pharos
communications limited

pharos whitepaper

PHWP-3002.01_
Moving_out_of_the_Control_Room_and_onto_the_Enterprise desktop.docx

Spencer Rodd
01/02/2005

Contents

1 The Situation

1.1 Challenge

Until recently most Pharos control and media management systems were used by a closed group of users, physically located in a central technical area and connected to the system by a dedicated LAN.

The increasing use of electronic media and file transfer, together with the move of many production activities to the desktop, has led to an increasing need to access the transmission and media management systems from outside this core area and throughout the enterprise. This increases the number of users of the system from tens to hundreds. The current architecture does not naturally support large user groups.

1.2 History

The mediator system is built around a client/server architecture. The clients connect directly to the database and to a messaging system. The middleware tier communicates with the clients either by messages or the use of additional tables in the database used to store status information. The databases are configured as a pair with one being active and the other standby.

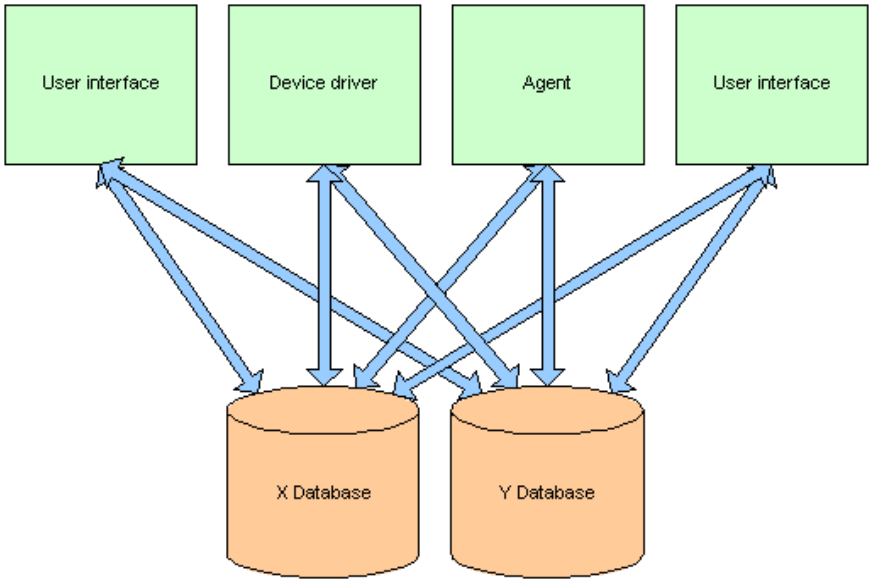

Figure 1. Illustration highlighting the traditional method of connection between databases and messaging systems.

All reads come from the active database, this constrains the system performance to the number of reads which can be made from the active database in any given time period. For a typical transmission control system, this gives an upper limit of about 50 users before performance starts to be impacted.

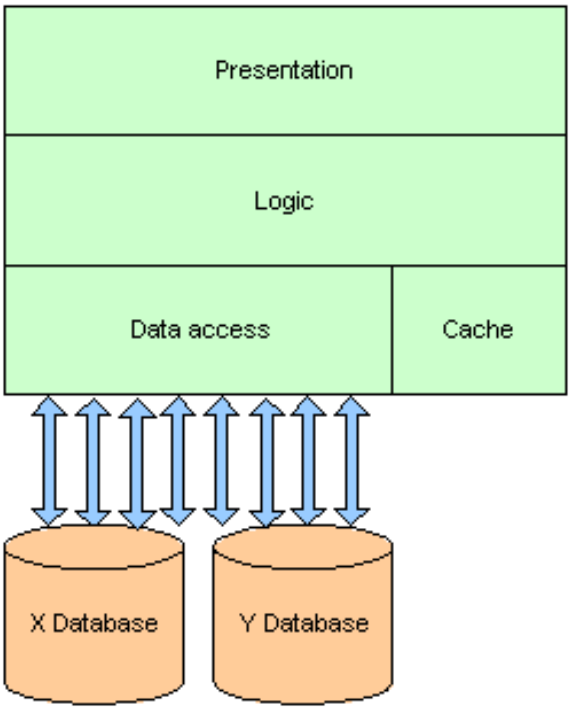

Figure 2. Illustration highlighting the connection between databases and data cache.

To allow expansion of the system beyond 50 users a data cache has been added to the data access layer of the client to hold a copy of frequently referenced data. This allows the system to support more users but creates the problem of rolling out changes. Since so much of the business logic now resides in the application any changes require all of the clients to be updated. In large enterprise systems this can be a very slow process involving local engineering and IT approvals.

2 **Options**

Investigating possible solutions it was clear that a web browser based user interface was preferred. The key risks to this approach were the lack of a mechanism to send unsolicited updates to the web browser and a potentially slow user interface. A research project was started with the objective of evaluating these risks. The chosen project was a router control system with a web based user interface, this project highlighted both risks. We settled on the use of java applets as the best way to provide a mechanism for unsolicited messages. A system of objects was devised which allowed the downloaded page with its applet to subscribe to updates from a server. This model mirrors our client server messaging model.

2.1 Database Access

In order to move the rest of the Pharos product range to a web based GUI, fast and reliable access to the SQL database, which forms the core media management and scheduling engine, is required. In previous 'view only' web interfaces to the database we had used a Delphi program on the server which when run by the web server produced html for the returned page. This required a database connection to be set-up and cleared for each data set returned and proved difficult to extend or adapt to support more complex user interfaces.

2.2 J2EE

Java 2 Enterprise Edition aims to solve the problem of widespread access to common data in a scaleable manner. The standards are administered by Sun and supported by industry partners including IBM, BEA and Oracle and open source work under JBOSS. The J2EE platform simplifies enterprise applications by basing them on standardized, modular components, by providing a complete set of services to those components, and by handling many details of application behavior automatically, without complex programming. The architecture is used by many large companies for applications with extremely high transaction volumes.

2.3 Using J2EE for Pharos Mediator

J2EE fits easily into the middleware layer of the existing pharos systems. By adding it in this way we can preserve the existing interface as we migrate interfaces to the new access method. The following diagram shows the interfaces within a Mediator system.

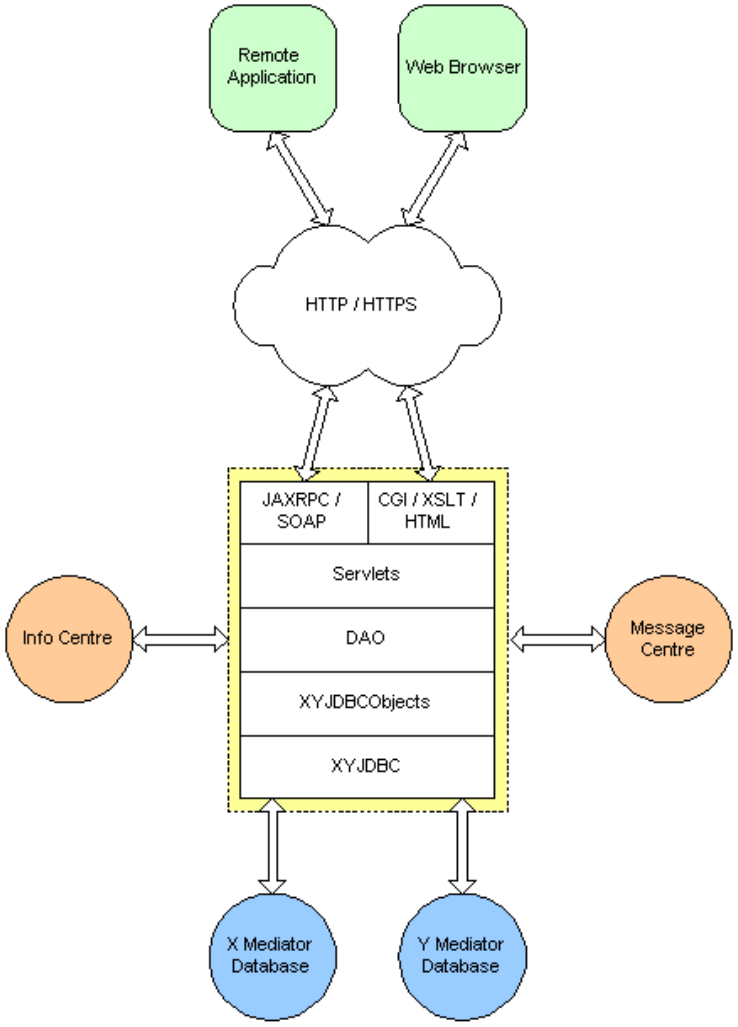

Figure 3. System architecture connectivity diagram

The yellow box in the diagram represents the J2EE application server. The boxes inside are:

XYJDBC provides the access mechanism to the existing dual databases. This will be built in a manner to allow more than 2 database engines to be running in the future.

XYJDBCObjects this is a cache of objects representing the data in the database. To allow applications to continue to use direct database access and to simplify the use of multiple application servers the cache will be used for reads only. Any writes of data will update the databases which will then be used to update the cache.

The DAO layer consists of the data access objects which provide a common object interface to the layers below. The objects here are

modeled on the underlying media management and scheduling concepts such as material, media, package etc.

Servlets are triggered by the application server in response to requests from either the JAXRPC/SOAP access mechanism or the CGI/XSLT/HTML mechanism. The servlet performs the task required using the DAO and internal logic within the servlet.

JAXRPC/SOAP access mechanism. This allows other business or control systems to interchange and access the data and control paths within the system.

CGI/XSLT/HTML access mechanism. This is used to provide the web based user interfaces into the system.

2.4 Front End

Web page user interfaces are used to provide a highly scalable architecture. The only software required at the client end is a standard web browser. So installation is very easy. A new user is simply added by providing them with a user name and password. Maintenance is also very simple as software changes are only ever required at the server end. Any changes to software are available at the next request.

The web pages delivered from the Application Servers can contain Applets when asynchronous communication is required. For example an Applet can have data streamed to it for display or it could be as simple as a progress bar showing accurate status.

Applets can also be used where more complex user interfaces are required, a playlist editor for example.

2.5 Network

Load balancing is used to route requests to available servers. A number of solid state load balancer appliances (these can be configured in pairs with a fail-over mode) are used. All requests are handled by the load balancers and dispatched to individual servers. If any server fails a health check it is immediately taken out of the cluster so that clients are not affected.

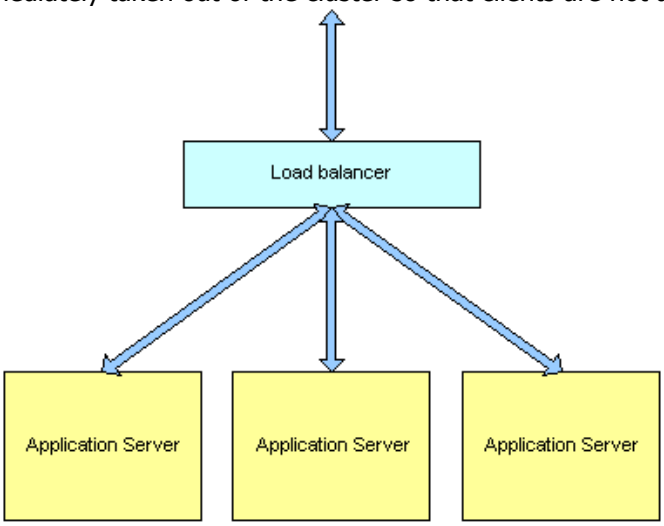

Figure 4. Network loadbalancing architecture diagram

3 Conclusion

The use of J2EE allows the existing mediator, playtime, pilot and almanac product lines to be migrated to a completely scaleable architecture for user and system access while maintaining the reliable and secure control model currently used. Early indications are that the architecture lives up to its promise.