Advancement Towards Development of Sharable Accelerator MMI

Sarbajit Pal, S. Dasgupta
Variable Energy Cyclotron Centre,
1/AF Bidhan Nagar, Calcutta 700 064
&
Isamu Abe
High Energy Accelerator Research Organization (KEK)
1-1 OHO, Tsukuba, Ibarki 305, Japan

Abstruct

The concept of universal sharable GUI objects for constructing MMI of Accelerator controls has been extended. Code writing for member-functions of MMI class and its various sub-classes, developed earlier, have been implemented in platform independent Java. The general purpose MMI screen has been further developed to accommodate multiple graph displays in addition to the spread-sheet type parameter display. The sub-class MMI_Injector has been loaded with detailed functionalities for operation and monitoring the vacuum system and the power supplies of the transport line elements of the ECR ion-source of the VEC Cyclotron. The control and setting information are interchanged through the functions of the database class. Development efforts are directed towards transforming the above independent MMI classes to ActiveX components to facilitate their usage in other accelerator control systems. The ActiveX components are constructed in VC++ for obvious reasons, and are plugged into our application on VJ++.

1 INTRODUCTION

The phased developments for preparing an universal MMI console[1] has progressed further to accommodate more types of activated GUI's. In the current phase, the major thrust of the work has been towards connecting the universal MMI applet to the control database for dynamic updating of tabular or graphical display of parameter values. With this objective, the GUI richness of VB, the platform independence of Java and inter-operability with specific DBMS, have been exploited as and when suitable.

Several schemes of on-line updating of the console GUI's with data from a database server have been tried, using tools available with MS Development Studio or in freeware market.

Our Java applet, is capable of generating the tabular presentations. After installation of the 'MSChart' ActiveX in our applet, it is also possible to generate graphical presentations by using its Java Type Library functions.

2 DATA TRANSACTION SCENARIOS

Depending upon the placement of the control database in the distributed system, different schemes of data exchange between our universal console running inside a browser, and the database have been evaluated.

2.1 Database inside HTTP server

The first scheme is designed to make exchanges with 'MS Access' database, when it resides inside the HTTP server itself. DAO (Data Access Object) is the programming interface to the MS Access. This needs to be available inside the host where the applet will be running, to access data from the database server and to bring it to the client applet. Our console, thus enables the user to transact the records in a recordset of 'Access'.

However in this case again, the DAO object must also be available, else to be downloaded as a CAB file along with the applet, in case it is not pre-installed in the client.

2.2 Database in a dedicated server

In the second case, data is in a SQL server (Oracle) residing in a dedicated SCO-Unix machine as shown in the Fig.1.

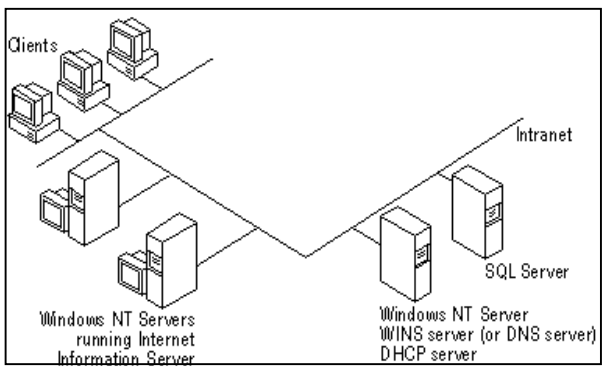

Figure 1. Client-Server NW structure

The scheme implemented for data communication with such a distributed database is by using the 'RDO' (Remote Data Object), available as a tool, in the Developer Studio. During programmatic operations, the properties and methods of 'rdoConnection' and

'rdoResultset' objects are used. But, the downloaded applet need to be accompanied by digitally signed CAB files in this case also.

2.3 Access from any Browser

These two schemes are satisfactory for data access from any node-computer within the control LAN so far as ease of use and speed of response are concerned. The node, to work as console, of course must be equipped with Java-aware Browser and with the required ActiveX components, the conditions which are easily satisfied. In absence of that, even providing digitally signed CAB files as applets, is also workable within the intra-net.

However, a facility to circumvent the above restrictions and allow connections to the stand-alone database server machine, from any Browser, via an intermediate application, has been implemented. Such an application, its development and workings are detailed below.

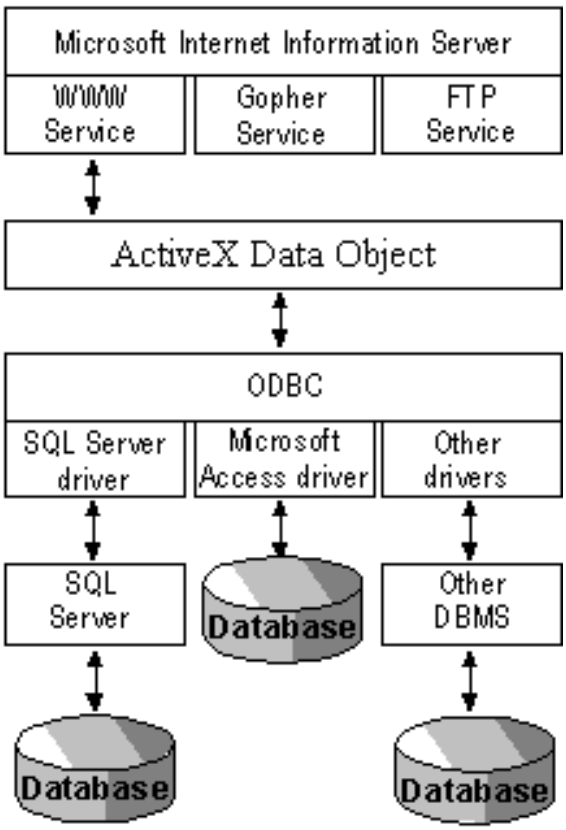

Figure 2. Data connection using ADO in server

3 CONNECTION THROUGH ASP

A small local client can be built within the server in a web page, using only HTML and sever side scripting in which HTTP sever does all the work. The advantage of this approach is that it works with any browser type. A tool 'MS-InterDev' helps in developing a Web application on the server and also connection to a database server. This web application, having file-name extension as the 'ASP' (Active Server Page)[2], is prepared with server script to process user input, access data, and generate dynamic HTML file to be displayed on client browser. Here is an account of how the console MMI queries can be satisfied through the actions of the ASP.

Our client browser's request is responded by HTTP server by creating a new **Session** object and initialising the global database connection objects. Server also sends a specific **Form** to the client for user interaction.

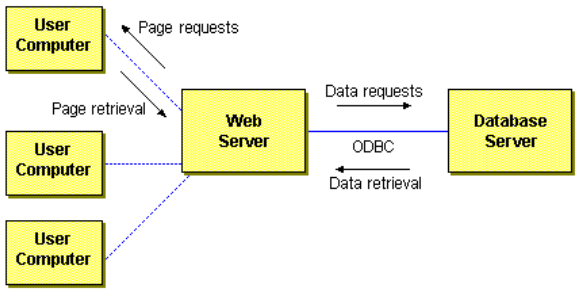

Figure 3. Data connection through http server

In the 'start.htm', the Universal MMI [1] applet is activated. In it, the user of the control system, can create named 'group's of parameters as per his choice in a pre-configured form. User gets this form, selects desired group and clicks the ***Submit*** button**.** The browser submits the input data specified by the user to the server with a **HTTP request**. The **Request** object holds data that is passed to the server application from where client address can be retrieved. Subsequently, as in normal practices[3], the **Request** will be sent by Java Socket from the applet to the URL of the ASP's host, to bypass the intermediate HTTP request.

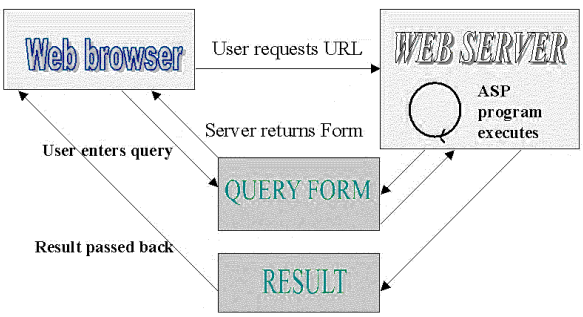

Figure 4. General relationship in ASP programming

Here, 'ADO' (ActiveX Data Object) is utilised to access and manipulate data from a local or remote database server, through a provider. Primary advantages of using ADO are, ease of use, high speed, low memory overhead. The other benefits of ADO for building client/server and web-based applications are - creation of independent objects, Batch updating, Free-threaded objects.

The start.asp has an **action** attribute of process.asp and method of **Post** with **target** as main. So process.asp gets evaluated after the arrival of HTTP request at the server

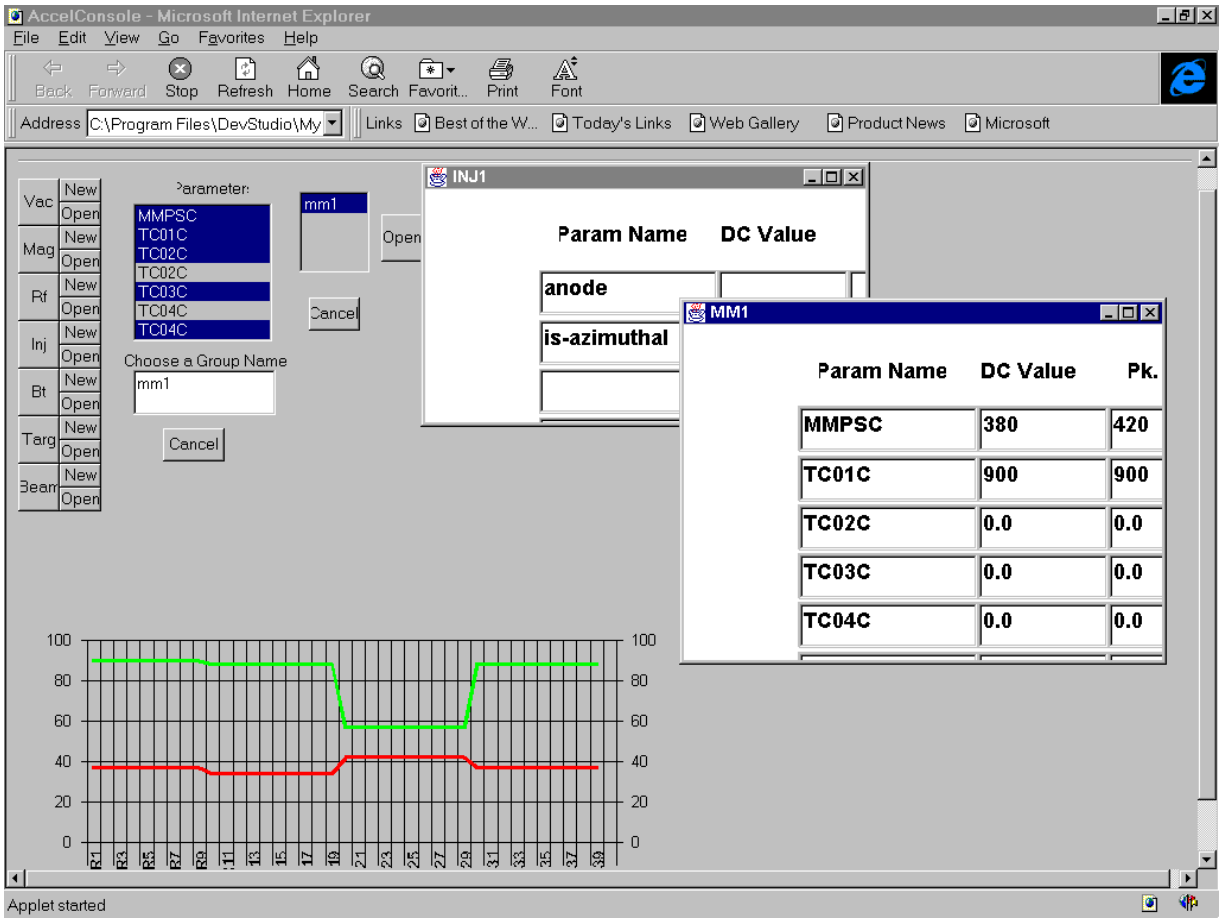

Figure 5. Extended universal MMI

site. The code or script in the process.asp executes an SQL select statement on the CONTROL database and creates a **recordset** object. The first record of the record set is set as the most recently stored input parameters of accelerator control and diagnostic systems. The HTML portion written to active page is transmitted which creates two frames on the browser. The 'index' frame will hold the *request Form* and the acquired values of the control parameters are displayed in the main frame in the tabular form.

4 SECURITY ASPECT

The security issues of the control database is a very relevant and important in this context. A user who requests an ASP page must have the correct permissions to view the page. This is ensured by setting permissions on virtual directories and by setting file access permissions (using Windows NT File System partitions) on our ASP files and folders. To improve security all other executable files are kept in a separate directory and are provided with only Execute permissions. Further security is provided by checking clients identity while database access by introducing a user authentication form at the beginning of a session.

5 CONCLUDING COMMENTS

The client server communication between browser client and 'ASP' is more versatile and more secure through proxy server or firewall when provisions of global connectivity is the isue.

In local data-transactios, quickest response is possible when an applet is directly connected to database server, even though it requires heavy-weight clients (equipped with VBX, ActiveX, DLL supports, which client applications often use).

ACKNOWLEDGEMENTS

The authors are thankful for the infrastructural supports and helpful discussions they had from T.K. Bhattacharya of this laboratory. They are also indebted to Prof. Bikash Sinha for his enthusiastic support in computerisation activities.

REFERENCE

1. S. Dasgupta et. al. 'Sharable GUI Objects for the Operators' Console' pp. 528, ICALEPCS '97
2. Microsoft Press ,'Visual Studio Getting Results'
3. N. Kanaya et. al. 'Remote Console System Using JAVA Applet for High Energy Accelerator Components at the Photon Factory' pp. 340, ICALEPCS'97