Migration, Continuity and Web-Based Apps

Leveraging Web-Based Apps for Business Continuity in HP3000 Migrations

Presented By: Michael L Gueterman
Easy Does It Technologies
http://www.editcorp.com
m_gueterman@editcorp.com

Overview

- Problem Definition
- The Solution – Web Applications
- Benefits of using web applications
- Alternative reasons to use web applications (besides migration from the e3000)
- Case Studies
- Design Considerations
- Where ***not*** to consider a web application
- Summary

The Problem

- Migration plans are not firm, underway, or even started.
- Critical/Required applications are still needed.
- Little time/resources available to deploy then migrate.
- Target migration platform may exist, but the application requires access to legacy information which has not been migrated.

The Solution

- Web-based application using an independent application server.

The Benefits

- Consistent graphical user interface across different end user hardware/software.
- Thin client requires no additional software deployment to the end user.
- Data locality remains centralized to host/application server.
- Back-end data access can change with little to no change in the user interface (back-end agnostic)
- ***Application can be developed/deployed today while migration efforts continue***

Alternative Reasons for this Strategy

- Application requires Internet/Intranet access to host-data.
- Distributed access to application required.
- Need a fresher (graphical) user interface.
- Application may require additional sources of information.

Case Studies – Client #1

Current environment; HPe3000 centralized host with dozens of smaller remote systems at plants spread across the country. They wanted a centralized application, accessible by all plants for common and site specific information. Their available talent was mainly COBOL/Image. Long range plans call for the replacement of the centralized host with either HP-UX/Oracle or MS Windows/SQL Server.

Case Studies – Client #2

Current environment; Single HPe3000 running all of the applications for a performing arts organization. The applications are a mix of COBOL/View and Powerhouse with TurboImage.

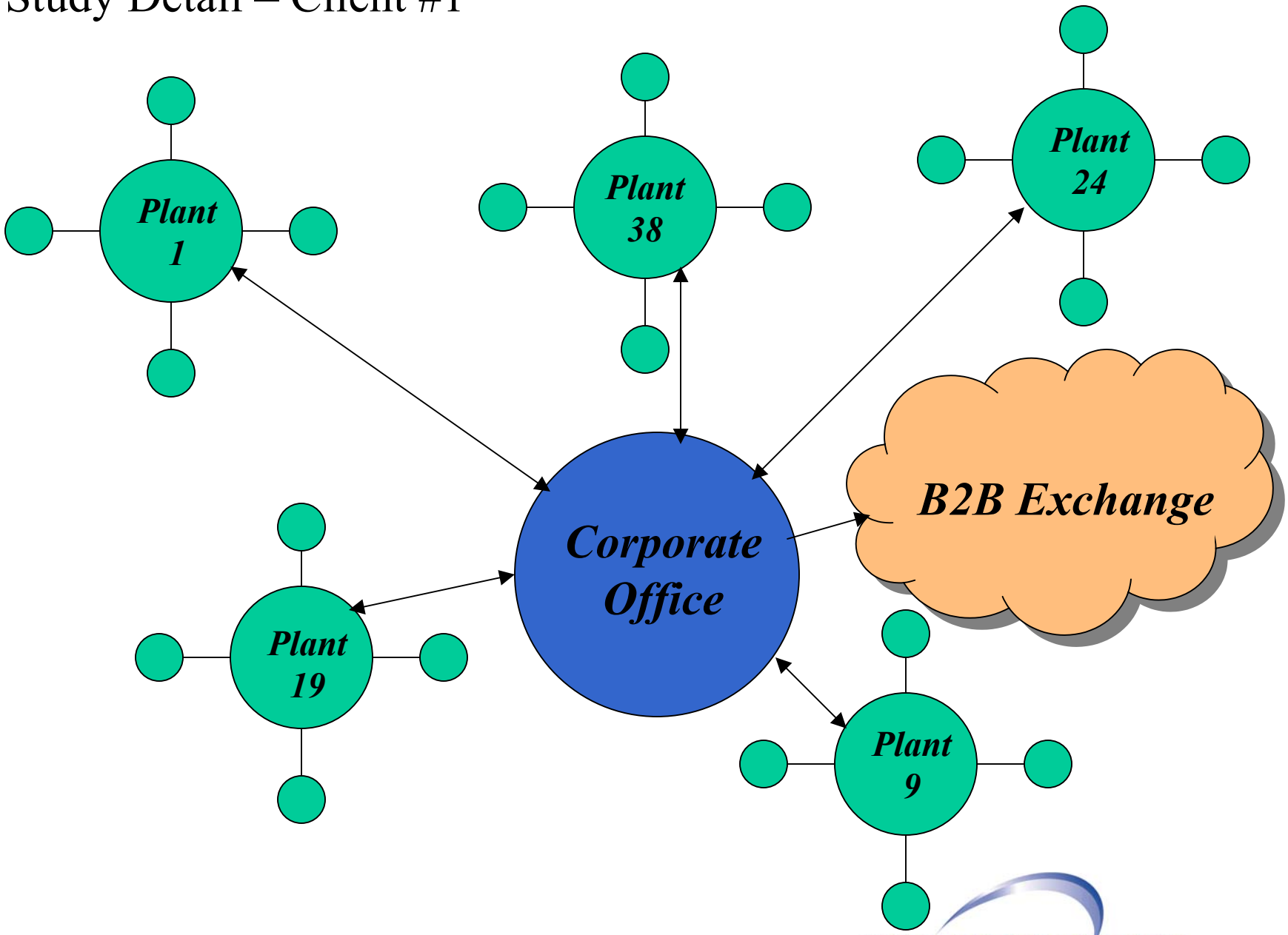
They wanted to extend the host-based functionality of their ticketing and patron development applications to their Internet customers.

Case Studies – Client #3

Current environment; Dual HPe3000 servers. One running all critical business applications and the other receiving daily data extractions of inventory information into a TurboImage database.

They wanted to create a web site that would continue to function regardless of what host-based application (or system) was in use.

Case Study Detail – Client #1



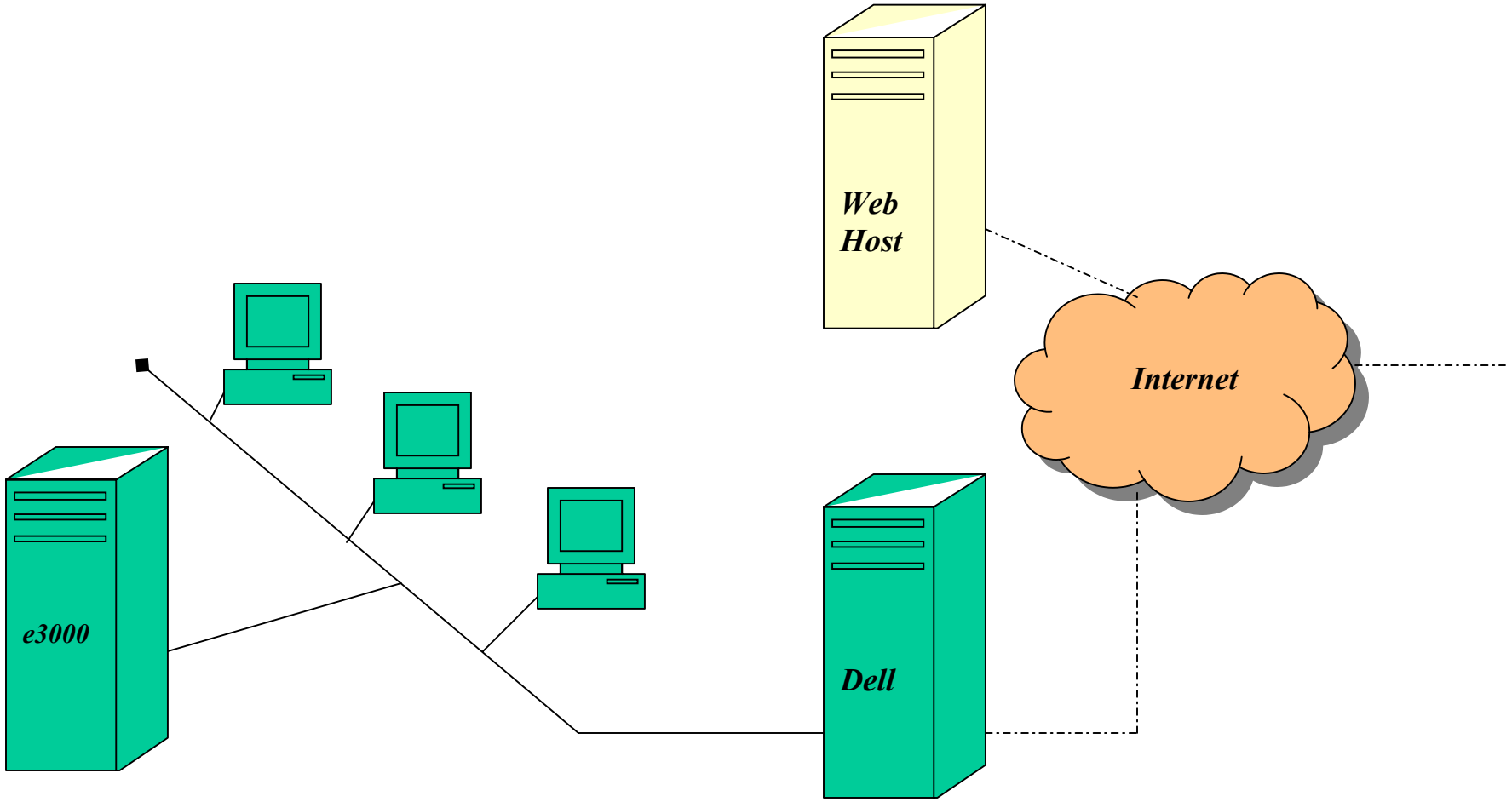
Case Study Detail – Client #1 (continued)

- The evaluation process for the replacement of the Corporate Office e3000 systems was already underway.
- The expected migration platform was HP-UX/Oracle, but a final design had not been made, and whatever the outcome, the replacement system would still be several years away.
- The new application to feed the B2B exchange was an immediate need that could not be postponed until after the e3000 migration.
- This new application must therefore leverage both the current MPE/TurboImage and future HP-UX/Oracle environments.

Case Study Detail – Client #1 (continued)

- A Centralized TurboImage database on the e3000 utilizing an ODBC Interface, maintained via a web application accessible to all plants over the Intranet was chosen for the initial design.
- HP Netservers running MS Windows, IIS, Macromedia's Cold Fusion and Minisoft's ODBC/32 would be used as the web application environment.
- This environment was chosen so that when the back-end database and B2B exchange application were migrated to the new platform, the application used by the plants would continue to function with only minor changes to the SQL routines. The user interface would remain consistent.
- As the existing staff was primarily COBOL/TurboImage, Cold Fusion provided a low learning curve, high performance solution.

Case Study Detail – Client #2



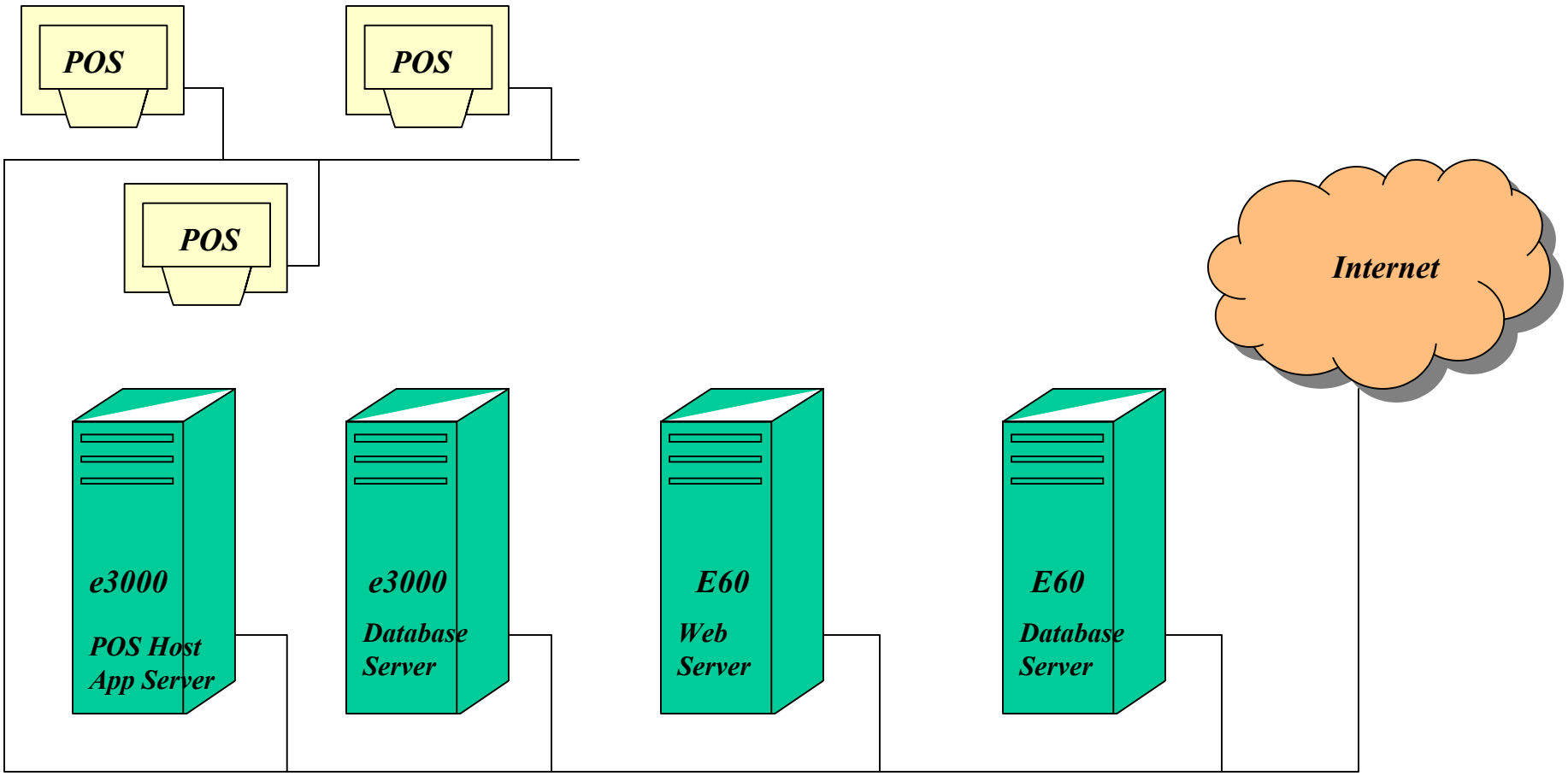
Case Study Detail – Client #2 (continued)

- The host-based application they were using was purchased by a rival vendor and an End Of Support date was set.
- They wanted a flexible internet application that could take advantage of the existing host application as well as interface with whatever was chosen to replace it.
- Additionally, they were already maintaining an existing website (static pages only) and did not want to impact it but retain the same look and feel for the "Online Box Office" (OLBO) application.
- Dell hardware running MS Windows, IIS, Macromedia Cold Fusion, and Minisoft's Middleman and ODBC/32 was utilized.

Case Study Detail – Client #2 (continued)

- As the host-based application would also be selling tickets concurrently with the OLBO, a very tight integration of the two were necessary.
- In order to try and maximize the likelihood that the OLBO could be integrated with a different back-end application in the future, the OLBO was modularized such that information that changed infrequently was cached in memory structures and the database accessed only for seating and sales information.
- This resulted in a design where the actual user interface will be able to remain consistent as long as the resulting back-end application provides similar information as that currently used (which is likely given the nature of the business).

Case Study Detail – Client #3



Case Study Detail – Client #3 (continued)

- The desired web site must be able to incorporate information from a variety of sources including the e3000.
- The web site must not be tied to any one database architecture (database agnostic).
- As their webmasters are students, the application must be easily maintainable and extensible.
- After several different designs did not provide the desired level of performance, flexibility, or reliability, a combination of HP Netservers running MS Windows with IIS, Macromedia's Cold Fusion, Minisoft's ODBC/32, MS Access and MS SQL Server was ultimately arrived at.

Case Study Detail – Client #3 (continued)

- Information from the host-based applications are periodically extracted to another e3000 and loaded into a TurboImage database.
- The information in the TurboImage database as well as information from MS Access and MS SQL Server are utilized in various functions of the website.
- The host-based application vendor is in the process of migrating their application to a Linux/HP Eloquence based solution. When/if this replacement is installed, their website will require little modification in order to continue in its current function.

Case Study Summary

- Although each client had very different reasons for creating web applications, each had one thing in common; they all needed to create applications that could be used today without knowing what environment their host-based applications would be using tomorrow.
- By selecting the use of a web-application server and industry standard data access interfaces, they were able to leverage the hardware and software already available to their respective users, and position themselves to take advantage of whatever data sources will be made available to them after the migration of their host systems.
- This results in fewer migration concerns, a consistent user application throughout the process as well as reduced costs as applications are not being created and then ultimately undergoing an extensive migration.

Web Application Servers

Although Macromedia's Cold Fusion was chosen by the companies in the preceding case studies for their applications, there are many difference choices that are available. The decision on which to use depends on several factors:

- Experience level of existing staff with JAVA, ASP, or a tag based language such as Cold Fusion.
- Timeframe until the application must be operational.
- Expected load and performance levels (although most web application servers can now utilize server clustering, the initial designs may need to take the possibility of using multiple servers into account).
- Availability of data access interfaces (ODBC, JDBC, Native Drivers) to the desired data sources from the particular web application server.

Design Considerations

As the application will be utilized with more than one primary data source over its lifespan, several design choices made early on in the process will ease the transition from one to another:

•Segregate all I/O calls (primarily SQL statements) into modules unto themselves without user interface elements. This allows the same module to be re-used in difference application areas as well as allowing for changes in the underlying structure to not directly impact the user interface.

•Choose design element names that correspond to the user interface and not the underlying data structures (ex. Use 'Part_Number' instead of 'INV01PARTNUM'). This avoids confusion when the underlying structure changes.

Design Considerations (continued)

- Move the values into and out of these element names either directly in the I/O modules or in an intermediate module. This reduces the number of modules that will need to be altered when the underlying structure changes. When dealing with multiple record sets (tuples), use language specific features to store those values (structures, queries, etc).
- Whenever possible, avoid repeated database operations by caching application results. Some underlying database engines handle table structures in radically different manners which can affect performance. By caching these I/O's, the database is effectively removed from the user interface in those areas, both increasing the application performance as well as removing a potential migration point in the future.

Design Considerations (continued)

- Programming for a web-based application is a different mindset than for host-based applications. Some things to consider are:
  - All processing/Validation that can not be performed directly on the users browser (ie client side, similar to V/Plus forms processing) must be done by the “target” or “receiving” program. This means that if a user enters information that is invalid or unusable (ex. The user enters a syntactically valid part number but that part number has not been assigned in the database) the program which must perform the check will not be the one which the user just left, but the “next” one. This is a subtle but important difference as host-based programmers would normally validate a users input in the same program that requested the information to begin with. (continued)

Design Considerations (continued)

- (continued from prior slide)
  If the validation failed, they would simply repeat the section of the program which requested that information. In a web application, it is common practice to either have programs "calling themselves" and passing a parameter specifying what to do (so if validation fails, they simply execute the code to re-prompt the user), or to "redirect" the user from the validation routine back to the display/request program. Different web application servers have various mechanisms to handle this, but by considering this into the initial design, the application can be developed quicker and cleaner.

Design Considerations (continued)

- Every call to the database will (likely) result in a transfer of information over one or more network segments. Place the web application server in close proximity to the source database server. As this is the only point where access to the source data is made, it is important to keep the amount of time required to transfer the information back and forth to a minimum. The user interface will only hold the information in a format usable for the manipulation by the user, so the amount of data sent to them will normally only be a fraction of that utilized by the web application server to create the page in the first place.

Design Considerations (continued)

- As data access is performed between the web application server and the database server, no specialized drivers or programs need be loaded onto the user (client) systems. This simplifies the entire environment as the upgrade of driver software on the database server does not require a corresponding upgrade on each end users system, only on the web application server.

Where Web Applications should be avoided

Although web applications can be used to replace host-based applications in most cases, there are a few situations where they should be either avoided, or used in conjunction with a host or client/server based counterpart:

- Where the primary transaction/function is not user oriented, but rather "batch" oriented (ex. The calculation of plant-wide inventory need dates in a MRP system).
- Where large/formatted reports are required to be printed In general, if your information to be printed can not be displayed adequately on the browsers display, it will not be printed correctly (there are always exceptions as most any output format can be utilized, so creating reports that are downloaded and then printed is entirely feasible).

Where Web Applications should be avoided (continued)

•In situations requiring the interfacing with specialized hardware that does not emulate the output of a video display, or the input of a keyboard. Although as time goes by, hardware of this nature is also being converted for use in a web environment (examples would include security devices like signature capture or fingerprint readers).

Summary

The deployment of web applications in lieu of traditional host-based or client/server based applications is continuing to rise dramatically. They are a very flexible means of providing your end users with the tools they require while not dictating where or how the actual information is stored. For many companies, they will provide not only a means to a simpler and quicker migration from the e3000, but also to a new standard of application in the organization.

For further information

Easy Does It Technologies
Michael L Gueterman
m_gueterman@editcorp.com
http://www.editcorp.com
888-858-EDIT or 573-368-5478