Developing Web Site as Database Mapping

Gleb Pogodayev[1]
Institute for Contemporary Education "JurInfoR-MSU"
Vorotnikovsky per., 7
Moscow, 103006 Russian Federation
gleb@msu.jurinfor.ru

Abstract

The information to be published on the Web can be stored directly as fixed HTML documents or be formed depending on user's request. In both cases Web documents can be created on the basis of the information from databases. The interaction between a Web browser and DBMS takes place within the framework of a three-level client/server architecture including a Web-server and a server extension as a middleware[13]. The intermediate level provides analysis of client requests, access to databases, representation of the selected information as a HTML document and its delivery to the browser. All this is carried out by means of a close interaction between the Web-server and the server extension, which is defined by certain specifications. This paper contains a brief comparative review of the most well-known specifications CGI and ISAPI and examine the main issues and peculiarities on the example of a complex site creation project for an editing company in which author took direct part.

1. Introduction

Nowadays quite a lot of people have access to Internet. It is perfect possibility for corporations and persons to tell about themselves by creating their own Web sites. Lately the computer technologies have already got their own Web sites. Web site becomes as usual as a telephone number or e-mail address.

[1]Also: Cybernetics Department, Moscow Engineering Physics Institute, Moscow, 115409 Russian Federation, gleb@jmsuice.mephi.ru

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CSIT copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Institute for Contemporary Education JMSUICE. To copy otherwise, or to republish, requires a fee and/or special permission from the JMSUICE.

Proceedings of the Workshop on Computer Science and Information Technologies CSIT'99
Moscow, Russia, 1999

Russian sector of the Internet has been considerably extended. Almost all companies dealing with It can perform a wide spectrum of functions: from giving brief advertising information to working with the overall information system having a considerable size of data and advanced tools of search. To create such system it's necessary to apply special technical and programming solutions beyond the ordinary Web design. The advantages of this system as compared to the traditional Web site based on the collection of static HTML documents are significant:

- accept to working with maximally full and actual information
- its effective structural data representation
- simple update of the information by using a database client application
- finding data likely to satisfy a given query
- a flexible adjustment of data representation
- the possibility to extend the server functionality
- standard methods of preserving and retrieval of multimedia objects out of the database

That's why complex Web sites integrating Web technology and DBMS become more and more popular. Currently many companies developing a software offer their own solutions to create such sites[4,13]. If you want to design your own middleware to interoperate between the Web browser and DBMS, you are sure to encounter the problems described below.

2. The definition of project aims

It's most important to precisely define the assignment of the developed software. Whether it is intended for electronic commerce? For support of the users? For distribution of the information inside the corporation? The common task knowledge defines the criteria to estimate functional possibilities of the product and substantially determines the complex of the required technical and software solutions.

Some time ago I've taken part in a project aimed at designing the Web site for an educational institution. The requirements were to supply representation of the information about com-

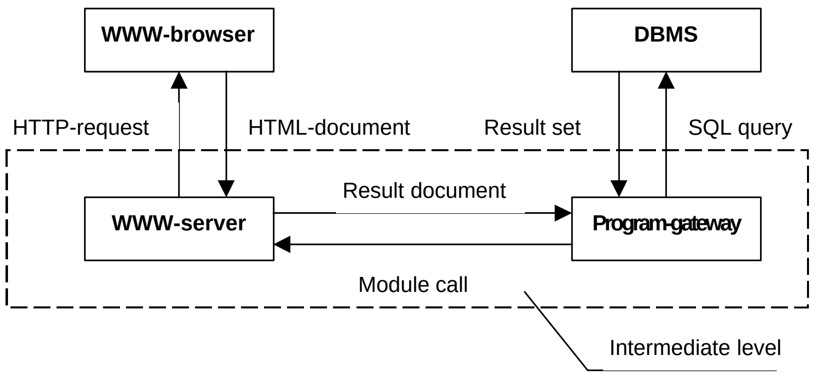


Figure 1. The third-level architecture of WWW application working with databases

pany workflows in its entirety, convenience and efficiency of data update, and possibility to search data on client's queries. Based on these requirements it was decided to employ databases for storing the information displayed on the Web site. The task caused the necessity to develop a special program-gateway providing the interaction between the Web-server and DBMS.

3. The software development tools

It is difficult enough to make competent and justified choice of the tools of development, especially because it seems easier to employ already families tools than to master a new though, perhaps, more suitable. In our project Delphi 3.0 was selected as the tool of development using Borland Database Engine (BDE) for access to Paradox databases. Delphi 3.0 has advanced possibilities to develop the Web-server applications. The choice of version 3 was not casual, because from just this version Delphi offers advanced possibilities for the development of Web server applications (the previous versions did not provide for special means to develop such programms). Delphi 3.0 supports the creation of three sorts of the Web-applications: CGI, WinCGI, and ISAPI/NSAPI. The developer can have only the general idea of the features of each of them, because the new project Delphi makes an automatic adjustment to the selected Web application. It allows to save time and fully concentrate on the development of the logic of processing the client queries.

4. The integration degree of a Web browser and DBMS

It's important to decide whether the Web-browser will be used for changing the database information or only carrying out requests for read and search of the data. Despite evident merits of the Web, such as the standardization of the user interface and client/server relations , the possibility of shared operation of the different applications from different platforms, it has also got a number of drawbacks as compared to the client part of DBMS. The Web-browser has a weak control of input and it doesn't handle transactions. Web is more useful for making requests to databases then for modifying its information. That's why a mixed approach based on both Web-browser and a client program of the database was applied in our project.

5. The project architecture

The WWW applications working with databases, combine some two-level systems to new its type based on the three-level client/server architecture. The client level is occupied by a Web browser, the server level is a database server , and the Web-server and server extension settle down on the intermediate level. Figure 1 shows this architecture.

Some comments to the scheme are given below. A Web browser formulates client request and sends it to the server. It can be formed by filling form or addressing to URL. The HTML-construction calling the server extension looks like the following: <form method={GET;POST} action= path and program name >

1) Receiving a request, the server determines the type of the required information resource and if it's a call to a program - gateway, loads it. The scheme of the interaction between the Web server and the program-gateway depends on the used specification (ISAPI, CGI)
2) The program - gateway forms SQL-query to the database on the basis of the transferred parameters
3) DBMS returns query results
4) Program-gateway forms the HTML document, using query results and data representation adjustments and delivers it to the Web server.
5) The server transfers the generated HTML-document to a browser

6. The interfaces between Web server and server extension

Currently there are many specifications defining interaction between the program-extension and the Web-server. CGI(Common Gateway Interface) is the oldest and the most well-known of it. CGI have a number of drawbacks, that newer specifications such as ISAPI and NSAPI don't have. A brief description of CGI and ISAPI and also their comparative characteristic is given farther.

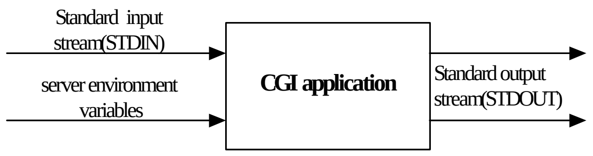

Standard input stream(STDIN)
server environment variables
CGI application
Standard output stream(STDOUT)

Figure 2. The scheme of CGI program

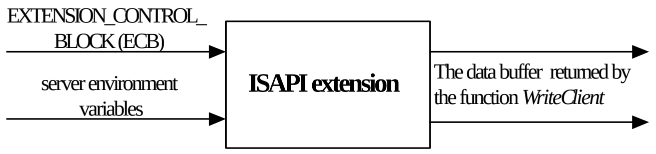

EXTENSION_CONTROL_BLOCK (ECB)
server environment variables
ISAPI extension
The data buffer returned by the function *WriteClient*

Figure 3. The scheme of ISAPI extansion

6.1. CGI (Common Gateway Interface)

6.1.1. Input data

There are two methods for sending form data by the Web server to CGI program. These are GET and POST.

- When a form uses the GET method to pass information to the extension, it is called like this: <http://www.test.com/cgi.exe?param1=value1&param2=value2. In that case CGI application gets the entry parameters by reading the server variable QUERY_STRING.

- If data is sent using the POST request method, than program reads the information from the standard input stream. To define the data size it uses the server variable CONTEXT_LENGTH

It should be noted that form data is encrypted for passing to CGI application. This encryption replaces blanks and other symbols to be invalid for URL on their hexadecimal equivalents (Figure 2).

6.1.2. Output data

CGI-program outputs the information in a standard output stream. It can return a ready HTML document or a reference to the document. Any lines not being directives of the server is transmitted directly to the client. CGI specification defines three directives of the server*:* *Content-type* (type of the returned document), *Location* (specifies that CGI response is the link to the document), *Status* (for example**,** HTTP/1.0 200 OK). For more information about CGI, see [1,2,3,4,5,8,10,11,13].

6.2. ISAPI(Internet server application program interface)

Input data

As ISAPI extension is executed within the context of the server process, it has access to the area of the server data. To interact with ISAPI-programs the Web server uses special structure ECB (EXTENSION_CONTROL_BLOCK). The execution of ISAPI program (Figure 3) begins by calling standard server function HttpExtensionProc. This function has the single parameter which is pointer to ECB structure.

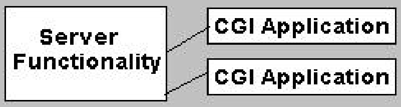
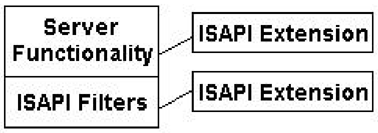
- When a form uses the GET method to pass information to the extension, it is called like this: http://www.test.com/extension.dll?param1=value1&param2=value2. When the extension is executed, it reads the information either by reading the server variable QUERY_STRING or by looking at pECB->lpszQueryString. Working with pECB->lpszQueryString is the preferred way of reading the data because you don't have to call any functions to receive the data.

Another method to send the form information is POST method. The first 48 kbytes of data from the POST request are pointed to by the EXTENSION_CONTROL_BLOCK pointer lpbData, with the total amount of data available shown by cbTotalBytes.

Output data

The two ways of writing to the client from the extension are using WriteClient and ServerSupportFunction. ServerSupportFunction has limitations in that it can only send ASCII data to the client. It is used for sending completion status information and other header details to the client. WriteClient just sends the stated number of bytes to the client so it can send both ASCII and binary data, but it can also be used for sending completion status information back to the client. For more information about ISAPI, see [1,2,3,4,5,8,12,13].

Table 1. CGI and ISAPI specifications

Categories	CGI	ISAPI
1. Performance	CGI works by creating a new process for each CGI request. The Web server responds to a CGI request by creating a new process, filling that process' environment with HTTP request variables, and starting the CGI application.	The ISAPI server simply creates a thread pool when the server is initialized. A free thread from this thread pool serves the incoming connection. If the threads in the thread pool are all in use, the server can create additional threads to handle the waiting connections. Thread creation is much faster than process creation..
2. Scalability	The memory needs of concurrent processes can rise quick.	ISAPI allows to build Web sites that scale up from one connection to hundreds of concurrent connections per second without massive additional resources such as server memory, because creating a thread takes much less memory than creating a new process.
3. Extensibility	 CGI is limited to extend the server.	 Server Functionality ISAPI Filters ISAPI Extension ISAPI Extension ISAPI allows a developer to extend the server, adding functionality at a more basic level than CGI allows.
4. Language independence	CGI applications can be written by any programming or command languages having the tools of operation with strings	The ISAPI extensions can be written only by languages supported API; the command languages (for example, Perl) are not used in the delivered ISAPI-units
5. Multithreading support	CGI application processes only single client request and then ends	The important advantage of ISAPI compared with CGI is working with several HTTP requests by one server process
6. Isolation of processes	Each CGI program works as separate server process and an incorrect one can't influence on the Web-server operation or get access to the confidential information	The ISAPI applications execute within a context of the server process, therefore their incorrect operation can result in the Web-server failures.
7. Compatibility	The CGI specification is the open standard. It means that script written for one server will also work with other servers supporting this specification.	The program to be written in accordance with the defined API can be used only at this "native" server (for example, applications based on Internet server API (ISAPI), can be used only on MS IIS)
8. Architectural independence	CGI doesn't depend on features of implementation of the server architecture.	The API-applications are dependent on the server architecture. If it changes, ISAPI usually changes too.
9. Distributing	CGI program must be installed on the same computer as the Web-server.	ISAPI program must be installed on the same computer as the Web-server.

6.3. The comparative characteristic of the applications based on CGI and ISAPI specifications

Let's summarize some results concerning CGI and ISAPI (Table 1). On the one hand, the programs based on ISAPI process client's requests faster than CGI, support multi-threading and allow to expand server functionality(for example, to carry out authentication of the users; from another it raises the riskness of crashing the Web server, limits a choice of the server architecture and the tools of development.

In our project a main criterion of a choice was the speed and the performance of the transaction processing. That's why ISAPI technology was chosen to be uses in the project.

7. The representation adjustment of the information publishing on the Web

The representation forms of the Web-site information is not less important, than its contents.

We see the Internet environment to be filled by higher and higher quality graphics, animation and sound. It is obvious, that the automatic creation of Web-documents on the basis of the database information requires using either universal, or adjusted HTML-templates. In our project the second approach is implemented: special HTML-documents (templates) are stored in the database and the real Web-documents looked through by the users are formed on the basis of these templates and the available business-data. The operation with templates has some specificity connected with the diversity of their types and features of filling. The number of templates is defined by all possible combinations of three parameters: the language (Russian, English and others), the object (book, publishing house etc.), the type of the template. There are also only three types of the template.

1) The template of the object - intended for representing full information about the object

2) The template of the object header - intended for imaging a brief information about the object. This information helps to identify it at review of the list of the objects.

3) The template of the object type - used for creation of Web documents, for example, containing the information about all released books, including the header of the document, the list of these books (the representation of information about each unit in the list can be formed by template 2) and some accompanying information.

There are also two special templates: the template of the Web-site main page and the template of the page to search the information in the database. The templates are filled on the basis of the database information, that's why the special conventions together with standard HTML tags are used in these documents.

7.1 The conventions accepted in our system

1) The links (an appropriate tag is substituted by the link on the file or the label in the document)

1.1 < #REFTEXT type=V1 [example=V2] [info=V3] [lang=V4] > (link with the explaining text)

1.2 < #REF type=V1 [example=V2] [info=V3] [lang=V4] > (link only)

The option *type* defines the type of the object,

example identifies the object of type V1,

info specifies the information section of the object V2,

lang chooses the language (Russian, English etc.)

Note. The optional parameters are placed in square brackets; V1, V2, V3, V4 are values of the appropriate options.

Example: < #REFTEXT type=bk example=102 info=23 lang=ru >

The given link with the text refer to the text of information section 23 of book 102 in Russian.

1.3 < #REFFLD type=V1 example=V2 kind = {ref, adr, text} > It's a special link to the object of type V1 which is identified by the primary key V2. The option *kind* defines the sort of the link. There are three sorts of the links: ref (link with the text), adr (address), text (text).

Examples:

< #REFFLD type=in example=id_info kind=ref > the link with text to the main information section of the current object

< #REFFLD type=pb example=id_publisher kind=ref > the link with the text to the publisher of the current book.

2) Information blocks (a tag is replaced by some information from databases)

2.1 < #FIELD info=V1 kind = {text, ref} >

The option *info* defines the name of an information field for the current object, *kind* - a format of imaging the field. There are two formats of imaging: text (value of the field with the additional text), ref (only the value of the field).

Example: < #FIELD info=PAGES kind=ref >

This tag specifies the information field PAGES of the current object of the book type.

2.2 < #EMAILFORM action = {add, delete} > the form for depositing or deleting e-mail from a mailing list.

2.3 (#Note) - remarks for the current object

2.4 (#SEARCH) the link to the form to be used for seeking in the databases

2.5 < #SEARCH type=V1 > the link to the form of searching the object of type V1

2.6 < #LIST type=V1 > the list of objects of the type V1

2.7 < #LIST type=V1 type2=V2 fld=V3 > the list of type V1 objects connected with the current object of V2 type by the field V3.

Examples:

< #LIST type=au type2=bk fld=id_book > the list of the authors of the current book

< #LIST type=re type2=bk fld=id_book > the list of the reviewers of the current book

< #LIST type=in type2=bk fld=id_book > the list of information sections of the current book

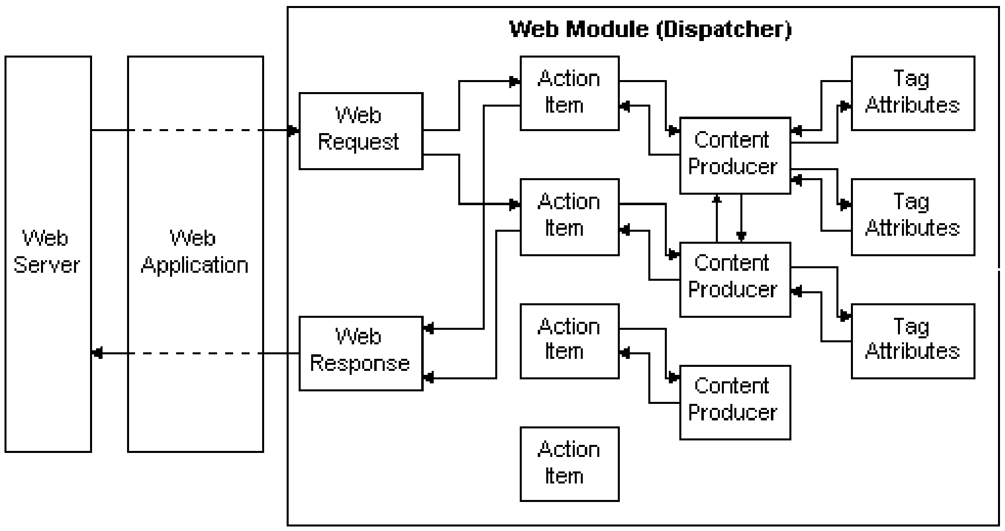


Figure 4.The scheme of the Web-server application in Delphi 3.0

8. Specifications of ISAPI development applications in the environment Delphi 3.0.

The development of ISAPI applications is more complex, than the development of CGI programs. It' caused by the necessity for the programming support of multithreading and for studying the internal server functions. However Delphi 3.0 allows to simplify the developing considerably. The developer may not have special knowledge of ISAPI because Delphi make automatic adjustment for ISAPI application. Consider the features of Delphi project of ISAPI application (Figure 4):

1) The header begins with the reserved word **library**, indicating that the project will be compiled as DLL.

2) After the description of used units the section **exports** follows, indicating what function will be exported. By default the export of three functions is required for ISAPI extension.(they are called directly by the Web server)

- The function *GetExtensionVersion* allows the server to get the extension version
- *HttpExtensionProc* returns the description string at initialization. The information is passed into it through the single parameter. It's a pointer to special structure, named EXTENSION_CONTROL_BLOCK (ECB).
- *TerminateExtension* stops the execution of ISAPI application.

3) There is a class, inherited from TWebModule. The entity of this class is the container for all components used in the project.

Now consider the common scheme of the Web-server application in Delphi. When the Web-application receives HTTP-request, it creates the object of the TWebRequest class representing the information of the request and the object of the TWebResponse class, that will contain the data returned to the Web-server. They are available in the Web dispatcher to manage the processing of HTTP-request.

2) The Web dispatcher has some action items to which it transfers the handling depending on parameters of HTTP-request.
3) The handlers can use special components (TPageProducer, TDataSetTableProducer) for dynamic creation of the HTML-code and analysis of the tags of the document being formed.
4) When all handlers complete the operation, the data is formed that returns to the Web server by means of filling the TWebResponse class object.
5) The application transmits the response to the client via the Web-server

Conclusions

If the organization wishes to have its own Web site, working with the database information, it has two alternatives. The first is the creation of Web site on its own (a brief description of this process is given in this paper on the example of a complex Web site, creating for an editing company). Another alternative is to use ready software solutions, offered by various companies developing software. Now many large manufacturers such as Oracle, Microsoft, CA, Informix, Borland support publishing information on the Web. It is carried out by means of integrating the possibilities of a database server and a Web server. So, the latest INFORMIX-online server variants provide the possibility not only of the access to the databases through the Web means, but also of the storage of HTML-documents in relational databases. No doubt, the appropriate support on the part of database servers will continue to increased.

Acknowledgement

This research was supported by Institute for Contemporary Education "JurInfoR-MSU" where the current project is under implementation with a programmers' team as the part of a general task. The computational ideas of the project streams are due to the principal project leader of the Institute L. Ismailova, who spent a lot of time and efforts to explicate and fix the main developing layouts and diagrams in conjuction with this particular research and the project in general. The author is indebtful to L. Bourmistrova for a durable collaboration at all the stages of the project. A special appreciation is to K. Zinchenko for his valuable advisory, engineering job and permanent readiness for fruitful discussions, without his assistance this research hardly ever would guided into the directions described in this paper. He applied a tremendous efforts to achieve a transparency and high modularity of all the principle solutions which were followed across the implementation stages.

References

1. Heller M. "Programming of the Web-server: a last boundary". *Networks and communication systems* 1996;8
2. Nance B. "Application program interfaces for the Web ". *Networks and communication systems* 1997;7
3. Parkherst "FASTCGI removes limitations of the popular Web interface ". *Network World* 1997; 10
4. Bulach E. "Tools of access to databases in Internet and DBMS POSTGRES95 ", *DBMS* 1997; 2
5. Khramtsov P. "Applications in the Web". *Cit Forum* (www.citforum.ru)
6. Artemyev V. "Intranet - myths and reality ". Open systems 1997; 4
7. Artemyev V. ""Intranet". *Cit Forum* (www.citforum.ru)
8. Kuznetsov C. " Internet and database ". *Cit Forum* (www.citforum.ru)
9. Kuznetsov C. " Access to databases with use of technology WWW ". *Cit Forum* (www.citforum.ru)
10. Nuzhnin C. " Use CGI for creation of interactive interfaces ". *Cit Forum* (www.citforum.ru)
11. Maximov K. " CGI - Common Gateway Interface ", *Cit Forum* (www.citforum.ru)
12. " Using ISAPI. Special Edition. ", QUE Corporation, 1996
13. Lang K., Chow J. "Database publishing on the Web&Intranets", Coriolis Group Book,Canada,1996