

# Applying Internet Technology to Production Environment Monitoring

Alberto Tafur  
Project Manager  
CIMTEC  
Charlotte, NC 28273

Jeff Fleming  
Project Engineer  
CIMTEC  
Charlotte, NC 28273

John B. Weber  
President  
Software Toolbox  
Charlotte, NC 28273

## KEYWORDS

Internet, objects, ActiveX, Enterprise, 3-tier architecture, Ethernet, networking, MRP, ERP, client-server, TCO.

## ABSTRACT

With the growing number of Internet users, a familiarity with the web browser interface provides a substantial advantage to systems that use this interface to display information. Anyone who uses a computer today is able to easily navigate through a browser interface. By leveraging the user's familiarity with this technology, a common user interface can be used from the plant floor operator to the CEO at a remote location.

The purpose of this paper is to present the advantages of using internet/intranet technology in a production environment to control and monitor existing systems. By providing an application based on a three-tiered architecture with a platform independent client, a developer can deliver production environment information to every level of the company. Using object-oriented technology in conjunction with Ethernet networking reduces the cost and time to market and shortens the classic implementation cycle. This paper shows how this open architecture implementation enhances data flow through enterprise and legacy systems.

The design criteria for this interface were transparency, portability, flexibility, and simplicity. Using the common interface provides transparency to the data source. The browser, by nature, is both portable and flexible. Finally, a good layout of the web pages provides a very simple interface that can be used by anyone on any platform.

## INTRODUCTION

This application involves a major manufacturer with facilities located throughout the Southeast United States. Their process involves a number of discrete processing steps, some of which are continuous processes within those major production stages. Many of the existing control systems were mechanical or manual control. There was an existing central database where data was stored but all production data was manually collected and entered. A ticket system was used in which a ticket traveled with the product as the product batch moved through production. Operators entered the required production data on the ticket and at the end of the production, the ticket was collected and input by date entry personnel. Downtime was also tracked manually separately and logged into the system. With the large amount of paperwork,

opportunity for human error, and delays in data entry, the production information system was not capable of providing the information with the quality, format, depth, and timeliness that the business needed.

A control system upgrade was planned that would involve automating many of the machine processes using PLCs. To facilitate the automated collection of production information, the PLCs chosen would be required to offer a TCP/IP Ethernet interface so that the PLCs could be put on a plant-wide network and tied into the existing wide area network connecting all the plants.

The manufacturer realized that in order to move their plant efficiency and profitability to a higher level, both the plant floor and front office teams required a better exchange of information. A variety of business functions need operating data from the plant floor, including production planning, plant management, quality control, finance and accounting, and others. At the same time, the teams managing the production processes need information for daily production control, maintenance management, and overall process management and control, both from the process and from front office systems. Each function has specific needs for the type and presentation of the data. The one thing the functions have in common is the data is in existing PLCs, control systems, ERP and MRP systems, databases, and other legacy systems. The issue is business-wide delivery and presentation of the data in formats that make the data useful information for each function's specific needs.

There is no existing common system for delivery of this information. Some data is not even gathered such as the data from the PLCs, or is presented only at the operator workstation level. Other data is accessed through a variety of user interfaces on different systems. For some functions to obtain answers to some basic business and process control questions, they must consult multiple software applications on multiple computers. This problem is compounded by the geographical distribution of the firms' facilities. If a composite picture of operating information is required across all plants, the data must be manually gathered from the disparate systems, re-entered into spreadsheets or imported, then organized for common presentation. Each time data is updated, the process is repeated.

In defining the criteria for a solution, the manufacturer cited four key factors: transparency, portability, flexibility, and simplicity. All of the functions involved did not want to have to remember which system and which application to consult to get the different data they needed. With fewer resources than ever to do their jobs, they needed information so they could spend their limited time doing things to improve the business' profitability, not gathering data. The sources of the data needed to be transparent to the user – as far as the user is concerned the data is all in one place.

The solution needed to be portable to as many desktops as possible. With a wide variety of computing resources both in the plants and between all the physical facility locations, the solution needs to be easily deployed to all locations and centrally maintained.

Because each function's needs are different and do change over time, the solution needed to allow for easy modification of information presentation formats and content. This includes the adding of additional data sources and the creation of ad-hoc, on-demand views by the user.

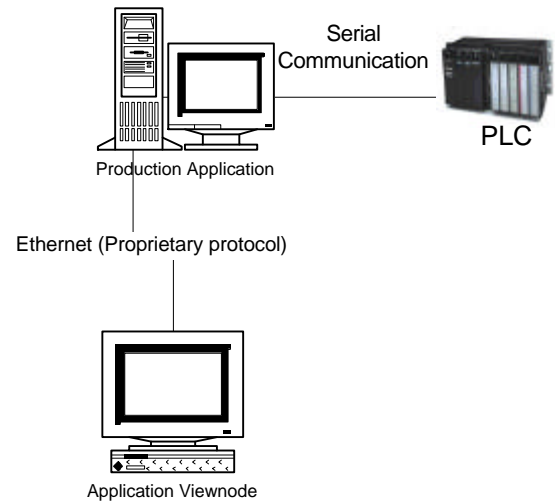
Last, the solution had to be simple to use and to maintain, both from the client perspective and from the systems management perspective.

## SOLUTIONS

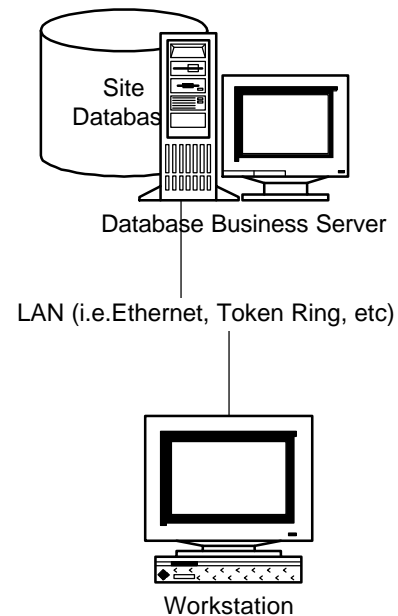
One solution considered for the application involved a traditional 2-tier application, as shown in Figure 1. This solution would deploy a common client application to each user that would have the capability to interface to each data system and gather the data needed through a variety of interfaces. The problems with this type of solution relative to the design criteria were numerous. First, the client application would require a custom application to be written and then deployed on every client machine. This immediately presents a maintenance and deployment issue. Since the client machines would be talking directly to all the disparate data sources, there would also be a network traffic issue with the overhead required for all the independent communications transactions. Even though the manufacturer had an existing TCP/IP-based Wide Area Network (WAN) in place between their locations, bandwidth was not plentiful. This solution also presents a potential issue with training on the use of a custom application and user interface. Although there are off-the-shelf software packages available that perhaps could have reduced the training costs, there still would have been significant configuration costs and the need for an expensive client-software application license on every PC needing access to the system. The type of software being used for the Human Machine Interface (HMI) strictly controls the look and behavior of this architecture. Furthermore, the view stations are required to have a specific operating system and high processing requirements to handle the specialized application required to connect to the application server to view and/or control the process.

Figure 2 shows the classic business/corporate application architecture that controls customer orders, inventory, shipping, etc., configured as an independent 2-tier system. In addition, information flow from the production environment monitoring system to this system is performed manually introducing human errors, information delay, etc.

Figure 3 shows the solution that was eventually chosen. It is based upon Internet technologies, including TCP/IP Ethernet networks and browser applications. In studying the various data sources, it was determined that all of the data sources would be connected to each plant's TCP/IP Ethernet network after the PLC control system upgrade was in place. With the existing WAN in place, the networks could be



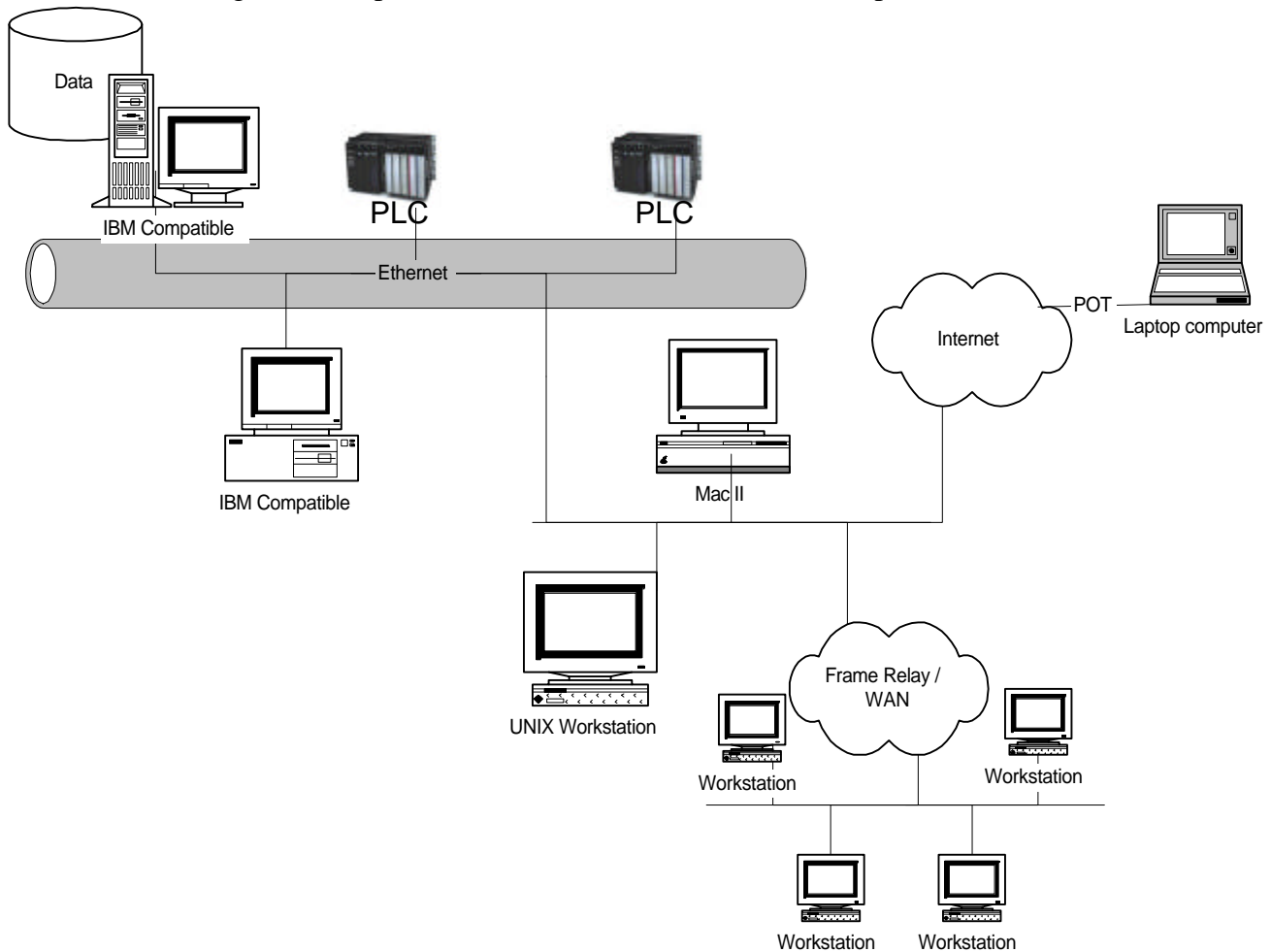
**Figure 1**



**Figure 2**

accessed from one plant to another. With some recent upgrades, all of the required PLC control systems were also on Ethernet or accessible via Ethernet gateway's to the proprietary PLC networks.

The challenge would be to find a way to bring the data from the sources together over the networks, organize it, and find a way to present it to the client in a simple, low maintenance yet flexible application. This type of problem called for a 3-tier solution and the use of a web browser for the client and web servers for the organization and presentation of the data. Since there was in existing secure, TCP/IP based network connecting all of the plants, the intranet infrastructure was in place.



**Figure 3**

The decision was made to use the web browsers that were already installed on all of the existing users desktop computers as the client presentation tier. Since different functions utilize different computer systems on their desks, using a web browser provides for the transportability and simplicity of user interface required by the design criteria. Web browsers are typically operating system independent, users already know how to utilize them thereby reducing training costs, and the end-user maintenance costs are very low with a browser since most desktop computers and workstations come with browsers pre-installed. Finally, Year 2000 issues on the client would be avoided by using a browser since Y2K compliant web browsers are available at no cost. The middle tier would handle the collection of data and serving of the data to the clients.

The design of the middle tier involved the creation of an application on a web server in each facility that would connect to the variety of data stores, the bottom tier, and organize the information for common presentation. The data stores remain in place in the lower tier in their native formats. The middle tier will serve the job of gathering and organizing data for presentation.

Because of this three-tiered design, the chosen solution would be scalable. Performance of the overall system is primarily determined by two factors. First, the amount of network bandwidth available could constrain performance. Existing networks were 10 Mb Ethernet, but upgrades to 100 Mb Ethernet, including switches where they would help optimize performance were planned as part of this upgrade. Second, the amount of processing power, memory, and disk space on the servers supporting the middle and lower tiers could become limiting factors. Since the database is in the lower tier and the user application support is in the middle tier, the system can be scaled by adding multiple web servers in the middle tier, all talking back to the database server. The database server can be scaled through additional processing power, memory, and network connections to support the added web servers in the middle tier. The key is that the servers supporting the end users can be replicated in a simple manner as the demands and numbers of the users grow.

Several key enabling technologies made the creation of the middle tier cost-effective from an implementation and long-term maintenance standpoint. These include objects, ActiveX, COM, and Visual Basic. Object oriented software technology is the most important of the enabling technologies used in creating the middle tier.

Object technologies evolved in the late 1960's from work done by Ole-Johan Dahl and Kristen Nygaard in Norway as a way to model the behavior of an experimental nuclear reactor.<sup>1</sup> The basic premise of an object is that all systems can be broken down into components or sections. Each section can be modeled and its behavior defined based on what inputs it is given and what actions are taken upon the object. Take the well-known physical world paradigm of a PLC. A simple modular PLC with a five-slot rack might have seven basic objects. These are the rack, the power supply, the CPU and four input/output interface modules. One can describe an input modules' **properties** in terms of size, number of points, color, etc. One can describe the modules' **interfaces** in terms of what type of signals the module can handle, how the module is wired, and how the module connects to the PLC rack. The **behavior** of the module can be described in terms of the **events** that occur based on what types of signals and stimuli are applied to the module. Similar types of modules, such as input modules, having common properties are grouped into **classes**.

Now move out of the physical world into the world of software and think of all the pieces of software running on a PC as components, **objects**, or "black boxes". For each object, the user can setup certain parameters, tell the software to do something, and get results. Taking to its logical conclusion, software objects can be created that model or represent behaviors of processes and machinery in the physical world.

Because of the widespread presence of Win32 technologies on the desktop, including Windows95/98, NT, and CE, developer support has blossomed for a specification for the creation of objects known as COM, or the Component Object Model. Developers have created thousands of objects that perform specific functions or provide for interfaces to other systems. Having a specification for the way these objects

---

<sup>1</sup> "Object Technology Eases Control and MES Applications", Jan S. Woien, Control Platforms, October 1997

connect to each other, behave, and are created has encouraged their development and the development of tools for their creation.

An example would be an ActiveX control to manage communications with a PLC over a standard TCP/IP Ethernet network, shown in Figure 4. An ActiveX control is a type of software object created using the COM specifications. An ActiveX control has a defined set of interfaces. Figure 4 shows the interfaces for this example control. ActiveX controls can be referenced by and used in a variety of applications built upon the COM.

Taking this concept further, because objects by their nature are event driven, when properly implemented, they can yield highly efficient solutions. A good object solution will advise other objects when something has changed or has happened that it needs to report on. This is a classic “reporting by exception” model. As previously mentioned, the desire for this application was to create a middle tier that would help organize and present data. The event driven nature of software objects supports that goal.

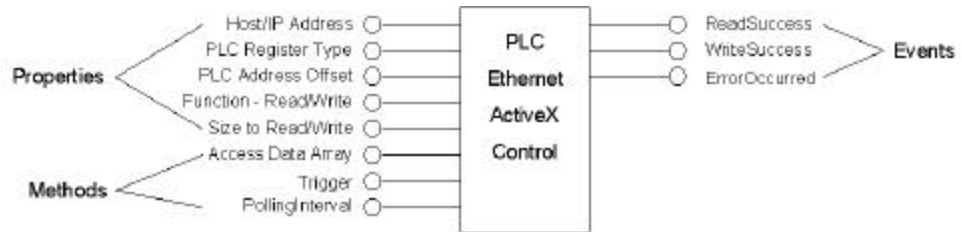


Figure 4

For this application, an off-the-shelf ActiveX control was used for handling of the PLC interface over TCP/IP Ethernet. This control is essentially the core communication engine enabling low-level devices (PLCs) to pass information to the middle tier using the prevalent Ethernet backbone. The control has several pertinent parameters that enable us define the communications with the PLCs as shown in Figure 5.

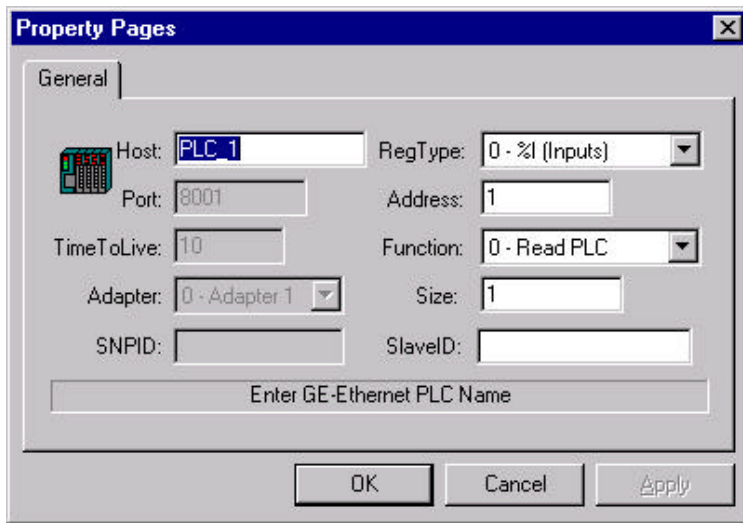


Figure 5

First, the Host property is a common or friendly name we assign to a given PLC. This assignment of IP address to friendly name is done with a configuration utility provided with the control. Next, we must specify the type of data we want, and whether to read or write the data.

When it came time to assemble the objects and create the middle tier, the Visual Basic technology was used as the “glue” to connect the objects. Although COM objects can be used with any object-oriented language on a COM enabled operating system, a very popular language today is Visual Basic (or VB), with over 3,000,000 users worldwide<sup>2</sup>. Because the web servers chosen for this application utilized a web

<sup>2</sup> "No Oxymoron with VBA", Neil Charney and Mike Gilbert in an interview with Software Strategies, January 1998, page 44.

server application capable of utilizing COM objects, ActiveX controls, and a subset of Visual Basic called VBScript as their default tools, Visual Basic was chosen as the tool for creating the middle tier.

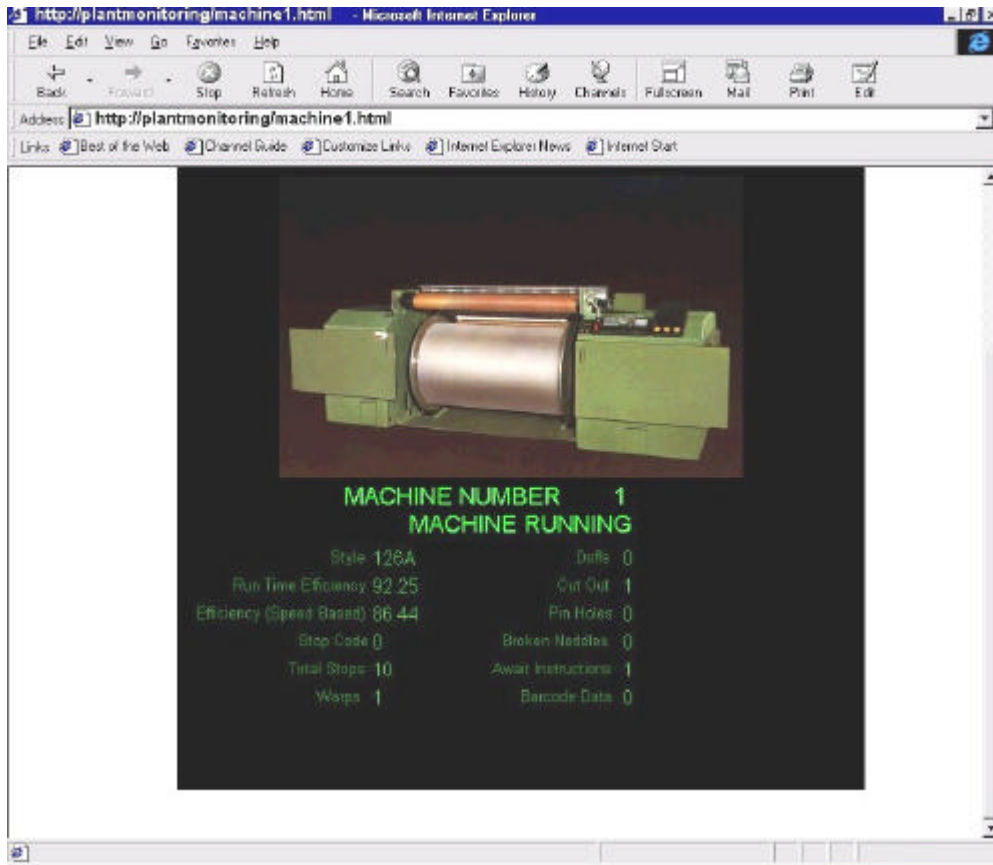


Figure 6

In our monitoring application we were only interested in reading data but the ability to have bi-directional communications was available by setting the Function property. Now we have enough information to communicate. Built on top of this object is a monitoring object that displays the values obtained from the PLCs. This is the deepest level of the application and is illustrated in Figure 6. This object retrieves data from an individual machine on the

production floor. It interfaces to other objects in the middle tier (server application) to propagate the data to the database. The two Active X controls for Ethernet PLC communications at the bottom of this screen are not visible at runtime. These controls gather the data displayed on this page. To update the current values the user simply clicks on the machine graphic. This becomes evident at runtime as the hyperlink cursor appears when the mouse pointer is over the area containing the machine graphic.

Taking one step out in the interface we see a 2D array of machines that represent a group of machines on the factory floor. This is illustrated in Figure 7. The machines appear on this page as they are actually arranged on the plant floor. The machine groupings are displayed along with the style of product running. Seeing this arrangement of machines in a web browser window, it is intuitive to get to the machine level. The user simply places the cursor over the desired machine graphic, and clicks the mouse. The important point here is that the interface makes navigation in the application obvious. Again the desired result is to leverage the ability of the users' Internet navigation skills.

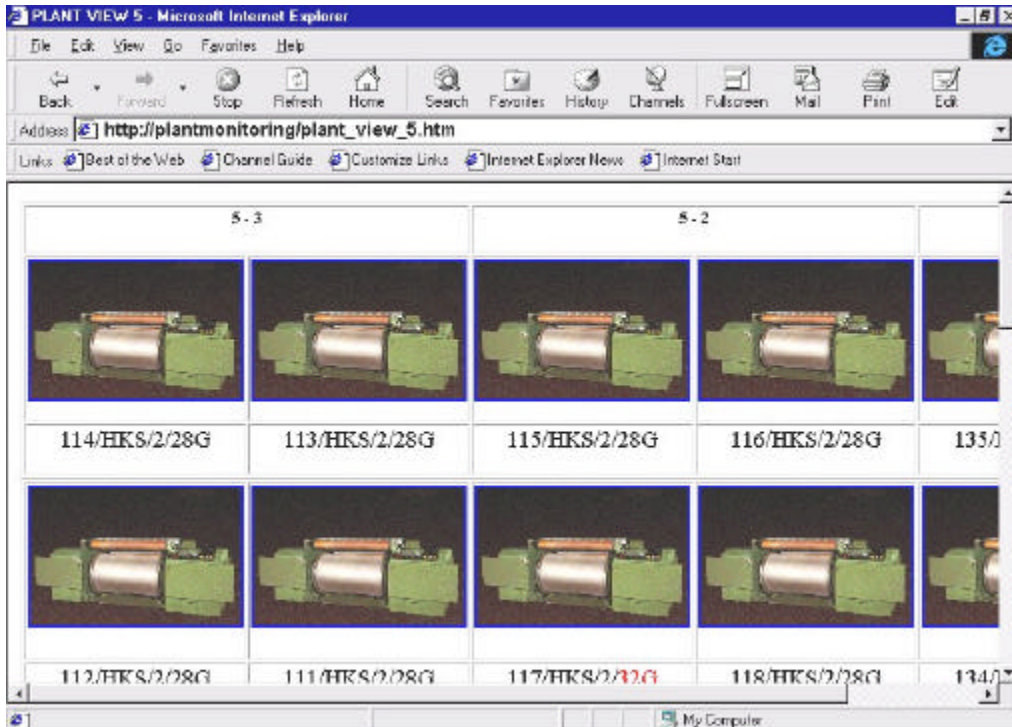


Figure 7

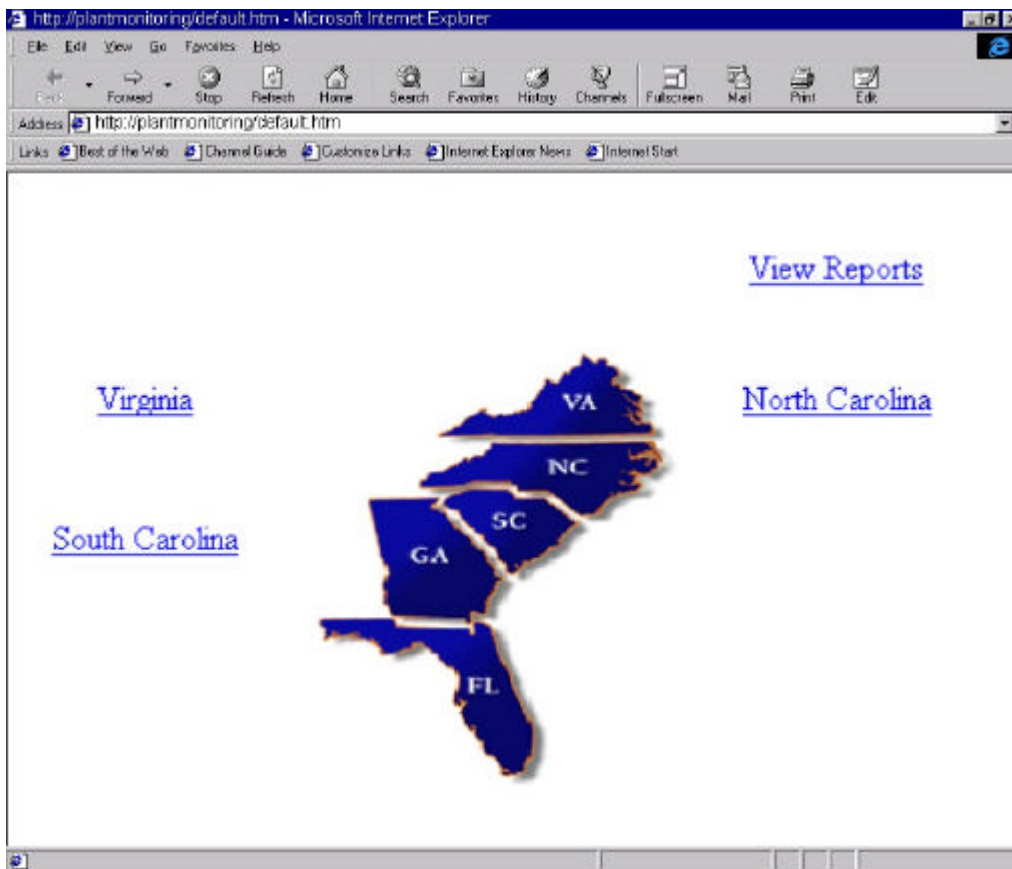


Figure 8

If we back out one more level we can see the highest level of the interface illustrated in Figure 8. This level of the application allows the user to navigate through the different plants that are in different locations. The basis for this level of the application is the existing Ethernet WAN. The middle tier takes advantage of the customer's existing inter-plant connectivity. In light of this, security is not an issue because the web server is behind the firewall requiring that views have access to the company intranet. No one outside the firewall would be allowed access to the data, but a properly set up dial in connection would allow an executive or service person on call to dial in and access the data in the same manner as one would at any other location in the plant – all through the standard web browser interface.

A look at this highest-level page reveals a simple, self-explanatory interface with an obvious navigation direction.

The intuitiveness of the interface completely eliminates the need for rigorous training. Since the interface is common to all users, a simple plant wide demonstration of the interface should suffice for those with even a little Internet browsing experience. These factors in conjunction with the fact that there is no runtime fee associated with the browser nodes lower the total cost of ownership of the solution.

## CONCLUSIONS

The success of object-based technologies has made it possible for plant floor and front office functional staff to eliminate the gap that prevents effective sharing of information. Internet and intranet based technologies allow for the creation of 3-tier solutions that provide for transportability, portability, flexibility, and simplicity. By using industry standard browser based interfaces, the end users of this application were provided with an application that was easy to learn, easy to maintain, and had a low overall total cost of ownership. The information that is provided by this system is enabling the user to improve profitability through improved quality, improved production yields, and increased quality.