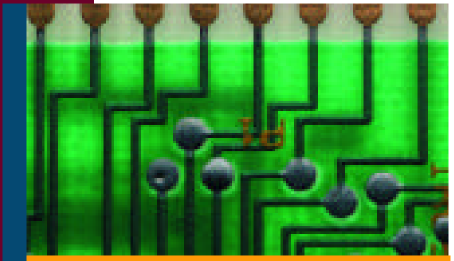


BY RICHARD B. VIPOND

Legacy Systems Enablement: Part II — Laying the Foundation for Selecting a Development Tool

The next step in our goal of Internet-enabling our legacy systems is the selection of a development tool that will enable us to create an Open System development environment.

Editor's Note: *The figure presented in Part I was incorrect. It is included here as Figure 1. We are sorry for any confusion this may have caused.*

THIS article is the second in a series on Internet-enabling our legacy systems. This series is based on two of the most important and controversial issues facing any enterprise today, namely adapting to the rapid rise of the Internet and the fate of legacy systems.

In Part I (***Technical Support***, July 1998) I stated that implementing object technology should not be the short-term solution to development requirements. Rather, we want to select development tools and emerging technologies that will enable us to create an Open System development environment from which we can grow. In Part I, I also noted that the available solutions can be overwhelming. I concluded that since keeping it simple is not an option, adhering to a standard is the next best thing. The standard, when applied to the selection of a development tool or emerging technology, consists of a set of capabilities that should be present in any development tool or emerging technology that is chosen. Whether or not you agree with this basic premise, I am interested in your comments.

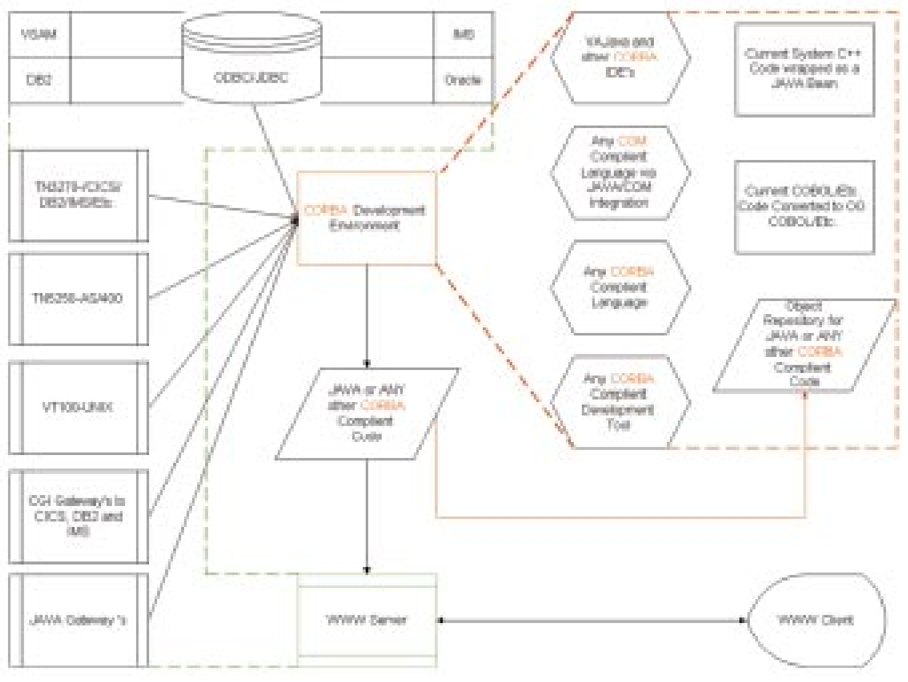
The first step in our journey is to use proven open technology standards in the design of all systems. Let's put things into perspective by providing an in-depth analysis of proven workable solutions that will produce open architecture components that will fit into a future implementation of an object-oriented standard such as CORBA (Common Object Request Broker Architecture). CORBA provides a basic framework on how objects can send and receive requests.

THE TESTING ENVIRONMENT AND ITS GOALS

I work in an excellent research and development environment with opportunities and challenges similar to those of most large corporations. My client is a multi-billion dollar global distributor. The production environment consists of numerous operating systems and platforms, including a CMOS mainframe that runs OS/390 V1R3 and OpenEdition. CICS, DB2 and IMS host many of the company's legacy systems. AS/400 systems run the remaining legacy systems that control the majority of order processing. Sun Sparc workstations run the Internet ordering system using the Netscape web server, which communicates to an Oracle database with EDI (Electronic Data Interchange), file transfer, and MQSeries to the legacy ordering systems. A newly announced integrated supplier system is the result of a joint effort with another multi-billion dollar global distributor. This system will provide one-stop shopping for the customers of both companies.

At the client site we have created a mirrored test environment of the production system with additional platforms and operating systems interspersed as needed to provide a more flexible test environment. We have IBM's ICSS (Internet Connection Secure Server) web server running under OpenEdition in a separate LPAR (Logical Partition) using OS/390 V1R3. An NT Server platform has been set up to run and test various middle-ware products. ODBC (Object Database Connection) is also being used on all platforms to provide communication with the Oracle database.

Figure 1: Example of Concept Requirements


VSAM
IMS
DB2
ODBC/JDBC
Oracle
TN3270-/CICS/DB2/IMS/Etc
CORBA Development Environment
TN5250-AS/400
VT100-UNIX
CGI Gateway's to CICS, DB2 and IMS
JAVA Gateway 's
JAVA or ANY other CORBA Compliant Code
WWW Server
VAJava and other CORBA IDE's
Any COM Compliant Language via JAVA/COM Integration
Any CORBA Compliant Language
Any CORBA Compliant Development Tool
Current System C++ Code wrapped as a JAVA Bean
Current COBOL/Etc. Code Converted to OO COBOL/Etc.
Object Repository for JAVA or ANY other CORBA Compliant Code
WWW Client

Another NT Server runs Microsoft's IIS web server and supports a Data Warehouse that is being implemented using legacy system data. A Sun Sparc workstation runs a cloned test version of the Internet ordering system using the Netscape web server. The primary client operating system is Windows 95, however, several NT Workstations are used for heavier client operating system demands and experimentation requirements.

WHY DO WE NEED TO INTERNET-ENABLE OUR LEGACY SYSTEMS?

I already know the first suggestion that everyone in the mainframe world is going to make: "Just put it all under OS/390 and get rid of the junk." While I might agree that the OS/390 server is the most advanced and functional software available on the planet today, it is not a cure-all and never will be. All of these systems have their strengths and we need to position ourselves to take advantage of them. We also need to be able to take advantage of the existing expertise on other platforms for the same reasons that we need to Internet-enable our legacy systems.

Protecting our investment in application development is a primary reason for using our legacy systems in our Internet initiatives. This protection should continue to be a priority in the development of future systems.

SELECTING A DEVELOPMENT TOOL

Now that we know what our goals are, where do we start? How do we quickly eliminate those development tools that do not provide what we need? Well, they have to perform a defined set of functions very well! I have also come to realize that GUI development tools are a mandatory requirement. I feel GUI and WYSIWIG development tools are vital to the future success of application development in the 21st century. Part III will explain why.

While this article lays the foundation for what we should be looking for in a development tool, it does not specifically list the products we reviewed at my client's site. Since each company is unique, it's best to let you do your own review based on the following guidelines. We only evaluated tools that fell within these guidelines. There are many different tools that provided various functionality required for a particular project, but if they didn't fit within these guidelines then they were quickly eliminated from consideration.

Defining the Features and Capabilities

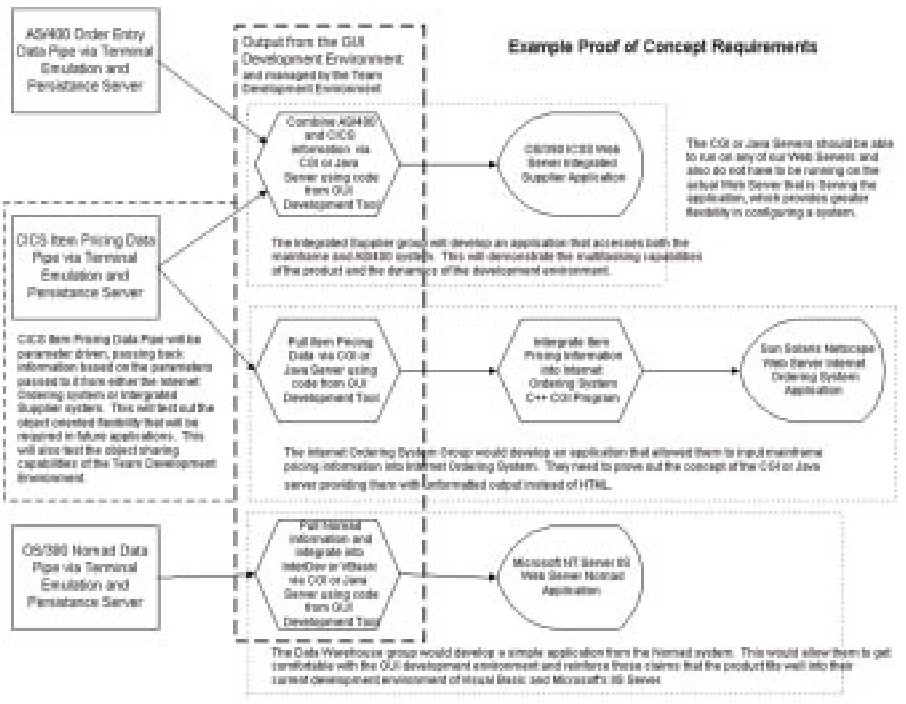
We must find products that contain all of the elements necessary for the creation of an open development environment. Whatever the application development tool creates must be capable of being integrated into an existing environment with minimal effort. This provides enormous application development flexibility for new and existing applications. Communication with programs outside of the development tools environment must be supported, as must communication between application programs. The concept of communicating with a program that isn't actually executing refers to dynamic code reuse and code manipulation of objects. Part III will explain this concept in much greater detail, but the code produced by the development tool must be able to accept input parameters from other programs or processes outside its own execution environment. Failure to use an open communication environment creates an execution environment that is limited to self-contained transactions.

The following three features or capabilities are so general that they almost seem unnecessary to mention. However, I am amazed at how many software vendors and individuals who fail to include these essentials in their products!

- First, all code that is developed should enhance the functionality of the web browser and client platform, not restrict any of the existing functionality. Failure to do so locks the company into an architecture that is not compliant with any of the chosen standards. It also makes code reuse outside of that development environment impossible. Since browser technology is also changing rapidly, open standards will be critical. Code should only be limited by the limitations of the web server being used, never by the development tool. Code must be executed by non-proprietary components in any web server environment.

- Second, the application development environment must allow for team development. If the design of a project has many pieces that could be developed simultaneously by many developers, then the development tool should support these capabilities. Ensure the product provides automated documentation capabilities. Even small projects need to have an automated documentation process that is supported by the development tool being used. This documentation provides the necessary information for development personnel to take up where someone else left off and also to perform future maintenance.

Figure 2: The Proof of Concept Pilot


AS/400 Order Entry Data Pipe via Terminal Emulation and Persistance Server
Output from the GUI Development Environment and managed by the Team Development Environment
Example Proof of Concept Requirements
Combine AS/400 and CICS information via CGI or Java Server using code from GUI Development Tool
OS/390 ICSS Web Server Integrated Supplier Application
The CGI or Java Servers should be able to run on any of our Web Servers and also do not have to be running on the actual Web Server that is Serving the application, which provides greater flexibility in configuring a system.
CICS Item Pricing Data Pipe via Terminal Emulation and Persistance Server
The Integrated Supplier group will develop an application that accesses both the mainframe and AS/400 systems. This will demonstrate the multitasking capabilities of the product and the dynamics of the development environment.
CICS Item Pricing Data Pipe will be parameter driven, passing back information based on the parameters passed to it from either the Internet Ordering system or Intergrated Supplier system. This will test out the object oriented flexibility that will be required in future applications. This will also test the object sharing capabilities of the Team Development Environment.
Pull Item Pricing Data via CGI or Java Server using code from GUI Development Tool
Intergrate Item Pricing Information into Internet Ordering System C++ CGI Program
Sun Solaris Netscape Web Server Internet Ordering System Application
The Internet Ordering System Group would develop an application that allowed them to input mainframe pricing information into Internet Ordering System. They need to prove out the concept of the CGI or Java server providing them with unformatted output instead of HTML.
OS/390 Nomad Data Pipe via Terminal Emulation and Persistance Server
Pull Nomad information and integrate into InterDev or VBasic via CGI or Java Server using code from GUI Development Tool
Microsoft NT Server IIS Web Server Nomad Application
The Data Warehouse group would develop a simple application from the Nomad system. This would allow them to get comfortable with the GUI development environment and reinforce these claims that the product fits well into their current development environment of Visual Basic and Microsoft's IIS Server.

- Third, beware of middleware servers that are limited to running on only one platform or that are not readily scalable. While the server eliminates the majority of screen scraped data sent out to the client and should provide optional persistence and terminal emulation capabilities, it also presents a major potential bottleneck in a system running thousands of concurrent users. Ensure that simultaneous (multithreaded) access to multiple host systems is fully supported. This is a fundamental requirement to providing adequate response times. Also ensure the product has no concurrent license fee based on the number of concurrent users. The ability to enforce concurrent licensing is a sure sign the product is not using a viable open architecture. Please note that I am not discouraging the use of middleware servers; however, ensure that they are scalable and do not lock you into concurrent users license fees.

The Proof of Concept Pilot

I've spent many hours looking at development tools and have come to the conclusion that we must do a proof of concept before making any final decisions. The Proof of Concept Pilot we use to evaluate development tools is shown in Figure 2. Each area has separate and distinct requirements that must be met. No one product will be a cure-all, but should be a very strong complement to the development model that is shown in Figure 1. The product should easily integrate with other tools and processes that will inevitably be found to further ease and enhance the development effort. Since every company has different and varying levels of expertise, this should also be taken into consideration when selecting a product.

We have three areas that will initially take advantage of these development tools. This is only the first step to integrating departments, subsidiaries, partner companies and future acquisitions into a development model fitting to a global company. The success of this integration process depends on utilizing proven tested standards. (**Note:** This proof of concept model will vary with each company, but is provided here to demonstrate where to start based on the open architecture objectives previously discussed.)

The Proof of Concept Pilot shown in Figure 2 consists of the following:

While I might agree that the OS/390 server is the most advanced and functional software available on the planet today, it is not a cure-all and never will be.

- The Internet Ordering System Group is incorporating application logic into their existing system. This allows them to input mainframe pricing information into the Internet Ordering System. They will also be entering real-time orders in the near future but they need to ensure that the CGI or Java server will provide them with unformatted output instead of HTML. While numerous products and facilities provide unformatted output instead of HTML, finding a product that will do this and provide a GUI development environment with no requirements to change legacy system code is a real challenge.

- The Data Warehouse group is developing a simple application from a mainframe system Nomad database. They will eventually be Internet-enabling large amounts of legacy system data. However, the pilot will allow them to get comfortable with the GUI development environment and reinforce any claims that the product fits well into their current development environment of Visual InterDev and Visual Basic running on Microsoft's IIS Server. This also means that it is necessary to find a product that will fit well into both Microsoft's DCOM (Distributed Component Object Model) and the CORBA environment.

- The Integrated Supplier group is developing an application that accesses both the mainframe and AS/400 system. They will eventually combine numerous companies' legacy systems into a common Internet ordering system. The pilot will demonstrate the multitasking capabilities of the product and the dynamics of the development environment.

There are also other common requirements that must be taken into consideration. For example, the CICS Item Pricing Data pipe must be parameter-driven, providing feedback based on the parameters passed to it from within the Internet Ordering System or Integrated Supplier system. This will test the object-oriented flexibility that will be required in future applications. This will also test the object sharing capabilities of the Team Development Environment software that should be part of the product. The CGI or Java Servers should be able to run on any of our Web Servers and also have the flexibility to run on a server different from the Web Server that is serving the application. This will allow for much greater flexibility in configuring and tuning a system.

THE NEXT STEP

In Part III, I will explain why I feel GUI and WYSIWIG development tools are vital to the future success of application development in the 21st century. The concept of communicating with a program that isn't actually executing using dynamic code reuse and code manipulation of objects will also be discussed in detail and related to current "bleeding edge" development concepts. I will also further discuss the CORBA standard and the other standards that I mentioned in Part I, which are under development. Remember, in the event that we do want to adopt another standard in the future, at least we will have a standard to convert from.

A special thanks to NaSPA members and technical editors Dwight S. Miller and Stephen J. Pryor for their help with this article. ts

NaSPA member Richard B. ViPond is a senior consultant for Ciber Network Services, Inc.