# Development of a Complex Web-Based Advertising System

Krešimir Fertalj, Tomo Helman, Vedran Mornar

Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia

This paper describes the architecture and development of a web-based system for producing, gathering and exchange of multimedia advertisements. The advertisements are classified and stored into a database on the server. The information about an advertisement includes free text and a set of values for predefined attributes. Optionally, any other computer file can be attached. A web site has been established and standard browsers can be used to manipulate the data online. A set of client programs has been developed to administer the data offline and periodically exchange data with the server. Several utility programs have been developed to provide the system for additional features. The main problems experienced during the development process are discussed. Possible solutions to these problems are proposed.

*Keywords:* advertising system, web programming, overdevelopment, system failure

## 1. Introduction

Several years ago the authors have developed a mailbox system for electronic interchange of advertisements ([2], [3], [5]). Offers and demands were prepared offline on a PC in user-friendly MS-Windows environment to reduce the telecommunication costs. A modem connection to a remote mailbox server was used to exchange the data with the server. In further development a whole family of products has been developed as an extension of the initial project. Although there were thousands of occasional users connected to the system, the system was never put into the real operations, due to the constant development and improvement. As the Internet was rapidly growing, the system had become obsolete and the project was cancelled.

Two years ago another life cycle was started. Almost the same developers were engaged to build a new system by using the old experience and modern technology. The new system is founded on Internet technology, a relational database management system (DBMS) and standard data transfer protocols. Up to now the project has passed through several phases. During the first half of 1998, an extensive rapid prototyping took place and functional prototypes were built soon. Later, the prototypes were fully developed, tested and debugged. By the end of the year, the system was nearly ready for deliverance. Unfortunately, the investor was concerned about the features of some competitive systems appearing on the market at that time. Thus, the investor required more new features. During the first half of 1999, the system was enhanced by additional routines. Until now, constant development turned into adaptive maintenance of the system which has not been put into the real operations mode yet.

## 2. The concept of the system

The advertisements are classified in two hierarchical levels (*Topic* and *Subtopic*), as presented by the simplified data model in Fig. 1. To classify the information further on, there are some predefined attributes: check boxes (*AttributeCheck*), combo boxes (*AttributeList, ListElement*) and numerical attributes (*AttributeNum*). The basic ad (*Advertisement*) information consists of a free text. Optionally the values for predefined attributes are stored into the respective tables (*AdCheck, AdList* and *AdNum*). Pictures can be attached, usually as a pair of

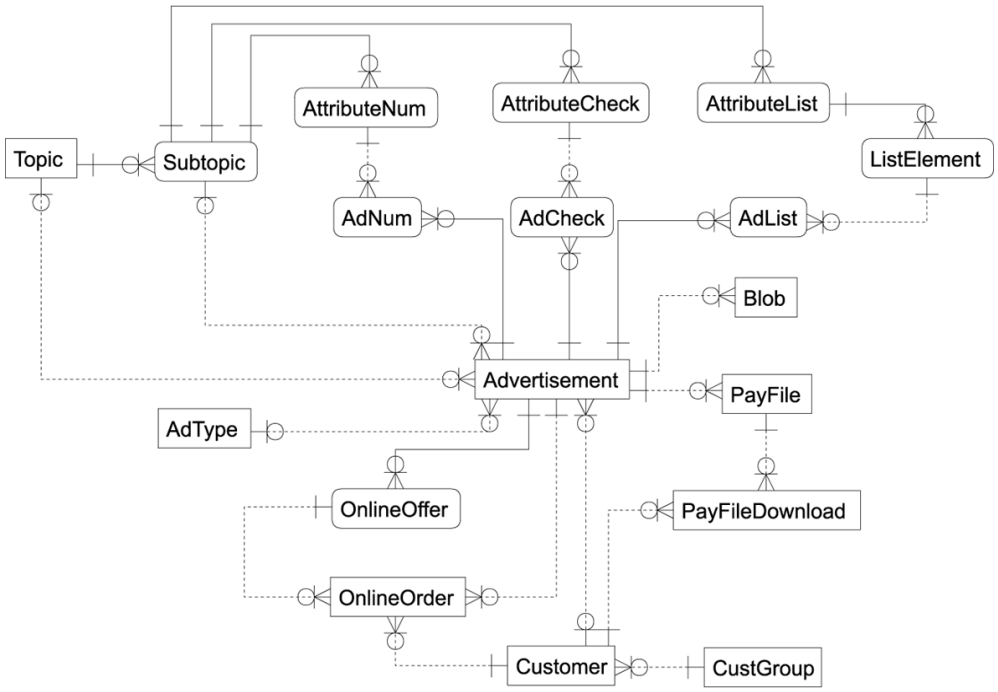


*Fig. 1.* The simplified data model.

a full picture and a thumbnail (*Blob*). Sound, animation or any other computer files can be attached separately (*PayFile*), and can be downloaded (*PayFileDownload*) for some fee.

The system stores the ads of different types (*AdType*), such as private and business ads. Additional ad types were defined for site advertising purposes (banners, content-dependent ads, etc.). Furthermore, registered users can submit information requests (search criteria), which are stored as ads of a special type. Based on the request, the user can get an e-mail notification when the ads, which satisfy the given criteria, get into the system.

The system supports online preparation of offers and online ordering based on those offers. Basic information about an offer is stored as an ad. The additional data about an offered product is stored separately (*OnlineOffer*), and comprises the pricing, payment methods and delivery conditions. A user wishing to order an offered product, he/she is prompted to enter the information about desired quantity, payment and shipping options, and the total price is calculated. The order is stored into the database (*OnlineOrder*), and the user who created the offer automatically gets an e-mail notification about the order.

The users (*Customer*) are categorized as guests, registered users, agencies, agency clients and administrators (*CustGroup*). Registration fees and privileges are defined at the category level. For example, guests pay no fee and can only search for ads and submit private ads. Before becoming active their ads need to be approved by an agency. The registered users, who supply their bank account upon registration, pay a registration fee. They have the right to update their own ads, submit business ads and download the *PayFiles*. Other registered users do not pay for registration and their ads should be approved before activation. Such users can only update their own ads. For some additional fee, the agencies can approve ads and submit ads for their clients. Agency clients do not have the right to log on the web site. The administrators have full access to the system.

## 3. System organization and components

Main components of the system are shown in Fig. 2.

### 3.1. Web site

Briefly, the web site supports the registration of users, preparation of ads, browsing by topics and subtopics, simple search, advanced search,

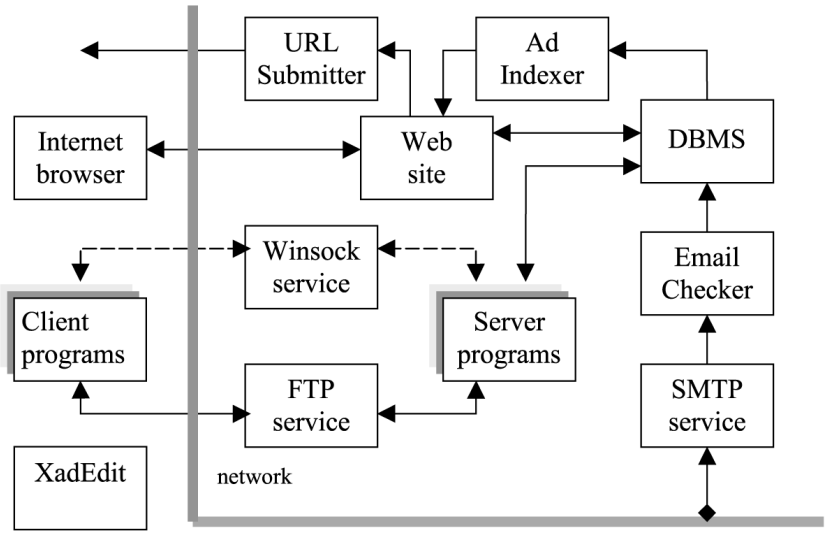


*Fig. 2.* Organization of the system.

and batch import of ads from formatted files. The user registration and other secure data (e. g. credit card details) are handled by using secure web forms. A separate part of the site supports some administrative tasks, such as handling of the ad categories and attribute definitions, management of ad and customer data, and monitoring of system statistics.

## 3.2. Client programs and server programs

A set of client programs supports the remote data management.

*CategoryManager:* This client program handles the ad categories and attribute definitions and supports translation of phrases used by multilingual features.

*CDProduction:* The original purpose of the program was to produce electronic catalogues based of the database contents to be stored on CD-ROM disks and to reproduce multimedia information. Now this program is also used to update the status of ads (pending for approval, active, cancelled, expired). In addition, the user of the program can re-assign the ads to alternative categories.

*AdManager:* The program is used to prepare the ads locally and send them to the server. The user can decide whether he/she wants to transfer only his/her own data or he/she wants to receive other users' data from the server.

*MoneySharing:* This program performs automatic invoicing and simplified accounting, based on financial transactions (registrations, commercial ad submissions, *PayFile* downloads etc.).

Client programs enable the users to work offline and exchange the data with the server on demand. A proprietary data replication mechanism has been developed to replicate the data between the client database and the central database. The mechanism is capable to exchange high volumes of data over a slow network or over Dial-Up networking. Each client program communicates with the respective server program via our own service (*Winsock service*), while the compressed data is transferred by FTP (*File Transfer Protocol*). Simultaneously with the data transfer, a potential upgrade of the client program and the client database structure is automatically downloaded from the server.

## EmailChecker

Several services exist on the Internet, which collect ads from various sources and send them as formatted e-mail messages to the subscribed e-mail address. A program called *EmailChecker* runs on the server, periodically checking for messages, which come from the subscribed services to SMTP (Simple Mail Transfer Protocol) service on the server. Incoming messages are parsed and extracted information is stored into database. *EmailChecker* proved to

be one of the most useful server components, bringing in approximately 200 new ads a day. The program supports a number of e-mail addresses, which real users may use to submit the ads by e-mail. For example, the ads formatted as e-mail messages sent to the address `offer.cars@server.com` are imported to the database classified in the topic *Cars*.

### XadEdit

One of the functions supported by the web site is batch import of ads from formatted files. Transfer files can be prepared offline, and then sent to the web site via form-based file upload protocol (RFC 1867). At first, a very simple format was used (one ad per line, field values delimited by "|"). As the data structures grew in complexity, the format became inadequate, and a new format had to be chosen. Extensible markup language (XML) seemed to be the right choice for the new format hence it is a standard defined for describing structured data. *XadEdit* was designed to prepare XML files, which can be sent to the server for import.

### AdIndexer

When Internet is searched for some information by using a search engine (AltaVista, Yahoo etc.), only the information that resides within the web pages can be found. That means that search engines cannot detect the data stored in the database. To present the ads to the search engines each ad would need a HTML page with a unique URL. Thus, *AdIndexer* creates HTML files named as `http://server.com/ Topic/ Subtopic/AdIdentifier.html`. *AdIndexer* is scheduled to run periodically and, on each run, it builds a completely new index of HTML files. For expired ads, the pages are deleted from the web site.

### URLSubmitter

New HTML files are not immediately available to search engines. Every search engine has a different policy regarding the submission of web pages. Most engines periodically revisit subscribed web sites and update their catalogue with new links, but it may take a few months between two subsequent visits. *URLSubmitter* automates the task of submitting URLs to search engines, which can be scheduled to run in the background.

## 4. Some development problems

The project was contracted by one German investor and a team of developers at the University of Zagreb, Croatia. In two years, many sophisticated functions were developed, but the purpose and usability of the system as a whole are increasingly doubtful. In spite of relatively high development and maintenance costs the system is still expanding. Let us point at some development-related problems and doubtful decisions related to the project.

### 4.1. Lack of precise contract and strategy

The project was started without an initial investigation or a feasibility study of the scope, objectives and recommendations of the system that was to be developed. The project timetable, definition of software requirements and criteria of acceptance were never contracted. Instead, the project was driven by a sequence of frequently modified requirements made by the investor and his representatives. When the project was started, similar web sites already existed, especially in the United States. Their advantages and weaknesses were not evaluated to define the objectives. Instead, the investor used them as milestones for the features that he required.

### 4.2. Permanent software extension

The lack of strategy caused dispersion of the development over many functions, which were sometimes developed without particular priority. In other words, the requirements had to be served as they entered the priority list. Of course, some ideas that had already been developed were later abandoned and some functions became neglected during the time. Permanent software extension significantly slowed the development and, as expected, decreased the effectiveness of the system ([4]).

### 4.3. Overkill of features

Since the beginning, the investor insisted on various features to be implemented into the system, sometimes forcing the evasion of standards. For instance, many frames were used on web pages, which complicated the navigation and made the access to certain web pages difficult. Some parts of the site lacked a professional layout design and user-friendliness, which definitely can discourage the users from revisiting the site. Implementation of multilingual features was forced, although it appeared that there was no real need for them, or at least that it was too early for them. At the same time, multilingual features were implemented selectively. To conclude, overkill of features caused higher complexity of the system and unnecessarily consumed additional resources. Consequently, the development costs increased.

### 4.4. Replication of functions

A significant effort was made to develop client programs supporting some of the functions implemented on the web site and vice versa. For example, the management of ad categories and attribute definitions, and translation of phrases used by multilingual features are supported by both the web site and *CategoryManager*. Occasionally, there was a discrepancy between the functions available online and offline because some of the components were developed faster. Some of the functions were replicated among different client programs, but this was over-bridged by reusable software components. However, the development of client programs imposed the need for frequent redesign of client databases and central database, which had to be synchronized in structure.

### 4.5. Development of "surrogates"

Some system components were re-invented and developed from scratch, although purchasing of similar products would probably be cheaper. For example, the investor required *URLSubmitter*, though several companies had offered submission of URLs to more than a hundred search engines for a reasonable price.

### 4.6. Lack of real users and lack of real data

Although the system was established on the Internet almost at the beginning of development, only the developers, the investor and some occasional users currently use it. Until now, roughly 30 real users have been using the system. The lack of real users resulted in the lack of real data, which we consider to be of highest importance for commercial success ([5]). For example, there are roughly 15,000 active ads and 75,000 of expired ads currently in the database. The *EmailChecker* collected 97% of these ads and the rest are stored by real users.

### 4.7. Offline development and maintenance

The requests for development are passed to developers by e-mail, which imposed some barriers in human communication. The production server is physically located at one Internet service provider in Germany, and the development center is placed in Croatia. Therefore, the production server is maintained remotely. Some diagnostic functions had to be incorporated into the client programs and into administration part of the web site. Finally, the lack of real users resulted in the lack of a good feedback from users. Due to all aforementioned problems, at least 40 percent of development efforts were consumed on testing, debugging and maintenance.

### 4.8. Solutions to the problems and proposals to the investor

Several measures were taken by the developers to ensure better project planning and to avoid or at least reduce the problems described above. Periodical reports are made to keep the investor informed about status of the project. Lists of requests are produced to facilitate the collection of requests and the refinement of ideas. The Email correspondence is traced in order to make it more efficient. A test sheet was created to be filled out by the pilot users in order to ease the debugging process. Instructions for maintenance performed by the investor's staff were created.

On several occasions, the investor was told that only pilot users could perform good system testing. The investor was advised to give the client

software and special privileges to several agencies, which would use the system free of charge for some time. By this, the agencies would provide the real data. In turn, the system would be filled with real data and developers would get the feedback of real users. The existence of real data would further attract additional users to the system. The project leader also advised the investor to engage a person who would permanently maintain the server and test the software. Periodically, the project leader proposed to the investor to stop the development and to put the system into real use. Unfortunately, the appropriate feedback was missing, or there was no feedback at all.

## 5. Technical considerations

The server runs under Microsoft Windows NT. On the server side, Microsoft SQL Server manages the data. Internet Information Server 4.0 was selected as a platform for the web site hence it supports dynamically created web pages by using Active Server Pages technology (ASP). ASP provides a powerful scripting language (VBScript or JavaScript), and some built-in objects that can connect to the database by using the ODBC data source. Besides the ASP, the site comprises some proprietary objects used to handle the database. These objects, server programs and utilities were developed in Visual Basic. The secured web pages rely on HTTPS (HyperText Transfer Protocol Secure).

For the client programs, Microsoft JET Engine was chosen to store the data locally. The client programs were written in Visual Basic, enhanced by some ActiveX controls ([8]). Some parts of the software, for instance the file compression routines ([6]) were written in *C* language.

## 6. Conclusion

In this paper, we have presented a complex web-based advertising system. During two years of development a number of system components were created and many features were implemented. Recently we have started to worry about the success of the project. Having in mind our bad experience with the old mailbox system, we did our best to avoid the well-known causes of system failure ([1], [4], [7]). Nevertheless, we are still concerned about the possible failure.

If the development was continued, as before, it would never be finished, due to the increasingly expensive development with no return of investment. If the development were stopped today, it would probably take months before the system becomes thoroughly tested and ready for the full public access. In latter cases, the system, as is, would probably fail to gain acceptance by clients, and it would be a commercial disaster. A possible way to overcome possible failure is to stop development temporarily, to analyze the current situation, and to redefine the goals and objectives. The components developed and experience collected so far should be used to rebuild the system and make it more simple and effective, thus more usable.

## References

[1] P. BEYNON-DAVIES Information systems 'failure': the case of the London Ambulance Service's Computer Aided Despatch project. *European Journal of Information Systems*, **4(3)** (1995), 171-184.

[2] G. DITTEL, V. MORNAR, K. FERTALJ, D. KALPIĆ, Computer Technology 1995 — Decline or Revival of the Graphic Arts Industry. Presented at the *Proceedings of the* $12^{th}$ *Conference Intergrafika*, (1993) Zagreb, Croatia, 254-258.

[3] G. DITTEL 69! Data Highways — Präsentationsmappe für Kooperationspartner (in German). Günter Dittel Software Entwicklung, Erlenbach (Main), Germany, 1994.

[4] J. J. KAASBØLL How evolution of information systems may fail: many improvements adding up to negative effects. *European Journal of Information Systems*, **6(3)** (1997), 172-180.

[5] V. MORNAR, K. FERTALJ, D. KALPIĆ Multimedia Production and Distribution System. Presented at the *Proceedings of the* $17^{th}$ *Int. Conference on Information Technology Interfaces*, (1995) Pula, Croatia, 501-506.

[6] M. NELSON, J. L. GAILLY, *The Data Compression Book*. M & T Books, New York, 1996.

[7] A. POULYMENAKOU, A. HOLMES, A contingency framework for the investigation of information systems failure. *European Journal of Information Systems*, **5(1)** (1996), 34-46.

[8] A. WILLIAMS Visual Basic 5 and ActiveX Controls. *Dr. Dobb's Journal of Software Tools*, **22(3)** (1997), 74.

*Received:* October, 2000
*Accepted:* November, 2000

*Contact address:*
Krešimir Fertalj, Tomo Helman, Vedran Mornar
Faculty of Electrical Engineering and Computing,
University of Zagreb
Unska 3
10000 Zagreb
Croatia
e-mail: kresimir.fertalj@fer.hr,
tomo.helman@fer.hr,
vedran.mornar@fer.hr

KREŠIMIR FERTALJ is an Assistant Professor of Computer Science at the Faculty of Electrical Engineering and Computing at the University of Zagreb, Croatia. He earned a PhD in Computer Science from the same university, where he currently teaches several computing courses. His main research interests include software engineering, CASE tools, database design and applications development. Fertalj was an information systems developer and consultant in a number of real life projects. He has also developed his own sorce code generator independent of the target fourth generation language.

TOMO HELMAN received his BSc from the Faculty of Electrical Engineering and Computing at the University of Zagreb, Croatia. Currently he is a researcher at the Department of Applied Mathematics (chair Computer Science). His research interests lie in software development, including Web programming and administration and database implementation.

VEDRAN MORNAR is an Associate Professor of Computer Science at Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. He received the PhD degree in Computer Science from the same university, where he currently teaches several graduate and undergraduate computing courses. His academic interest is in operational research and database design and implementation in real world information systems.