server architecture has proven to be very effective in solving workgroup problems. "Workgroup", as used here, is loosely defined as a dozen to 100 people interacting on a LAN. For bigger, enterprise-class problems and/or applications that are distributed over a WAN, use of this 2-tier approach has generated some problems.

What typically happens with client/server in large enterprise environments is that the performance of a 2-tier architecture deteriorates as the number of on-line users increases. If something happens to the connection, the client must go through a session reinitiating process. With 50 clients and today's typical PC hardware, this is no problem. When one has 2,000 clients on a single server, however, the resulting performance isn't likely to be satisfactory.

The data language used to implement server procedures in SQL server type data base management systems is proprietary to each vendor. Oracle, Sybase, Informix and IBM, for example, have implemented different language extensions for these functions. Proprietary approaches are fine from a performance point of view, but are a disadvantage for users who wish to maintain flexibility and choice in which DBMS is used with their applications.

Another problem with the 2-tiered approach is that current implementations provide no flexibility in "after the fact partitioning". Once an application is developed it isn't easy to move some of the program functionality from one server to another. This would require manually regenerating procedural code. In some of the newer 3-tiered approaches to be discussed below, tools offer the capability to "drag and drop" application code modules onto different computers.

The response to limitations in the 2-tier architecture has been to add a third, middle tier, between the input/output device (PC on your desktop) and the DBMS server. This middle layer can perform a number of different functions - queuing, application execution, database staging and so forth. The use of client/server technology with such a middle layer has been shown to offer considerably more performance and flexibility than a 2-tier approach.

Just to illustrate one advantage of a middle layer, if that middle tier can provide queuing, the synchronous process of the 2-tier approach becomes asynchronous. In other words, the client can deliver its request to the middle layer, disengage and be assured that a proper response will be forthcoming at a later time. In addition, the middle layer adds scheduling and prioritization for the work in process. The use of an architecture with such a middle layer is called "3-tier" or "multi-tier".

B. 3-Tier With a TP Monitor

The client connects to the TP monitor (transaction processing monitor) instead of the database server. The transaction is accepted by the monitor, which queues it and then takes responsibility for managing it to correct completion.

On-line access to mainframes was available through one of two metaphors – time-sharing or transaction processing (OLTP). Time-sharing was used for program development and the computer's resources were allocated with a simple scheduling algorithm like round robin. OLTP scheduling was more sophisticated and priority driven.

TP monitors (TP Heavy) have staged a comeback because their queuing engines provide a funnelling effect, reducing the number of threads a DBMS server needs to maintain. The client connects with the monitor, which accepts the message and queues it for processing against the database. Once the monitor has accepted the message, the client can be released for further processing. The synchronous session based computing of a 2-tier architecture, then, becomes asynchronous through the insertion of the TP monitor into the equation. The monitor smoothes out and lowers the overhead of accessing the database server.

Some other key services a monitor provides are: the ability to update multiple different DBMS in a single transaction; connectivity to a variety of data sources including flat files, non relational DBMS, and the mainframe; the ability to attach priorities to transactions; and robust security.

C. 3-Tier With a Messaging Server

Messaging provides still another technology to implement 3-tier computing. Messages are processed asynchronously with the appropriate priority level. And, like a TP monitor, a

Messaging server provides connectivity to data sources other than RDBMS. A message is a self-contained object that carries information about what it is, where it needs to go, and what should happen when it reaches its destination. There are at least two parts to every message; the header contains priority, and address and an ID number. The body of the message contains the information being sent, which can be anything-including text, images or transactions.

A primary difference from TP Monitors is that message server architecture is designed around intelligence in the message itself as opposed to a TP monitor environment, which places the system intelligence in the monitor or the process logic of the application server.

Messaging systems are designed for robustness. By using store and forward logic, they provide message delivery after and around failures. They also provide independence from the enabling technologies such as wired or wireless or protocols. They don't require a persistent connection between the client and server. They are robust because message delivery can be programmed to occur after or around failures. Because messaging systems support an emerging wireless infrastructure, they should become popular for supporting mobile and occasionally connected workers.

D. 3-Tier With an Application Server

When most people talk of 3-tier architectures, they mean the approach of an application server. With this approach most of the application's business logic is moved from the PC and into a common, shared host server. The PC is basically used for presentation services - not unlike the role that a terminal plays on a mainframe. Of course, because we are talking about a real PC here, it still has the advantages of being used for client side application integration (via OLE or other Approach) if desired.

The approach of putting business logic on a server offer a number of important advantages to the application designer: When less software is on the client, there is less worry about security since the important software is on a server in a more controlled environment. The resulting application is more scalable with an application server approach. With a middle application server tier it's much easier to design the application to be DBMS- agnostic. If you want to switch to another DBMS vendor, it's more achievable with reasonable effort with a single multithreaded application than with thousands of applications on PC's.

E. 3-Tier With an Object DBMS

A variation on this theme of application server is the idea of using an object DBMS (ODBMS) as the middle layer. Data in a relational DBMS is usually stored in normalized fashion across many tables and for access by different applications and users. This generalized form of storage may prove inadequate (performance wise) for the needs of any one particular application. An ODBMS can be used to retrieve the data from the common store, assemble it for efficient usage by your application, and provide a persistent store for that data as long as your application might need it.

Since extended data types like video or voice are not typically supported in today's RDBMS, those data types might also be stored in the ODBMS, which could then associate the appropriate multimedia data with the data retrieved from the RDBMS.

F. Distributed Components & the 3-Tier Architecture

This brings us to distributed object computing and components. Many software specialists are predicting a software future with the creation of application systems through assembly of software components. That kind of software approach is available today in a few proprietary object environments. The emergence of a broad based industry for component-based software will require the prior emergence of industry standards for interchangeable parts.

The distributed object implementation of client/server computing is going to change the way applications are built.

There should be some very interesting advantages to observe. If we needed fault tolerant computing, we could implement copies of objects onto multiple servers. That way if any were down, it would be possible to go to another site for service. With distributed objects being self contained and executable (all data and procedures present) it will be possible for a systems administrator to tune the performance of the network by moving those objects from overloaded hardware to under utilized computers.

Distributed object architecture should also offer other benefits for application developers. The same interface will be used for building a desktop, single location application or a fully distributed application. The application can be developed and tested locally and you'll know that it will work fine when it's distributed – you depend on the known services of an object request broker for distribution.

Since the application developer is dealing with an object request broker for transmission services, technical issues like queuing, timing and protocols aren't an issue for the application developer.

G. Data Warehouse & 3-Tier

A 3-tier architecture is also useful for data mining or warehouse types of applications. These applications are characterized by unanticipated browsing of historical data. The databases supporting this type of application can sometimes be huge (up to a few terabytes -10(12) bytes) and have to be structured properly for adequate performance (a few second turnaround).

Data mining and decision support applications typically need response times of a few seconds. If the system can't provide that kind of performance, the thought process of the human analyst is disrupted and the overall purpose of the system is foiled. A production database established for multiple users isn't typically in a form that can support ad-hoc inquiries. IT systems and operations managers usually don't want access to those tables to be on the mainframe.

Often this server is called OLAP - on-line analytical processor. In other circumstances this server can be a symmetric or massively parallel processor running an RDBMS.

IV. The Interaction of the Web Technologies

There are a number of technologies for data transformation and filtering techniques, which comply with these requirements. They examine the data preparation for DW while maintaining the quality and minimizing the risk during the process. The complexity of extracting, transforming and integrating of the data depends of the number and the variety of the sources. The process of extracting and integrating is almost impossible to be conducted with the traditional methods, because of the large amount of resources and the complex multi-step processing.

The connection between the user and DW is done by Meta-data. The meta-data has special status of meta-class, which does not posses other class or object in the system. Once we have finally transferred the data to the DW, then we must have metadata, including:

- DBMS system tables
- Partition settings
- Indexes
- Disk striping specifications
- Processing hints
- DBMS-level security privileges and grants
- View definitions
- Stored procedures and SQL administrative scripts

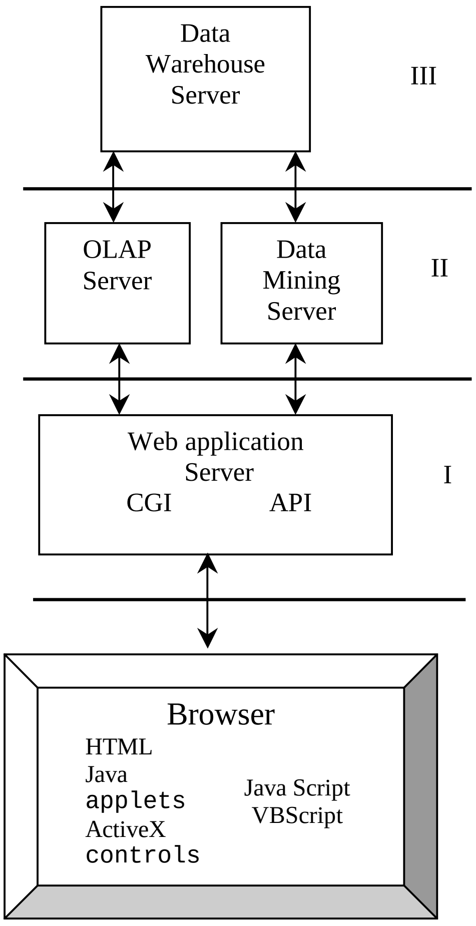

Fig.1. Approach to access uses Web Application server in 3-Tier architecture.

The data in these dimensions, called measurable, are in most cases aggregated. Detailed ‘raw’ data are used for the analysis needed by the marketing specialists. Because of this both non-aggregated data and different summarized level data (like that used for OLAP analysis) are stored in the repository of the DW.

The information system has a user-friendly interface, because the users examine the DW data using browsers. This way most of the data is very easy integrated and provided in any place in the Internet. By the same token the moving in the Web ensures independent platform mechanism for remote users, group application and consumer services. In this case the server’s task is to build the processes that on one side use HTML and HTTP and on other side communicate with OLAP applications and DB server. The Web server controls the communications between browsers and applications or the DB server. In this process the following server options are used: CGI script, API Web server, API application and API DW. On the client side – combination of HTML, Java applets, ActiveX controls, Java Scripts and VBScripts and interface (Fig.1).

OLAP is working in this mix through its installation in the net as an application server. The application server can be reached by any client application, including that spread by web browser. By this means the developer can use the ‘best fitting’ technology for every part of the program of the widespread Web-based applications.

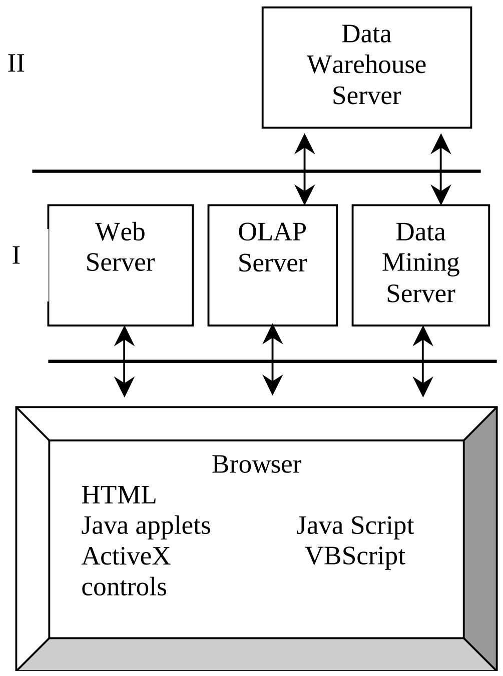

Fig.2. Approach to communicate directly to the application server without the use of the web server in 2-Tier architecture.

From the browser (Fig.2) the application components communicate directly to the application server without the use of the web server. Protocols like Distributed Component Object Model (DCOM, formerly Network OLE), Distributed System Object Model (DSOM), or Common Object Request Broker Architecture (CORBA) are used for finishing the communication. In such a way the productivity is increased and better functionality and security are ensured.

The necessity for providing information through the browsers is the basic philosophy of OLAP and Web application. We can conclude that this process requires HTTP access to the MDB server and the Data Mining server. The possibilities of data analysis are very important for the users of IMIS. Different levels of the OLAP reports, ranging over a large number of statistic’s reports with dynamic dimensions and drilling down the reports through the data.

The interface provided by the standard HTML presents plain text and stability features. The user must make several selections of two or three dimensions that are represented data. Only after that the user can watch the rotation after a selection.

Some product and time position changes can be made after which the user must wait for the updated report.

The approaches for granting access to the OLAP function in the Web can be presented in three categories:

- Static HTML reports, made in packet mode;
- HTML data patterns on-the-fly;

Java and ActiveX components, supported by the browsers.

The first approach uses preliminary lists for extracting OLAP data in packet mode, making static HTML reports using HTML templates. The templates are prepared in such a way that they can be used for all reports, no matter of their content. Then the reports are being controlled and sent to the browsers by Web server. The static reports are very convenient, fast and easy to transport to the browser. The disadvantage of the static reports is that they are not allowed to interact with DW data during the process. Some static reports simulate research of dimensions by navigation through the reports. In the browser list field a number of a report could be assigned that links it to the data using a hypertext.

In the 'on-the-fly approach' templates of the reports and the metadata are created in the browser. These metadata show the browser, which data are to be loaded in a HTML file, before sending them to the browser. There are two formats of metadata on the server – in the application tags and in the HTML templates. An alternative is storing them in the DB field in binary format. There is a possibility of another format for storing the metadata. Using its own HTML tags, the report presents the metadata already existing in the HTML file.

HTML also exists in two formats on the server. If off-the-shelf educator is used for creating templates they exist as HTML files on the server. By single user click on the name the report, both the HTML template and the data from the report are joined and sent as HTML reply to browser.

Web toolkits can also be used for making and storing metadata models in a binary format. In this case both the reports and the templates can be made using on-the-fly Web server processes. Web maintenance toolkit allows designing and formatting the report's pages. The obtained information is stored in a Web DB. Using this information it makes HTML templates, joins the data in a single report and presents it in the browser.

Regardless of the way the templates and reports are stored on the Web server, the server obtains the information, which is based on the code of the report sent by the browser. This information is merged with the template and all packets are sent to the browser as a HTML response. The default report offers several levels of OLAP functions. If the user is in dialog with an interactive report mode, user's code is sent with the rest of the information to the Web server. The application server starts a program that extracts and sends data to the browser. The alternative is if the code exists as values in the hidden HTML control of HTML file. Using this kind of project the program on the Web server identifies the user by reading the value of the user code as a control. Using this code and other information from the browser's session, the server downloads and formats the necessary MDB information and directs it towards the browsers.

Depending on the way the web server's applications are built the project can be used for multiple computer platforms. The CGI specification is mobile, but most servers don't use API.

The Java and ActiveX approaches use Java applets and ActiveX controls in order to minimize the communication between the browser and the web server and to improve the interface. There are two methods for their usage. In the first, the web server uses binary file with data from the reports and interface's controls are sent to the browser with an HTML files together. Using the client, the characteristics and the requirements of the control show the name of the corresponding data file to the browser. In this project the components provide functionality using deepening the data and changing the common format with "tricky" interface – with no need of communication with the server. Also, since the data are transferred in binary format between the client and the server, this method is more secure.

In the second method the interface's components communicate directly with the browser, transferring data to the user as result of a search. The interface components download the results from the server using HTTP stream. When the components present the result to the server, it makes selection and moves the data towards the object for visualization. This method has advantages similar to the ones of the first method. Nonetheless, in order to achieve the same level of security as for the first method HTTP protocols should be used. This method has its disadvantages as well - the JAVA work execution is faster, but it is not as easy transferred as HTML.

Generation of reports can be applied, based on browsers, presenting WEB based applications; mixing other web protocols with HTTP for direct communication with the same server process, used for the usual client/server applications. These network transport protocols can be used with HTTP for improvement in the communication characteristics. These reports include DSOM and DCOM, both based on CORBA. This design can improve the customer's experience, when browser is used and the abilities for growth, in order to make the systems usable in multiple environments.

V. CONCLUSIONS

In the development of the Web based marketing information system several conclusions can be drawn:

The applications directed to information adjust better to the web and obtain more advantages from their architecture.

The information systems that present the information use multiple formats and user friendly.

Using a browser for adjustment to the environment, the applications increase with a little bit their intensity, but this way difficulty for the users and the developers are created. The users need to administer additional software on the web server and the users obtain fewer possibilities from the interface if HTML is used.

REFERENCES:

1. Inmon W.H., Building The Data Warehouse (Second Edition). - NY, NY: John Wiley, 1993

2. Kimball, Ralph, "The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses", John Wiley & Sons, 1996

3. Архипенков С.Я., Завьялов Б.П., Шеховцов А.И. Object-oriented approach in the task of modeling complicated systems. Сб. Вопросы кибернетики, серия "Моделирование сложных систем и виртуальная реальность", Москва, 1997

4. Асадуллаев, С., Architectural corporative Data Warehouse, PC Week/RE'98

5. Буч, Гради, Object-oriented analysis and design, Second Edition, Пер.с англ. под ред. И.Романовского и Ф.Андреева, Издателтьство Бином, 1998

6. Вавилов К., С.Щербина, Web-integration of the corporative systems, http://www.profi-club.kiev.ua/, 2001