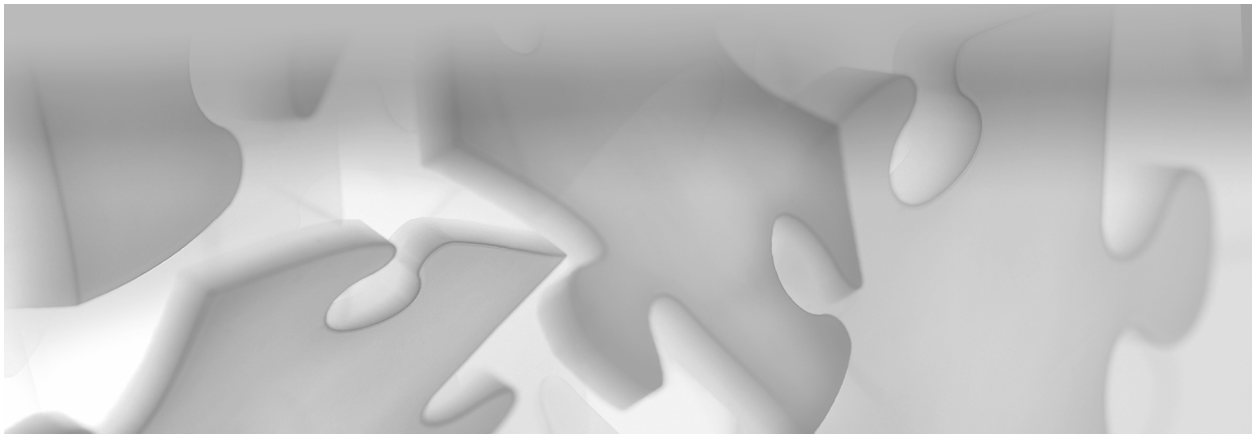


Part 1

An Introduction to SAS/IntrNet Software

SAS/IntrNet software opens SAS to the Internet, extranet, or intranet. Specifically, this software enables users to run ad-hoc reports and dynamic applications via the Web. Broadly speaking, SAS/IntrNet is divided into three areas:

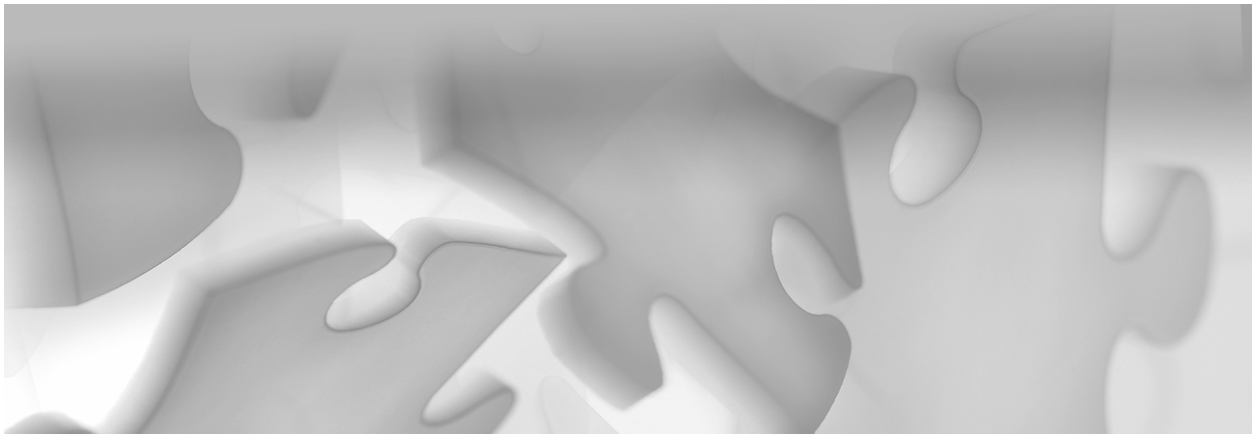
- **data services** that enable the user to make SQL-type queries to the SAS server. Users can query, update, and report data.
- **compute services** that give the user full access to the analytical capabilities of the SAS server. Users can access and use any non-visual functionality provided by the SAS server.
- **out-of-the-box applications** that use the SAS/IntrNet data and compute services to deliver access to OLAP cubes created with PROC MDDB (the MDDB Report Viewer) and to explore and report the contents of SAS catalogs and libraries (the Xplore Sample Application).

Organizations often use a diverse collection of technologies to store, manage, and report on information. Therefore, organizations should consider factors such as the following when choosing a technology for developing a Web-based information delivery system:

- user interface requirements
- data access and analysis requirements
- application performance
- costs of development, deployment, and maintenance

The Application Dispatcher component of SAS/IntrNet software has many facilities that can make the applications development task easier. This book explains many of those facilities including some that are not known to the broad population of users.

Chapter 1 gives a high-level overview of the Application Dispatcher component of SAS/IntrNet software and places the software in context with related Web technologies.



Chapter 1

Overview of SAS/IntrNet and Related Technologies

1.1 Is the Application Dispatcher a Good Fit?

One of the primary design goals for the SAS/IntrNet Application Dispatcher was to mask the complexity of other technologies, such as CGI and HTML, which are needed to implement SAS Web technologies.

The Application Dispatcher enables organizations to deploy a Web application without requiring the development staff to have extensive knowledge of CGI or HTML. The advent of ODS makes this even easier. Of course, some knowledge of these technologies is advantageous in implementing a Web solution. However, the application and the data usage factors should be the driving force behind determining which technologies to use.

There are many different factors and criteria that should be considered when deciding which components are most appropriate for a given set of development needs. In addition to the requirements and the skills available for the current application, other Web-based applications and skills available within the organization should be considered. Applications that meet the following criteria are likely to be good candidates for the Application Dispatcher:

- Existing SAS programs produce output and reports that should be made available to some or all users in an organization over the Web.
- The application is an extension of existing SAS applications.
- The reports are generated from dynamic data. Note that if data is reasonably static (e.g., only change monthly), then Web publishing the data via ODS might suffice to meet the application needs.
- Users need the ability to customize the reports (e.g., select specific subsets) to get the output they need.
- The development staff has a broad range of SAS programming skills. If there is only limited knowledge of or experience with CGI, HTML, Java, etc., then the Application Dispatcher with ODS would be a good choice.
- SAS programming expertise is available and the timeframe to deploy is a critical factor (i.e., time to train the staff in new technologies is not available).
- The application will be run on-demand to deliver specific reports.
- The data is maintained centrally (e.g., in a data mart or data warehouse). Data which is inherently local or specific to a particular user is typically not a good candidate for application or report distribution.
- The data files are potentially large and the reporting process reduces the size of the data (i.e., downloading a final customized report as HTML, PDF, or RTF is much faster than downloading the data for the user to process locally).
- The data already exists (or is accessible) as SAS data sets.
- There is a need to reduce paper reports by allowing users to access customized reports that meet their specific requirements.

1.2 Components of SAS/IntrNet Software

SAS/IntrNet contains a variety of components that can be used individually or in combination to address specific needs of a Web-enabled information delivery solution. If multiple components are used, each component provides services that best meet the requirements. The power of SAS/IntrNet lies in the remarkable range of its components. They are discussed in the following subsections.

For more information about terminology, refer to Section 1.6.

1.2.1 The Application Dispatcher

The Application Dispatcher has two required components: the Application Server and the Application Broker. It provides both compute (i.e., running a SAS program) and data services (i.e., running a SAS program that does a SQL query using PROC SQL). It also includes out-of-the-box applications (e.g., Xplore). The Application Server and the Application Broker are discussed in detail in Chapters 3 and 4.

The Application Dispatcher provides a CGI gateway (namely, the Application Broker) between a Web browser and SAS. This gateway lets programmers build dynamic applications that can access the power of SAS (the Application Server) from a Web browser. The gateway also provides the capability to run any SAS program that can be executed in batch mode. The programmers need to work only with the details of what results (e.g., reports) the SAS program is to generate. All of the details relating to CGI (e.g., communicating the parameter values from the HTML page and returning the results to the user's Web browser) are handled by the Application Broker and require no knowledge of CGI development or programming. The Application Broker passes all incoming requests to the Application Server, which can execute almost any batch SAS program. The Application Dispatcher allows the deployment of SAS programs to multiple users (whether or not they have SAS installed on their machines).

The SAS/IntrNet Application Dispatcher also includes an optional Load Manager that can be used to intelligently route and handle a large volume of incoming requests. Based on its configuration, the Load Manager can start and stop remote Application Servers based on demand, and route requests across multiple back-end application servers, regardless of the operating environment.

The Application Broker, Load Manager, and Application Server are described in more detail in Chapters 3 and 4.

1.2.2 SAS Design-Time Controls

SAS Design-Time Controls (DTCs) make it easy to create SAS/IntrNet applications without prior knowledge of HTML or SAS programming. DTCs are add-in components for What You See Is What You Get (WYSIWYG) HTML editors that support the SAS Design-Time Controls standard defined by Microsoft, including Microsoft FrontPage, Macromedia Drumbeat 2000, SoftQuad HoTMetaL PRO 5, webAF, and more. DTCs are ActiveX controls that run inside the HTML editor and generate text based on the user's input. In this way, a favorite visual HTML editor can be used to build SAS/IntrNet reports and applications via a point-and-click interface.

With SAS DTCs, static reports can be created and published to a Web server. Dynamic reports can be created using JSP and ASP technologies, retrieving and displaying the latest information when a user selects the page from the Web browser. Both static and dynamic DTC reports can be

viewed by any browser (e.g., Internet Explorer, Netscape, Firefox, etc.). Using the DTCs does not require any prior experience with developing ASP or JSP pages.

1.2.3 Xplore Sample Web Application

SAS/IntrNet includes a sample application called Xplore which exposes the SAS data environment to the Web. Using the Application Dispatcher, Xplore can dynamically access a variety of SAS data and file types for reporting, generating graphics, and performing drill-down analysis on the Web. The Xplore sample application is shipped with the SAS source and can provide a valuable resource of techniques and tools.

1.2.4 htmSQL

htmSQL is a CGI gateway to SAS and provides data services that, in turn, provide access to SAS data from a Web browser, enabling the building of dynamic SQL queries. htmSQL consists of a CGI program that resides on the Web server and can be used to access and update a SAS data set (including Read access through views to an external DBMS). The developer provides an input file, e.g., a .hsql file (note that the extension does not have to be hsql), containing SQL statements embedded in HTML, and htmSQL submits the statements to a SAS data server (either SAS/SHARE or the SAS Scalable Performance Data Server). htmSQL retrieves and formats the results according to the HTML embedded in the .hsql file. htmSQL can be used to create sophisticated, dynamic applications that let users manipulate data to address their specific information requirements. htmSQL can also be used to generate any other type of markup language such as Extensible Markup Language (XML), Compact HTML (cHTML), Handheld Device Markup Language (HDML), and Wireless Markup Language (WML).

1.2.5 SAS/CONNECT Driver for Java

The SAS/CONNECT driver for Java provides compute services. It is a set of Java classes that can be used to create Java applets, JSP, and Java applications that communicate with SAS software on a server, thus taking advantage of remote SAS computing resources. The driver provides functionality that is similar to what a SAS client can do with SAS/CONNECT, except that the functionality is available to any Java program and does not require SAS to be locally installed. Programs that use the SAS/CONNECT driver for Java can start a SAS session, connect to that session, create data sets, access existing SAS data, run SAS programs to analyze SAS data, and retrieve the results. The Java components in SAS AppDev Studio can be used to create applications that use the SAS/CONNECT driver for Java without requiring extensive knowledge of Java.

1.2.6 SAS/SHARE Driver for JDBC

The SAS/SHARE driver for Java Database Connectivity (JDBC) provides data services and is a set of Java classes that can be used to create Java applets, JSP, and Java applications that communicate with a SAS data server (either SAS/SHARE or the SAS Scalable Performance Data Server) via SQL queries. The Java programs can let a user view and update data by submitting SQL queries and statements through a direct connection to the SAS server. The Java components in SAS AppDev Studio can be used to create applications that use the SAS/SHARE driver for JDBC without requiring extensive knowledge of Java.

1.2.7 Tunnel Feature

The SAS/IntrNet tunnel feature employs HTTP tunneling to allow Java applets to communicate with remote systems via a CGI program running on the Web server. As a security measure, Java specifications dictate that an applet cannot make network connections to a machine other than the machine from which it was downloaded, unless Java 2 (or later) is being used and explicitly allows this. When deploying Java applets, this security measure might force the use of a server configuration that is less than ideal, because that server would have to be installed on the same machine as the Web server. In addition, many firewalls prohibit applets from communicating beyond the firewall, a restriction that can further reduce server configuration options. The tunnel feature addresses both of these configuration problems.

The tunnel feature, with Java applets written using the Java components in SAS/IntrNet or the Java components in SAS AppDev Studio, can be used to eliminate the restriction on where the SAS server runs in relation to a Web server and firewall. JSP applications do not have any of these restrictions because they run as native operating environment applications and have access to all resources.

1.3 Other SAS Web Components and Technologies

Other SAS resources provide facilities and tools that provide critical capabilities for any Web application that is based on SAS. Selected resources are briefly described in this section.

1.3.1 The Output Delivery System

Starting with Version 7 of SAS, the Output Delivery System (ODS), which is part of Base SAS software, allows the direct creation of Internet content, including HTML files. In general, ODS is a method of delivering output in a variety of formats and of making the formatted output easy to access. Instead of taking the output from a procedure and creating HTML via post-processing, ODS creates the HTML during the initial creation of the procedure output. This method is more efficient and provides many new possibilities for report layout. Through the addition of a few simple lines of code, developers can convert a standard program into one that creates HTML output. It's not necessary to know HTML or modify existing SAS code.

In addition to HTML, ODS supports many other types of output formats, including Adobe Acrobat Portable Document Format (PDF), Rich Text Format (RTF) for use in Microsoft Word and other word processing programs, CSV for use by Microsoft Excel and other spreadsheet tools, XML, and WML for use with WAP-enabled hand-held devices. ODS also produces GIF-, JPEG-, ActiveX- and Java-based graphics in conjunction with SAS/GRAPH software. This capability lets developers create a wide variety of content that can be delivered over the Web with little or no extra effort. Consider the following example that uses ODS to produce the desired output and automatically passes the results back to the user's browser:

```
ODS HTML BODY=_webout;
proc print data=sashelp.class;
run;
ODS HTML CLOSE;
```

ODS, which is the subject of a number of papers and SAS Press books, should be the default mechanism for producing the Internet content output required of most Web applications.

1.3.2 The Web Publishing Tools and Related Macro Tools

For users running SAS Version 6 and later, the Web publishing tools are a collection of tools that generate Internet content output. While these tools continue to exist in SAS software, the tasks they perform are usually easier to accomplish with ODS. However, there are occasions where their use might be a better fit. Also note that these tools are not part of SAS/IntrNet and may be used in any SAS program.

The Web publishing tools include the following:

- The HTML formatting tools, included with Base SAS software, are a collection of macros that format SAS data sets and procedure output into HTML pages that can be shared with Web users:
  - **The Output Formatter (%out2htm)**, which reformats output from any SAS procedure as an HTML file.
  - **The Data Set Formatter (%ds2htm)**, which displays SAS data sets as HTML 3.*x* tables. The Data Set Formatter supports WHERE clauses, BY-group processing, and other data presentation capabilities.
  - **The Tabulate Formatter (%tab2htm)**, which reformats the output from the TABULATE procedure into HTML 3.*x* tables.
- Thin-client graphics are part of SAS/GRAPH software and are specialized, lightweight, visual Java applets or ActiveX controls which do not establish a persistent connection to a server. Instead, they receive all the information they need from applet parameters. The following tools generate the applet and ActiveX HTML calls along with the custom data values that are used as input to the visual component:
  - **The GraphApplet HTML Generator (%ds2graf)** produces graphs and charts from SAS data. It generates HTML that uses the GraphApplet or SAS/GRAPH Control for ActiveX to display and manipulate the graphics.
  - **The MetaView HTML Generator (%meta2htm)** produces an HTML file that includes metadata through which users can view SAS/GRAPH output in a Java applet. To view the output as graphics, you must use the MetaViewApplet.
  - **The RangeView HTML Generator (%ds2csf)** produces an HTML file that displays a critical success factor (CSF) from a SAS data set using the RangeViewApplet.
  - **The TreeView HTML Generator (%ds2tree)** produces an HTML file that displays a hierarchical tree from a SAS data set using the TreeViewApplet.
  - **The Constellation Chart HTML Generator (ds2const)** produces an HTML file that displays data from a SAS data set as a constellation chart using the ConstellationChartApplet.

The following are two other macro tools from SAS:

- The **DS2CSV macro** takes as its input a SAS data set and creates a comma-separated value file. Optionally, the hexadecimal code for the separator character can be specified to create other types of output file (e.g., a tab-separated values file). Prior to SAS 9.1, the DS2CSV macro was shipped only with SAS/IntrNet for use with the Application Dispatcher. Beginning with SAS 9.1, the macro is available for use in any SAS program.
- The **FILESRV macro** is used to serve a wide variety of external files and catalog entries, including those that are not defined to a Web server. The FILESRV program enables the files that are to be served to be controlled (including what HTTP and MIME headers are served with the file). This macro uses an authorization data set to determine which files can be served. The mechanism for making this decision is RX function pattern matching. This data set is used to set patterns (or masks) for the files that can be served. If a requested file does not match one of these patterns, then it is not served and an error message is issued.

1.3.3 SAS AppDev Studio, a SAS Applications Development Environment

SAS AppDev Studio is a development suite that supports various ways to develop, deploy, and maintain information delivery applications. SAS AppDev Studio offers users the choice of building Java-based applications on the client or on the server, flexible CGI and HTML applications, ASP applications, or traditional full-client applications, with ease and efficiency, all from within one stand-alone development environment. In addition, this software provides access to the proven, extensive server capabilities of SAS for enterprise applications development.

SAS AppDev Studio includes the two Java components webAF and webEIS. webAF is a complete Java Integrated Development Environment (IDE) tailored for the rapid creation of Java applications, applets, servlets, and JSPs that use the power of SAS. SAS AppDev Studio also includes for development purposes multiple SAS products, including SAS/IntrNet, that are required to develop Web-based applications.

1.3.4 SAS®9 Business Intelligence Platform

The SAS®9 Business Intelligence Platform is a framework for providing business intelligence services that use SAS Stored Processes. Like SAS/IntrNet Application Dispatcher programs, SAS Stored Processes are traditional SAS language programs that use SAS DATA steps, procedures, macros, and the SAS Component Language (SCL) to deliver powerful SAS capabilities to client applications across an enterprise. SAS Stored Processes are stored and managed from a central server and must be described by metadata, which includes information about where each process is stored, how it is executed, and what output it generates.

However, SAS Stored Processes are slightly different from traditional SAS programs in that they are executed by a SAS Stored Process Server. SAS Stored Processes are conceptually similar to SAS/IntrNet Application Dispatcher programs. In fact, many Application Dispatcher programs can be executed as is, or with only slight modifications, by the SAS Stored Process Server. This is true because both use macro variables to communicate the user's parameters to the program or process being executed, and both use the same reserved fileref, _WEBOUT, to stream the results back to the user's browser.

Many of the examples and techniques presented in this book for the Application Dispatcher can also be used with the SAS Stored Process Server.

Because almost any valid batch SAS program can be converted to a stored process, the stored process capability delivers access to the full range of SAS analytics. Stored processes can be used to accomplish tasks such as these:

- generate report content or analytic results for Web browsers, Microsoft Excel, Web services, and other clients.
- generate report content or analytic results for SAS solutions and custom applications.
- implement Data Warehouse Extract-Transform-Load (ETL) job flows and other data transformation procedures.
- implement Web applications or functional components of Web applications.
- create or update data on the SAS server.
- run any batch SAS process.

Note, however, that there are currently differences between the two that will affect any decision to switch from the Application Dispatcher to the SAS Stored Process Server. Consider, for example, the following facts:

- SAS Stored Processes can be invoked from a broader range of clients (e.g., the Web, Microsoft Office, and SAS Enterprise Guide).
- Application Dispatcher programs can be external .sas files, catalog source entries, compiled SCL entries, and compiled macros. Stored processes may invoke any of these, but the process itself must be a .sas file.

Stored process results can be made available to users in a number of ways, including, but not limited to the following:

- **SAS Stored Process Web Application** is a Java Web application that executes stored processes and returns results to a Web browser. For Web-based applications it performs services comparable to what the Application Broker component of the Application Dispatcher provides.
- **SAS Information Delivery Portal** provides integrated Web access to SAS reports, stored processes, SAS Information Maps, and publish-and-subscribe channels. If the SAS Information Delivery Portal is installed, stored processes can be made available for execution from the portal without the need for additional programming.
- **SAS Add-In for Microsoft Office** is a component object model (COM) add-in that extends Microsoft Office by enabling dynamic execution of stored processes with the output embedded in Microsoft Word documents and Microsoft Excel spreadsheets. Additionally, within Excel the SAS Add-In for Microsoft Office can be used to access and view SAS data sources or any data source that is available from a SAS server.
- **SAS Enterprise Guide** is a client application for the Microsoft Windows environment that provides a guided interface to SAS for business analysts, statisticians, and programmers. It can be used to create as well as access SAS Stored Processes.
- **SAS Web Report Studio** is a Web-based interface that provides access to query and reporting capabilities.

1.3.4.1 The SAS Stored Process Web Application

For Web-based applications similar to what can be done with the Application Dispatcher, stored processes can be accessed from a browser using the SAS Stored Process Web Application (part of

the SAS Web Infrastructure Kit). The user interacts with a Web interface and clicks a link to retrieve results, or the user can interact with an HTML form to specify any required parameters.

Other similarities to the Application Dispatcher include these:

- The user interacts with a Web browser interface. In most cases the user enters parameters into an HTML form and clicks **Submit**. The request is then sent to the SAS Stored Process Web Application, a Java Web application located on the Web application server that can execute stored processes that are on a SAS Stored Process Server and return results to a Web browser.
- To write a Web application that uses SAS Stored Processes, only SAS and HTML programming skills are needed; no other programming (e.g., Java, CGI) is required. Thus, applications can be Web-enabled very quickly.

1.3.4.2 SAS Information Delivery Portal

The SAS Information Delivery Portal disseminates information in a targeted, secure, and personalized way. It allows companies to distribute SAS products and solutions to users within an enterprise and to external customers, vendors, and partners through a secure, Web-based client interface. It also supports various languages.

The SAS Information Delivery Portal is a J2EE, N-tier Web application that is deployed on Java application servers across various operating environments. Built using the SAS Intelligence Platform, it shares infrastructures and metadata with other SAS solutions and technologies.

Any stored process that is registered in the SAS Management Console (a component of the SAS®9 BI tool set) can be readily searched and run using the SAS Information Delivery Portal. After selecting the search link, the user can search specifically for SAS Stored Processes.

1.3.4.3 SAS Add-In for Microsoft Office

SAS Add-In for Microsoft Office enables users to connect to a SAS Server in order to do several things:

- provide users access to SAS analytics from within Microsoft Office. This is accomplished by providing a self-sufficient, automated way to populate and maintain Microsoft Office reports with information from corporate data stores as well as from the results of analytic analysis.
- make enterprise data from multiple platforms easily available to business users. Users can be given broad access to all relevant data sources, even data repositories larger than those allowed by Excel. Users can access and switch between any enterprise data source and control the amount of data that is loaded into their Excel applications.

1.3.4.4 SAS Enterprise Guide

SAS Enterprise Guide is a Microsoft Windows interface that allows a user to access a SAS server (either a locally installed or remote SAS server). Here are some features of SAS Enterprise Guide:

- The software has a graphical user interface that allows access to SAS. A process flow diagram facility lets users organize, view, and maintain their projects visually. It includes reporting, graphical, and analytical tasks, as well as more than 60 wizard-based tasks. Advanced users can create programs that produce tables and charts that can be easily embedded in other SAS applications and easily and securely shared.

- Users can perform data management functions allowing them to visually access any data types supported by SAS and native Windows data types. Users can create, update, subset, and join tables themselves using a graphical query builder.
- Data stored as OLAP (i.e., multidimensional data stores) can be presented allowing users to navigate through multidimensional data, add topic-specific business calculations, and extract information from multidimensional sources for further analysis.

1.3.4.5 SAS Web Report Studio

SAS Web Report Studio has a Web-based interface that provides access to query and reporting capabilities on the Web. It is targeted to non-technical business users, providing them with self-service access to corporate data. It is completely Web-based and does not require the installation of any local software on the end-user's PC.

A Report wizard enables novice or casual users to quickly create basic queries and reports. Users can search and choose the information they need. As users' needs evolve, more sophisticated layout and query capabilities are available from a Report Builder interface. Business users can create folders for organizing reports. Existing reports can be edited, moved, copied, deleted, or renamed. Any user with the appropriate access can search for relevant reports in public or private folders.

Users can print or save a PDF version of their report. Graphs and tables can be exported to Microsoft Excel as images. Alternatively, the data presented in a graph or table can be exported as tab-delimited text.

1.4 Industry Components

By definition, the Web is an open environment. The use of other technologies is an integral part of Web-based solutions. Of course, Web servers and Web browsers are a standard component. However, there are other technologies that should be considered in using SAS/IntrNet and other SAS Web technologies.

1.4.1 Scripting Languages

Scripting languages, such as JavaScript and VBScript, are lightweight interpreted programming languages with simple object-oriented capabilities. Scripting language programs operate at the client side (i.e., the Web browser), and they allow executable content to be included in Web pages. Executable content within the Web page means that Web pages can include dynamic programs that interact with the user, control the Web browser, and create the HTML content dynamically.

If development staff is well versed in VBScript and if the user community is using Microsoft Internet Explorer as the Web browser, then VBScript is a viable choice for a scripting language. Some will say that JavaScript is more widely used and supported, and should be the scripting language of choice. One of the more important capabilities of JavaScript is its ability to define code fragments that are to be executed when a specific event occurs. JavaScript code can be used to perform functions such as the following:

- Validating input (e.g., check for required fields, numeric values, etc.) before submitting a CGI request.
- Submitting a CGI request automatically when an item is selected from a list box (an HTML select tag).
- Updating a user's choices based on previous selections without requiring a round-trip to the server.
- Reading and writing properties of, and invoking methods of, Java applets and plug-ins.

For example, general purpose JavaScript functions are used in the Xplore sample application. As one example, consider the JavaScript functions defined in the page titled Select Variables to Process. This is the page users receive when they click on a non-summary SAS data set. Those functions are used to update both a text display and a cumulative list of variables in drill-down order. Figures 1.1 through 1.3 demonstrate this. Figure 1.1 shows the initial display. Figure 1.2 shows the results after the user selects the **Region** check box; note how the text display was also updated. Then Figure 1.3 shows the display after the user selects another check box (**Subsidiary**).

Figure 1.1 JavaScript Used by Xplore to Provide Dynamic Content—Initial Display

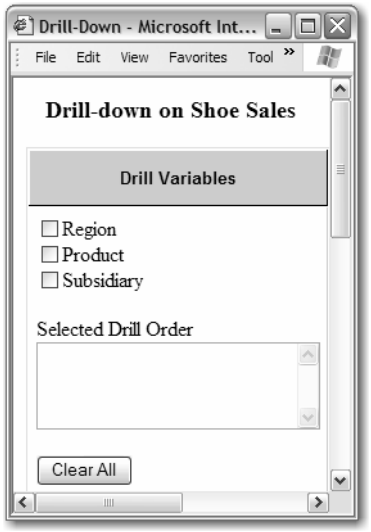


Figure 1.2 JavaScript Used by Xplore to Provide Dynamic Content—User Selects Region

Figure 1.3 JavaScript Used by Xplore to Provide Dynamic Content—User Selects Subsidiary

1.4.2 Dynamic HTML

As discussed in Section 1.6, Dynamic HTML (DHTML) provides facilities to make HTML-based reports dynamic in terms of interactivity and visual effects in response to user actions without requiring a round-trip to a server to regenerate HTML. It was created to overcome the interaction limitations of standard HTML. Unfortunately, different browsers support different versions of DHTML. However, in an environment (e.g., an intranet) where browser vendor and versions can be assumed, DHTML can provide an easy and powerful mechanism to extend the functionality of a Web application.

For example, with Microsoft DHTML, a simple (and generic) JavaScript function can be used to expand and collapse simple lists that were built using the HTML <LI ..> tag. Xplore has an Internet Explorer implementation of the SAS library viewer that is automatically used whenever the Web browser is Internet Explorer 4 or later. While the browser is loading, the content of all libraries and catalogs is defined in the initial HTML. However, the DHTML capability in Internet Explorer is used to expand and collapse libraries and catalogs without requiring any additional server processing. Figure 1.4 shows an initial folder list produced by Xplore. If the user clicks a folder (e.g., SASUSER), Figure 1.5 shows the resulting display.

Figure 1.4 Library List Produced by Xplore

Figure 1.5 SASUSER Library Expanded

1.4.3 Web Services

Web services is a framework in which applications that provide specific content in response to a request are packaged as services that can be made available over the Web. All communication between services (i.e., the request for a service, and the results of running a service) is handled via XML files. Web services have been embraced by a number of vendors (e.g., the Microsoft .NET framework provides extensive support and use of the Web services model). Web services are discussed in more detail in Chapter 21, which discusses how the Application Dispatcher can be packaged to provide Web services.

1.5 Component-Based Architectures

The future of Web applications development consists of a component-based architecture in which developers can mix and match a variety of programming languages and concepts, such as the SAS and industry components mentioned previously, using whichever is most appropriate for each application.

Consider an example where Pareto charts are generated on historical shop-floor data. Using ODS, the code is relatively straightforward to generate one or more HTML pages with the appropriate graphics. Management decides that such charts need to be produced so that the staff monitoring the shop floor can see the most recent data being collected. But there is a wrinkle: the requirement is that the graphs need to be refreshed every ten minutes (and, of course, it is discovered later that the users may want to change that refresh frequency) with no user intervention. At first glance it is not clear how to do this using SAS or SAS/IntrNet. However, the solution is quite simple: HTML constructs exist to force a page to be refreshed on a scheduled basis. The solution is to include an

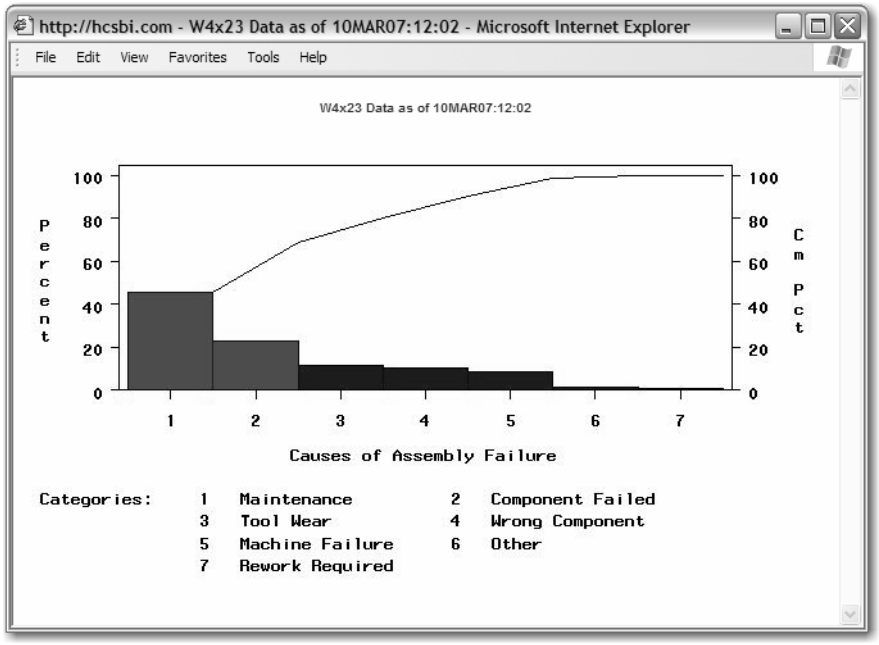
appropriate META tag in the HEAD section of the HTML. The following code (where the LINE data to select and the refresh rate are passed in as parameters) addresses the requirement:

```
goptions device=jpeg cback='white';
ods html body=_webout (title="&Line Data as of &sysdate:&systime")
         metatext="http-equiv=""refresh"" content=""&refresh"""
         path=&_tmpcat (url=&_replay)
         rs=none nogtitle nogfootnote
         style = sasweb
         ;
proc pareto data=qcdata.failure ;
  title .h=1 "&line Data as of &sysdate:&systime";
  where line = "&line";
  vbar DEFECT / freq = COUNT weight = WEIGHT chigh(2)=red
                cframe=gray cbars=blue nohlleg;
run;
ods html close;
```

The results of running this program using SAS®9 are shown in Figure 1.6. The only additions to the code to cause the refresh and to update the display with the information about the time are highlighted in bold. By combining SAS technology components (e.g., the Application Dispatcher, ODS, the PARETO procedure from SAS/QC) with an industry component (e.g., the META tag), the complete requirement has been addressed.

For demonstrations and examples described in this section, go to the sample environment for this book and select **Chapter 1**.

Figure 1.6 Pareto Chart—Automatically Refreshed with No Additional User Input Needed



1.6 Terminology

Web technology has had a dramatic impact on information delivery and the applications development process. Today's technologies provide many benefits over the client/server model because it is no longer necessary to distribute and install applications on each desktop; information is accessible using a Web browser. The Web, as the main delivery mechanism, provides the infrastructure to ease the deployment of applications so that a client can request data or compute services.

Web-enabled information delivery systems can be described in terms of the function they serve, how they work, and the types of technology they use. Here are the broad categories of how Web technologies can be used:

- **Web publishing** is the creation of static reports, which are made available via a Web server. Any user with a Web browser can access the information. The generated reports can be produced in a variety of ways. Scheduled batch jobs are an example. The only constraint is that the files containing the reports must be written in valid Internet content form. HTML for text-based files and GIF or JPEG for graphics files are some examples of valid Internet content form.
- **Report distribution** means that reports can be customized for specific user needs. Simple queries constructed on the clients (usually a Web browser) are passed through the Web server to a data repository, and then formatted as a report to be viewed on the client.
- **Application distribution** means that requests for decision support services can be initiated on the client (e.g. via a Web browser), and then executed by an application server with the results available to be viewed by the client.
- **Data services** enable the creation of Web-based applications that let the user view and query data from a Web browser. These applications are implemented using Web-enabled SQL. At the user's request, dynamic queries can be sent to a server.
- **Compute services** provide the ability to not only query or view data, but to actually process the data on the back-end server, thus making it possible to analyze the data, create custom reports with interactive graphics, carry out OLAP analysis, and interact with data in a data warehouse all via a user's browser.

SAS/IntrNet data and compute services can be accessed via the Common Gateway Interface or Java technology.

Here are definitions of common technology terms.

- **CGI (Common Gateway Interface)** is a programming interface that allows a Web server to communicate with an external program. Typically, CGI programs are small, written in a script or a high-level language, and reside on the Web server to act as the interface between a Web browser and a content server (providing data and compute services). When a Web browser accesses a Uniform Resource Locator (URL) for a CGI program, the Web server executes the CGI program. Usually, the CGI program invokes a session with a database or an application server, which processes the request from the client and returns the result to the Web browser via the Web server. Strictly CGI-based applications need repeated interaction with a server or servers in order to provide an interactive interface for the user.

- **Dynamic HTML (DHTML)** provides facilities to make HTML-based reports dynamic in terms of interactivity and visual effects in response to user actions, without requiring a round-trip to a server to regenerate HTML.
- **JavaScript** is a lightweight, interpreted programming language with simple object-oriented capabilities. JavaScript operates on the client side (i.e., the Web browser) and enables executable content in Web pages.
- **Java** is an object-oriented programming language originally developed and promoted by Sun Microsystems. It now has broad support across a variety of platforms. This language is expressly designed for use in the distributed environment of the Internet. Java enables the creation of two types of applications relevant to the Web:
  - Perhaps best known are **Java applets**. When deployed, an applet contains two parts: an HTML page, which contains a reference to the applet, and a set of Java classes, which contains the program logic. The HTML page specifies which Java applet is to be invoked. (In fact, a single HTML page can include multiple applet references.) When a Web server returns an HTML page to a Web browser that contains a reference to an applet, the Web browser reads the applet information and requests the applet classes from the Web server. After these classes are downloaded, the applet is executed, based on input from a user. For example, the applet can establish a connection to a database or application server, submit requests for processing, and receive and display the results. Because the Java applet is executing locally on the client side, it can provide a much richer interactive environment than a CGI program.
  - **Java Server Pages (JSP)** allow the execution of Java code on the Web server. A JSP page contains HTML interspersed with Java code that is executed when the page is requested to produce the desired dynamic content. This process reduces the need to download Java classes to the local machine, reduces the need to ensure that the clients all support the necessary version of Java (a major problem in the Java applet space), and speeds up execution.
- **JavaBeans** is a component architecture framework for Java. This component defines a software Application Program Interface (API) that lets developers write reusable components once and run them anywhere a Java virtual machine is available. These components can be used in either Java applets or Java Server Pages.
- **ActiveX controls** are an implementation for Windows only that takes advantage of the Microsoft Windows implementation of COM technologies to provide interactivity and interoperability with other types of COM components and services.
- **Active Server Pages (ASP)** can be compared to JSPs in that ASPs are programs that execute on the Web server and return Internet content to the client. ASP is a Microsoft standard and is available on Microsoft Web servers.
- **.NET** is the Microsoft Web-enablement architecture. .NET facilities are available or planned for virtually every Microsoft application.
- **Web services** provides a mechanism that cross vendor, application, hardware and operating system boundaries. Web services uses XML as the transport mechanism for defining both the request as well as the delivery of results (e.g., the data).