WAY: A user adapted access to information

Camino Fernández Paloma Díaz Ignacio Aedo
Computer Science Department. Universidad Carlos III de Madrid
Av. Universidad 30, E-28911 Leganés (Spain)

ABSTRACT
WAY is a model of architecture devoted to the construction of adaptive information systems. Its main goal is to make the information searching process easier, offering systems that adapt their interface to the user and save user time performing parallel searches. These tasks are developed by intelligent agents capable of taking decisions and propose new searchings and changes in the interface.
WAY-Z39.50 is an implementation of the WAY general model. This particular system accesses servers located at digital libraries which funds are available through the Z39.50 search and retrieval protocol. Its agents’ intelligence and, therefore, its decision making process is based on fuzzy reasoning systems. This paper is focused in the process of interface adaptation to the user through the detailed presentation of a session where some aspects of the whole process are discussed paying special attention to the ones related with adaptation.

Keywords: Adaptive Interface, Information Access, Agents, Z39.50.

1. INTRODUCTION
Bibliographical searching is improved when the user can access multiple information servers. Thus, if the search does not succeed in one of the servers, the user can repeat the query in another one. It would be very helpful if alternative searchings would be automatically generated and shown in a customized environment. This paper presents an implementation of the WAY model, that defines how to build systems that adapt to the user reducing the complexity of the searching process, saving user time and offering personalized interfaces. The implementation of the model, WAY-Z39.50, shows an example of application to the field of digital libraries and supported by servers that implement the Z39.50 search and retrieval protocol [4]. A complete session of a user accessing the WAY-Z39.50 system serves as connecting theme to discuss the main aspects of the interface adaptation as well as the advantages and disadvantages of providing parallel searchings.

2. GENERAL MODEL
WAY is a model of architecture for adaptive information systems whose main goal is to easy the searching process by offering an adaptive interface. This interface evolves according to the user preferences while helping her in the information searching process. WAY adaptation to the user is based on the interaction between them. This adaptation is made both on the interface, and on the searching process, which are the two main tasks of the system. Another mission of a WAY implementation is to keep updated information about the available information servers. Figure 1 shows an example of how WAY adapts its interface according to the user. The system provides a different interface to each user: a text interface for User 1, a graphic interface for User 2 and a multi-window interface for User 3. Adaptation process generates different results starting from a base interface and depending on the actions that each user has performed in her interaction with the system. The result is a different and adapted interface for each of them.
- She does not need to leave one server just to visit another one in order to repeat the same searching, in the case that she does not succeed in her search.

- Searchings proposed by WAY are made while the user is checking the results obtained by her own searching. In this way, if she wants to consult the results of an alternative searching, she does not need to wait for the new server to send the answer.

WAY main activities can be summarized in the adaptation of the interface to the user, the support in the searching process based on the user actions, and the extraction and maintenance of updated information about the available information servers. Figure 3 shows the basic architecture.

In the figure the following elements can be observed:

- **The user** who interacts with the system in her particular way which will be probably different from other users. Each user is usually interested in her own goals when using a particular information system, and if two users would have similar goals, the way each of them would try to achieve them might be different. For instance, when a user is getting archives using FTP (File Transfer Protocol), she could prefer to access to the server using a unix-shell style while, another user could choose web interface and use a browser that supports FTP.

- **The WAY system** that is a mediator between the user and the information servers. The structure of this systems is made up of the following modules:

  **User Interface Adaptation:** is the module directly related to the user of the system and its function is the interaction with her. This modules receives user requests and it sends them to the module in charge of dealing with the communications with the information servers. It receives the answers obtained from the servers and shows them to the user. In order to accomplish this process it receives data about servers configuration (as for example searching fields offered by a particular server) from the module in charge of keeping updated this information.

  On the other hand, it observes the whole process in order to extract information useful to propose to the user modifications to the interface. For example, if a user searches very often using the fields in a particular order, she will be proposed to modify her interface to get fields in that particular order.

  **Collaboration in the searching process:** is the module in charge of the communication with the information servers.

  First, it proceeds with the requests of access to the information servers. These requests may have been generated by the user or by the system itself. In the first case, the requests have been generated by the user through the interface, and in the second one they are requests generated by this module itself, or requests of information about the services offered by the information servers. In the last case these requests have been generated by the module in charged of keeping information about servers updated.

  Second, this module proposes new searchings based on the information obtained from the user requests. This means that in addition to the requests formulated by the user, the system launches, at the same time, new searchings on the user preferred servers. This way, the process is speed up, as the user obtains the results from other servers without repeating the searching.

  **Information about servers:** This module is in charge of keeping updated information about information servers.
Its first mission is to locate servers in the net. In order to do that, this module accesses information sources about servers directly. This information sources may be web pages, electronic mail lists, USENET groups of news, etc.

Once the agent has located the servers, this module obtains and keeps information about the services offered by each server. To do so, it counts on the help of the module that communicates with the servers. Some examples of the required information can be which kind of searching does the server support -useful information for the module in charge of the communication with the servers- or the names of the data bases contained in each server -useful information for the module in charge of communicating with the user.

- **The information servers** available through the net can be grouped depending on the communication protocol they support. The implementations of WAY have a strong dependance on the kind of servers they are accessing, as they determine the protocol that the system must also support. For instance, web servers use the HTTP (HyperText Transfer Protocol) protocol, so the system accessing this kind of servers will compulsory have to support that protocol.

### 3. WAY-Z39.50

An implementation of the general model is WAY-Z39.50. The name of the system is due to the protocol z39.50, the one implemented by the information servers accessed. The application field is digital libraries as most libraries all over the world are implementing this information search and retrieval protocol in order to offer a standard way to consult their funds.

**Description**

The three main tasks of the system, that is, adapting the interface, proposing and performing new searches and keeping updated servers information, are performed by three intelligent agents [2] whose decision making process is based on a fuzzy logic system:

- **The Interface Agent** The task of this agent is to spy the user actions in order to decide the best way to adapt the interface to the user. The agent takes decisions and transmits its requests to the interface software in order to perform them. The user can always decide not to allow the agent to change anything.

- **The Information Agent** The task of this agent is to spy the user searching process. When a user makes a query to one information server, this agent looks for the servers she prefers and launches other searching processes to the most appropriate servers at the same time. Thus, if the search has not been successful, other searches can be consulted without having to wait for the servers delay.

- **The Servers Agent** The task of this agent is to maintain updated information about the accessible servers. To perform its task, this agent can use some information sources as web pages, mailing lists or news groups. The agent periodically consults these sources and updates the servers’ list. The period is also updated by the agent itself. There is only one servers agent for the whole system.

**Implementation**

Since the main goal of the model is to make easier for the user to access information, the applications based on the model should themselves offer, as long as possible, an even easier way to be accessed. In order to accomplish this feature, it was found that a web page could be quite a common and broadly used vehicle to get to the user. Using the web, some important points are achieved:

- the environment is already familiar to the user
- the user needs to install nothing but connecting to a web site
- the execution of the application takes place at the user’s computer although no software has to be locally stored

To achieve the last point, it was decided to build a java applet, that is, a java application running in a web page and executed every time a user visits that page. This way we had an application running in the user’s computer with no local storage. The problem that arises next is where to store the information used to adapt the interface to each user if we are not accessing the user disk. The solution adopted was to use the applet server machine as a data server too. So, there is a machine where the applet code is served to anyone visiting the WAY-Z39.50 page that uses a PostgreSQL data base to store important information for every user. The access to data bases is quite simple if JDB (Java Data Base Connection) is used, as it provides the required connectivity by means of the well-known SQL language.

The problem of using applets is that, in order to protect web users from suspicious implementations,
they can only establish connections with the machine where its code resides. So, we had to face another problem: how to access the Z39.50 servers. In this case, the answer was to use RMI (Remote method Invocation), the distributed part of java programming language. This way, the part of the distributed system running on the user side could access the server side and the server side could access the servers and give the user the data back. This solution gave us also the possibility of storing the user session in order to obtain valuable data for the evaluation phase.

As a result of everything mentioned above, and with the help of a essential tool as design patterns are (see [1]), the implementation of WAY-Z39.50 is an object-oriented distributed system written in java (using Java Development Kit v. 1.1.5) composed by the following packages:

- **Interface**
  In this package, all the graphic elements of the system are included. (Data related to the interface package) 12,500

- **Agents**
  This part includes a subpart where the mind of the agents is implemented using fuzzy logic (see [5] for a detailed description of this issue or [3] for authors' related works on that field). The number of code lines is three thousand.

- **Z39.50**
  The primitives that the protocol specifies for the client side of a Z39.50 communication are included in this part. (Data related to the Z39.50 part) 5,000

4. A SESSION WITH WAY-Z39.50

Once the system has been described, a working session is presented in order to show in detail its operation. To access the system, the user will have to connect to a particular web page, so she will have to make use of a web browser. When a user enters the system, she is asked to enter a user name and a password. These data do not have a security purpose, they are only used to save the information related to the system adaptation for that particular user for future sessions of that same user. This way, anyone can use the system without any previous requirements.

At this point, the system builds a particular interface for that user. If this is the first time the user enters the system, a default interface is shown depending on:

- The power of the user machine. For instance, a text interface will be shown to someone accessing from a 386 while a graphic one will be selected for a user owning a pentium III.

- The selected language of the browser used. Up to now, only English and Spanish are supported, so, the interface will be shown in Spanish if that is the preferred language or, in any other case, an English one will be selected.

During the user session, the system keeps watching the user actions. This information is used to suggest changes in the interface that the user can accept or refuse. For instance, if a visual handicapped user from England has chosen a Spanish interface with quite big fonts, as long as these are decisions directly taken by the user, the system will just obey them. But if a user searches very often by the author field and if almost every time that the search is done using this field the user gets a satisfactory answer, then this field will be proposed to be placed in the first place.

When the user makes a bibliographical query to one server, the first result obtained is the number of records matching that query. For instance, if the user has asked for 'Cervantes' in the author field, she will get the number of works from that author available in that server. If the number of records obtained is considered to be acceptable by the user, the next thing she can do is to consult a summary of the results in form of short records.

From the actions just mentioned, the system will deduce if the search has been somehow successful or not. It turns out to be quite obvious to think than if the user does not even take a look at the short records, the reason is that she considers the number of records obtained either too high nor too low. ...............If the information shown in these records seems interesting, she can visit their extended version. In this case, it can be supposed that she has found some of the information required or, at least, an interesting part of it. This kind of action is spied by the information agent. Thus, if the user enters nearly always book extended records, books will appear at the top of the results.

While the user is querying the system, the information agent is spying the process. This agent has information about the servers that the user visits most or are more adequate for her type of queries. While the request is being sent to the Z39.50 server by our system, the agent is deciding and launching new searches on the servers considered more suitable for that user. This consideration is reached through a fuzzy reasoning process whose inputs can be, for instance, the number of times that the user has accessed a server, the connection success average, or the speed of the connection and the number of long records looked up in that server.

The searches launched by the agent are suggested
to the user indicating the number of records found matching her query on other servers. The user saves time that she would employ in going to another server and repeat the query, as it should be done in the classical systems that integrate many library servers. Finally, the user can choose to go deeper into one of the proposed searches, refine the query of choose another server, beginning the whole process again.

5. CONCLUSIONS

In summary, the main advantages of WAY based systems are the following:

- They provide an interface that adapts to the user actions while she uses the system. To do that, the only information source are the user actions without asking her.
- They speed up the searching process by launching parallel queries and, again, without disturbing the user with any question, just spying past actions.

6. REFERENCES


