

Brio[®] Enterprise: An Architecture for Web-Based Business Intelligence

A technical discussion of Brio's Web architecture, focusing on the Brio OnDemand Server and the Brio Web clients, Brio.Insight and Brio.Quickview.

Architecture
January 1999



© 1999 Brio Technology, Inc. All rights reserved. Brio is a registered trademark and Brio Enterprise and Adaptive Reports are trademarks of Brio Technology.

Brio Technology, Inc. • 3460 West Bayshore Rd. • Palo Alto, CA 94303 • (650) 856-8000 • www.brio.com

Brio Web Architecture	2
EXECUTIVE SUMMARY	3
Audience.....	3
COMPUTING INFRASTRUCTURE	4
ONDEMAND SERVER SOFTWARE COMPONENTS	6
End user	6
Web server	9
OnDemand Server	11
Brio Enterprise Server Administrator	15
Database server	16
COMMUNICATION PROTOCOLS	17
HTTP and HTTPS.....	18
HTTP and Brio Web Clients.....	18
File system read and write.....	19
Brio calls over TCP/IP sockets	19
Database connectivity API.....	20
SEQUENCES OF EVENTS	21
Viewing the logon page	21
Client-side logon and Adaptive Report list.....	21
Selecting a report.....	23
Query over the Web.....	25
Zero Administration.....	26
CONCLUSION	28
APPENDICES	29
Appendix A: System Requirements	29
Appendix B: Zero Administration Components.....	32
Appendix C: About Brio Enterprise 5.5	34
INDEX	35

LIST OF TABLES

Table 1: Summary of typical computing infrastructure.	5
Table 2: End-user components.....	6
Table 3 : Web server files.	9
Table 4: Core OnDemand Server software components.....	11
Table 5: Zero Administration Specifications	33

EXECUTIVE SUMMARY

When we talk about Brio's Web architecture, what we're really discussing is how Brio Enterprise products work within a *Web environment*. More than just things or concepts, an environment influences what you can and cannot do. For example, physical environments determine that you can't snow-ski in Jamaica, and even if you could, you wouldn't want to water-ski in the Arctic. Computer environments (client/server or Web) are not much different. The way in which software, hardware, and communication protocols are developed and integrated—or architected—determines what you can and cannot do with them.

One of the characteristics of a Web environment is that applications are typically small. Since they run on a central server as CGI modules or Java servlets along with many other applications, or automatically download to the Web browser as plug-ins, JavaScripts or Java applets, Web applications need to be small and efficient in order to reduce network traffic and keep performance reasonable. For this reason, most business intelligence products designed for Web environments are often less functional and have a minimal user interface when compared to their client/server counterparts.

Brio Enterprise's architecture, however, uniquely takes advantage of technology typical to Web environments, such as self-installations and freedom from database connectivity APIs, without sacrificing the robust functionality, rich interface, or performance we often take for granted in client/server products. We also maintained the ease-of-use and file compatibility in our Web products that have always been a distinguishing hallmark of Brio's client/server products. The end result is that you can do just about anything in a Web environment as you can within a client/server environment. But because these are *Web* products, you can easily and securely deploy to more users, which ultimately affects your bottom line as more people in your organization benefit from the enhanced knowledge and power that business intelligence brings.

In particular, Brio Enterprise supports three features that are essential for efficient, distributed business intelligence. This white paper explains in detail how Brio's Web products work together to enable these important features:

- Query over the Web, using graphical drag-and-drop methods.
- Zero Administration deployment and maintenance of functionally rich business intelligence applications.
- Report level security, using Adaptive Reports, to avoid multiple server installations.

Audience

This white paper is written for DBAs, Webmasters, and anyone else with a high degree of technical expertise. It assumes that the reader is familiar with Web, database, and network terms, such as HTTP, RDBMS, proxy server, and the like. Though it assumes a basic understanding of Brio's product line, its purpose is to educate you on how Brio Enterprise works specifically within an intranet, and offers tips on how to avoid difficulties when using Brio Enterprise in Web configurations that are less often used for business intelligence.

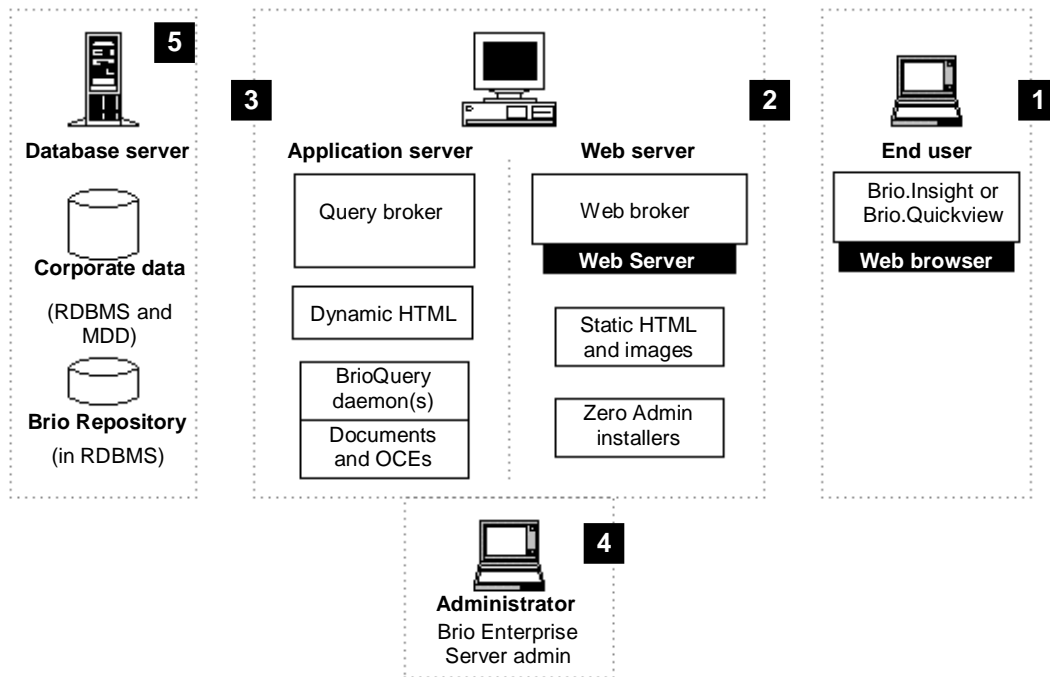
COMPUTING INFRASTRUCTURE

Most companies that pursue business intelligence already have a variety of hardware and software in place that supports Brio's distributed Web architecture. At a minimum, you usually have at least one database server, and users generally have Web browsers installed on their computers. You may also have a Web server that supports corporate users within an intranet or extranet.

To allow corporate users easy access to reports and information stored in the database, Brio adds a Web application server, called the OnDemand Server. The OnDemand Server works on behalf of the Brio Web clients to deliver authorized reports and data refreshes whenever a client requests new information. Unlike many Web-based business intelligence products, which scale back functionality so that all the application logic can run on a single Web server, Brio distributes processing among client computers and servers, as shown in the diagram that follows. Brio Enterprise software also runs on a wide variety of platforms, allowing you to take advantage of UNIX and Macintosh computers as well as your Microsoft Windows computers.

By taking advantage of the processing power available on both clients and servers, Brio Enterprise creates a balanced, responsive, and highly functional business intelligence environment. And, since the application logic is distributed, clients can continue to work off-line with saved reports and data, even if a server for some reason becomes unavailable.

Though you can install all OnDemand Server components on a single server for testing or demonstration purposes, the components should be distributed between client computers (computers 1 and 4, below) and two or three servers (computers 2, 3, and 5) for optimal scalability and reliability.



The following table summarizes the purpose of each computer introduced in the preceding diagram.

Table 1: Summary of typical computing infrastructure.

<i>Computer type</i>	<i>Technologies used</i>	<i>Purpose</i>	<i>Notes</i>
1. End User	Web browser with Brio plug-ins (compiled in C++).	Allows individuals to view, analyze, and create reports and multidimensional analyses. Brio plug-ins adapt their functionality level, depending upon the user's privileges for a particular report.	JavaScript and Cookie acceptance must be enabled in the Web browser for Zero Administration and session tracking. These settings are enabled by default.
2. Web server	JavaScript, HTML, and CGI/NSAPI/ISAPI.	Provides the entry point and communications link between the clients' computers and the OnDemand Server.	Brio recommends that you install all OnDemand Server software (computers 2, 3, and 4 in this table), on the Web server for best performance and easiest administration. Regardless of your configuration, images, some HTML documents, and the Web broker must always reside on the Web server.
3. Application server	Combination of 100% pure Java server software, compiled Brio applications (written in C++), HTML, and JavaScript.	Runs the core OnDemand Server software that manages client security clearance, report retrieval, and requests for data from the database. Also stores some OnDemand Server configuration settings in ".ini" files.	Though Brio recommends installing the core OnDemand Server components on the Web server, you can install them on its own application server. This option frees up resources on the Web server and enables load balancing.
4. Administrator	Brio-specific compiled C++ code.	Configures and maintains the OnDemand Server settings and the Brio Repository.	After the initial configuration, the BES Administrator can be installed on a separate computer for remote administration, or it can remain on the OnDemand Server computer (see points 2 and 3, above) itself.
5. Database server	Relational and multidimensional database software.	Stores corporate data in a data warehouse, data mart, or other databases. Relational databases are also used for the Brio Repository, which stores OnDemand Server configuration and security information.	The Brio Repository table schema is described in detail in the documentation that accompanies the Brio Enterprise Server.

ONDEMAND SERVER SOFTWARE COMPONENTS

This section describes each of the OnDemand Server components that run on the different computers.

End user

All an end user needs to get started with Web-based business intelligence is a Web browser, between 2-4 MB of free hard disk space*, and a URL, user name, and password provided by the BES Administrator. The OnDemand Server's Zero-Administration feature takes care of installing and maintaining the most recent versions of the Brio Web clients, Brio.Insight or Brio.Quickview, relieving IT and end users from the tedious aspects of software maintenance.

Table 2: End-user components.

<i>Brio Web client component</i>	<i>Technologies used</i>	<i>Purpose</i>	<i>Notes</i>
Web browser	Microsoft Internet Explorer and Netscape Navigator/Communicator (see Appendix D for specific versions).	Provides the fundamental user interface and most communication between the Web client and the Web server.	JavaScript and Cookie acceptance must be enabled in the Web browser for Zero Administration and session tracking. These settings are enabled by default.
Brio.Insight or	Internet Explorer or Netscape plug-in for Windows browsers; Helper applications for all other platforms.	Integrated ad-hoc query, multidimensional analysis, report viewing, and report creation.	Brio.Insight adapts between five functional levels. Its default level is Analyze (see Adaptive Reporting™ on page 8).
Brio.Quickview	Internet Explorer or Netscape plug-in for Windows browsers; Helper applications for all other platforms.	Report viewing and data refresh.	Brio.Quickview adapts between two functional levels. Its default level is View Only (see Adaptive Reporting on page 8).

Web browser

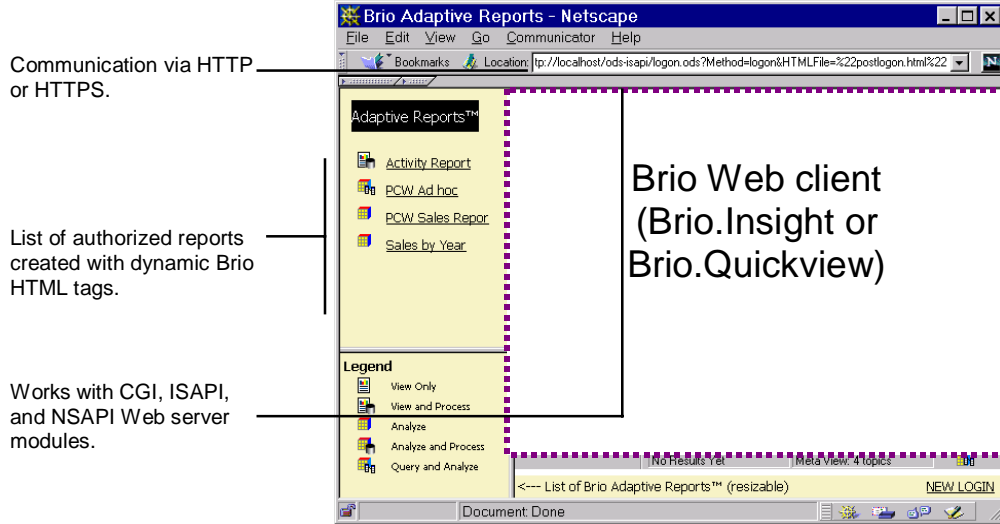
The Web browser handles all fundamental presentation and communication between the client's computer and the Web server. Using HTML, JavaScript, cookies, and standard HTTP or HTTPS, it:

- Retrieves the main logon.html and postlogon.html pages from the OnDemand Server, formats them for viewing, and populates the frames with static HTML documents and image files retrieved from the Web server. It then presents the fully formatted pages to the user.
- Transmits the user's login information and document requests to the OnDemand Server.
- Remembers the user's login information for the session using temporary "session" cookies, so users are not prompted to log in multiple times during a session.

* Disk space requirements for UNIX operating systems may be higher. Refer to Appendix A: System Requirements, for more information.

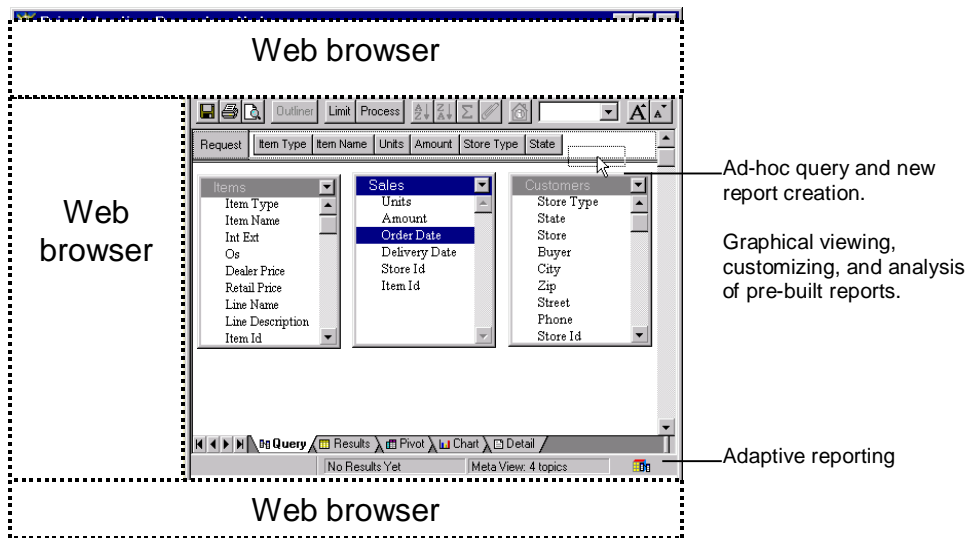
- Automates the installation and maintenance of Brio's Web client software by processing the Zero Administration JavaScript commands, and maintains proper version control with persistent cookies, the plug-in version information itself, or the Netscape registry.

The screen below shows that built-in Web browser features are used for many OnDemand Server features.



Brio Web clients

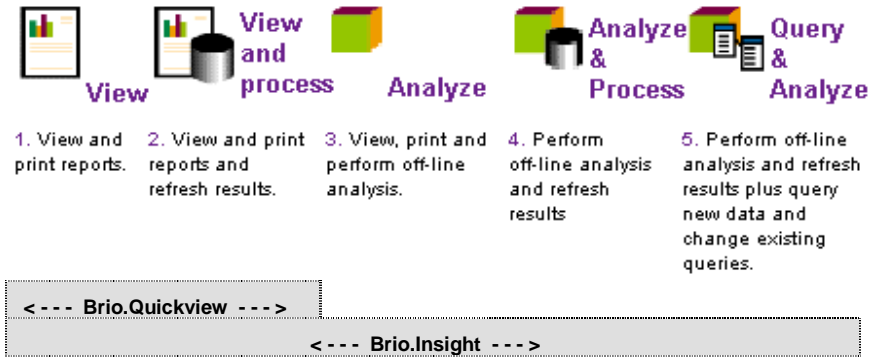
The Brio Web clients, Brio.Insight and Brio.Quickview, are Brio DLLs that “plug in” to Windows-based Web browsers, or run as “Helper apps” for Macintosh and UNIX clients. On their own, they allow corporate users to create, analyze, or simply view reports. When used with the OnDemand Server, they become much more powerful, allowing dynamic data refreshes and unique Adaptive Reporting. And unlike any other Web product available, Brio.Insight provides a graphical, drag-and-drop interface where users can make their own queries against a variety of databases (shown below).



The remainder of this section explains in greater detail how Adaptive Reporting, Query over the Web, and off-line analysis are activated in the Brio Web clients when used with the OnDemand Server.

Adaptive Reporting

Fundamentally, Adaptive Reporting is a security mechanism that authorizes users for limited, extensive, or no access to reports, depending upon their role in the company and the nature of the data. The Brio Enterprise Server (BES) Administrator sets up default user privileges, which can be overridden for individual reports. For example, the Marketing group may be granted full analysis and query capabilities (5) for data pertaining to lead generation, but may have view only (1) or no access at all to corporate financial data. As shown below, Brio.Quickview is a report viewer that can adapt between the first two modes, while Brio.Insight is capable of full report building, interactive analysis, and ad hoc querying.



BES Administrator stores Adaptive Report privileges for users in the Brio Repository.

When Brio.Insight or Brio.Quickview opens a document or report, it reads a token that the OnDemand Server inserts into the document itself (refer to “Query broker” on page 13 for more information). This token instructs the Brio Web client to dynamically “adapt” its user interface, access to toolbars, menus, and other features, to one of the five modes indicated above. This Adaptive Report approach to business intelligence also saves IT from the trouble of setting up, deploying, and maintaining different server products and security privileges for reporting, querying, and analysis, and saves users from the trouble of learning multiple products.

Query Over the Web

In contrast to traditional query and reporting tools such as BrioQuery, which use client/server APIs to connect directly with a database, the Brio Web clients retrieve data indirectly through the Web server and OnDemand Server. When the user clicks the Process button, Brio.Insight or Brio.Quickview opens its own HTTP connection directly with the Web server, strips the Brio document of all unnecessary information, and sends a shell of the Brio document to the OnDemand Server for processing over an HTTP data stream. This document shell includes the query, any limits, and the name of the Brio OCE that the OnDemand Server should use (Table 4 on page 11 lists OCEs and their function). When the query is done, the OnDemand Server returns a highly compressed data set to Brio.Insight or Brio.Quickview, at which point the user can modify or create new reports, analyze data, or create subsequent queries as their Adaptive Report privileges allow.

Off-line support

In addition to embedding Adaptive Report tokens in Brio documents, the OnDemand Server also inserts “homing” information, which includes its own IP host and port information. This information makes it easy for users to work with a Brio document when completely disconnected from the Web. When it’s time to refresh

the data, the Brio Web client reads the homing information, uses this IP address and host information to locate the original OnDemand Server, and prompts the user to log in again to ensure that the data remains secure.

Web server

The Web server is the gateway for Web users to access the OnDemand Server. Three OnDemand Server components must reside on the Web server: the Web broker, static HTML files, and Zero Administration Client installers.

Table 3 : Web server files.

<i>Web server component</i>	<i>Technologies used</i>	<i>Purpose</i>	<i>Notes</i>
Web broker	CGI, ISAPI, or NSAPI.	Communications module that transfers client requests and information between the Web server and the Query broker.	If using NSAPI or ISAPI, the Web broker is loaded by the Web server at startup as a DLL, where it runs inside the Web server's address space; if using CGI, the Web broker is loaded dynamically with each request in its own memory space.
Static HTML	HTML documents and GIF image files. Also includes the JavaScript code for Zero Administration.	User interface elements for the logon, Adaptive Report list, and work area pages.	The static HTML and GIF files populate frames defined by the dynamic HTML pages that reside on the OnDemand Server.
Zero Administration Web client installers	JAR, CAB, BinHex, or Self-extracting Windows archive files.	Self-installing Web clients.	JAR and CAB files include self-running installers and digital certificates. Other file types include installers only.

Web broker

The Web broker is a small communications module that transfers requests and information between the Web server, and the Query broker. This communication module is available in three forms: a CGI for any type of Web server; ISAPI for use with Microsoft IIS Web servers; and NSAPI for use with Netscape Web servers.

Since the Web broker is small and only transfers data, there is no *appreciable* performance gain by selecting ISAPI or NSAPI over CGI. For this reason, Brio generally recommends using the CGI module for the initial installation, as it is easier to install and configure. If you find that the OnDemand Server is very heavily used, you may want to switch to the ISAPI or NSAPI module to minimize resource consumption.

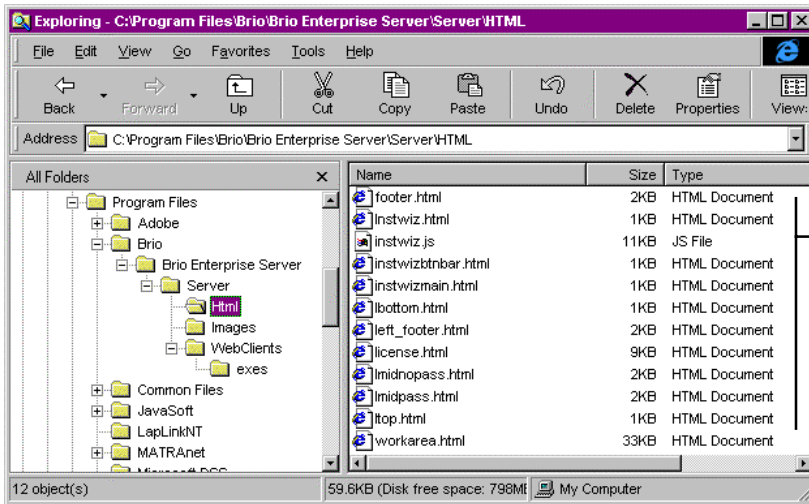
Static HTML

The images and instructional text that users see when they log in reside as HTML and GIF files on the Web

server. Since these files do not need to be processed or modified in any way, they are called “static” files.

The static HTML files also include the JavaScript code for the Zero Administration clients. This code evaluates what operating system, Web browser, and Brio Web client (if any) exist on the user’s computer, and assists the user by retrieving the most suitable Web client for the user. (See “Appendix B: Zero Administration Components” for more information.)

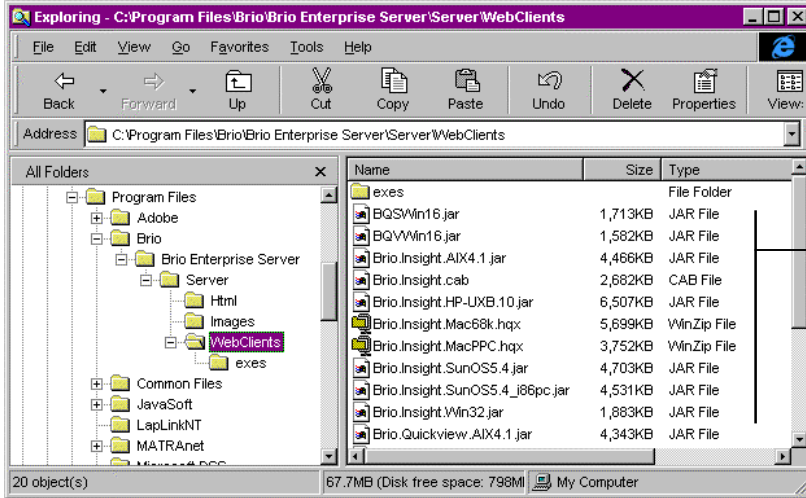
The administrator (with the help of a Webmaster, if necessary) can change the appearance and behavior of the OnDemand Server by modifying the static HTML documents, shown below. Refer to Brio’s “Customizing the OnDemand Server” technical note at <http://www.brio.com/support/bq5500.html> for instructions and requirements.



Static HTML documents that are stored on the Web server.

Zero Administration Client Installers

If a user does not have a Brio Web client, or it is outdated, the Zero Administration Client code that runs from the static HTML pages (above) retrieves the correct Web client from the Zero Administration directory on the Web server. The Web server stores CAB, JAR, and other archive file types, supporting a wide variety of computing platforms. If using a Windows Web browser, the JAR and CAB files include the installer, plus a digital certificate for extra security. Refer to “Appendix B: Zero Administration Components” on page 32 for more information.



The Zero Administration Web client installers are available in a variety of file formats to support multiple platforms.

OnDemand Server

The OnDemand Server components that perform user authentication, document retrieval, and document processing must all reside together on the same computer. These software components can be installed on the Web server (with the Web broker, static HTML documents, and Zero Administration client installers), or on its own application server.

Most core OnDemand Server components are written in Java. Each Java component requires its own instance of a Java Runtime Environment (JRE), which is installed automatically with the OnDemand Server for Windows NT and Solaris. The other components are compiled C++ applications, as summarized in the table below.

Table 4: Core OnDemand Server software components.

<i>OnDemand Server component</i>	<i>Technologies used</i>	<i>Purpose</i>	<i>Notes</i>
OnDemand Server Windows NT service or UNIX shell script	Windows NT Service or UNIX shell script.	Starts the Query broker components.	At least two JREs and the Process Factory run simultaneously when the OnDemand Server is active. One or more BrioQuery daemons may run, as well.
Query broker · Manager · Node · Process Factory	The Manager and Node are 100% pure Java. Managers and Nodes run within their own Java Runtime Environment (JRE). The Process Factory is a small C++ component.	The Query broker is the core of the OnDemand Server, handling all user requests, and forwarding them as appropriate to the database.	The Query broker Manager runs on port 5500 (or whichever port was selected by the Administrator). The Nodes cycle through the next five sequential port numbers, for example, 5501-5505.

BrioQuery daemon	Brio-specific compiled C++ code.	<p>Queries a variety of data sources on behalf of Web clients.</p> <p>It is also used to retrieve security clearance and user privileges from the Brio Repository, which is stored in a relational database.</p>	<p>Connects to a variety of relational (RDBMS) or multidimensional (MDD) databases.</p> <p>The BES Administrator can specify that several (or zero) BrioQuery daemons pre-launch when the OnDemand Server starts.</p>
Open Catalog Extensions (OCE)	Binary Brio files.	Defines which database connectivity API and other login information the BrioQuery daemon uses to connect to specific data sources.	The BES Administrator or client/server BrioQuery user creates OCEs.
Database connectivity API	C++ connectivity API software. Examples: ODBC, OLE DB, SQL*Net, Open Client, etc.	Client/server connectivity software used by the BrioQuery daemon to connect to a data source.	Compatible connectivity API software must be installed on the database server, the OnDemand Server, and BES Administrator computers.
Brio Documents	Documents created by BrioQuery client/server tools.	Shared queries, reports, and analyses.	<p>Brio documents can be stored in the Documents directory on the OnDemand Server (see screen shot, below), using the server's file system, or in the Brio Repository on the database.</p> <p>The Broadcast Server can be used to run reports on a scheduled basis, and save them in the Documents folder for shared use.</p>
Dynamic HTML	HTML with custom Brio tags and JavaScript.	<p>These HTML files include frames that define the display structure for the static HTML documents that reside on the Web server.</p> <p>Also includes custom Brio tags that are dynamically processed by the Query broker to initiate users' authentication and to populate the list of Adaptive Reports.</p>	<p>Includes logon.html, postlogon.html, and reportlist.html.</p> <p>These pages can be modified. The Brio technical note, "Customizing the OnDemand Server" (http://www.brio.com/support/bq5500.html) includes instructions and requirements for changing the appearance and function of these files.</p>

Query broker

The Query broker is the core of the OnDemand Server software. It consists of three tightly integrated components: a Manager, a Node, and a Process Factory. Together, these three components route Web clients' database requests to BrioQuery daemons for processing. The Query broker also initiates calls to the database to authenticate users and to resolve special OnDemand Server tags in HTML files, such as placeholders for the list of Adaptive Reports that a particular user may use.

Manager - The Manager controls the workflow of requests, performs queue management, and asks the Process factory for new Nodes on a regular basis. The Manager is a Java server application that runs in its own JRE at TCP port 5500 (or whichever port is selected for the OnDemand Server). The Manager cycles through the five port numbers after its own (for example, 5501-5505) and assigns them to Nodes as they are spawned. One Manager will always be running when the OnDemand Server is running.

The Manager keeps track of the port number for the active Node (5501, for example). As client requests come in from the Web broker, the Manager refers the Web broker to the active Node. The Manager does this for each request. After a pre-determined limit (the default is 200 consecutive requests), the Manager asks the Process factory for a new Node at the next sequential port number (5502), and then sends subsequent Web broker requests to the new Node. It then manages a shutdown of the original Node, thus cleaning up resources and increasing overall reliability.

Node – A Node works as the middleman between the Web broker and BrioQuery daemons. A Node is a Java server application that, like the Manager, runs in its own JRE. Nodes can be thought of as temporary workers, as they handle a pre-determined number of transactions (by default they each handle 200 consecutive requests), then shut down while another Agent takes over new requests.

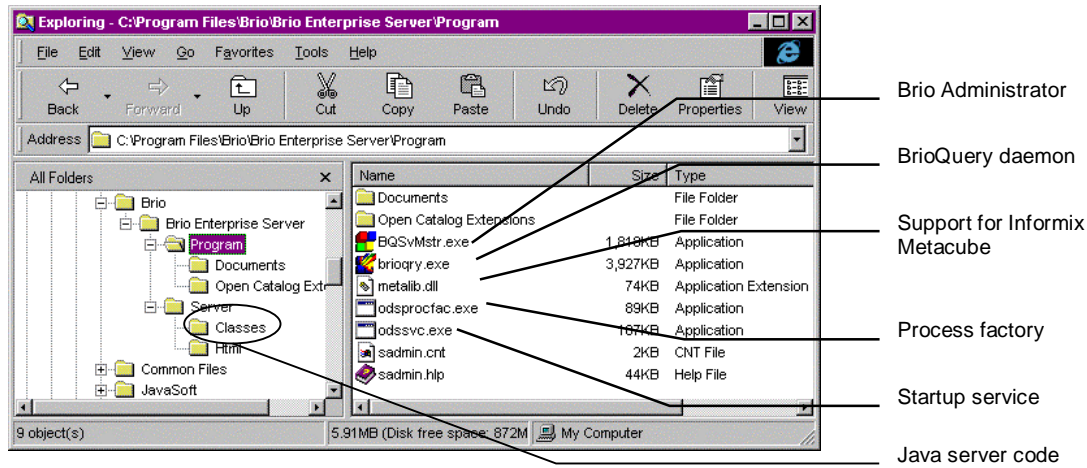
Process factory - The Process factory is a continuously running server process that simply spawns, or launches, a Node or BrioQuery daemon. Do not confuse the name "process factory" with "processing" queries. The name refers only to launching Query broker components *as background server processes*. User requests do not pass through the Process factory.

BrioQuery daemon

BrioQuery daemons handle all database access, including Web clients' queries and inquiries for data from the Brio Repository, such as user authentication and Adaptive Report privileges.

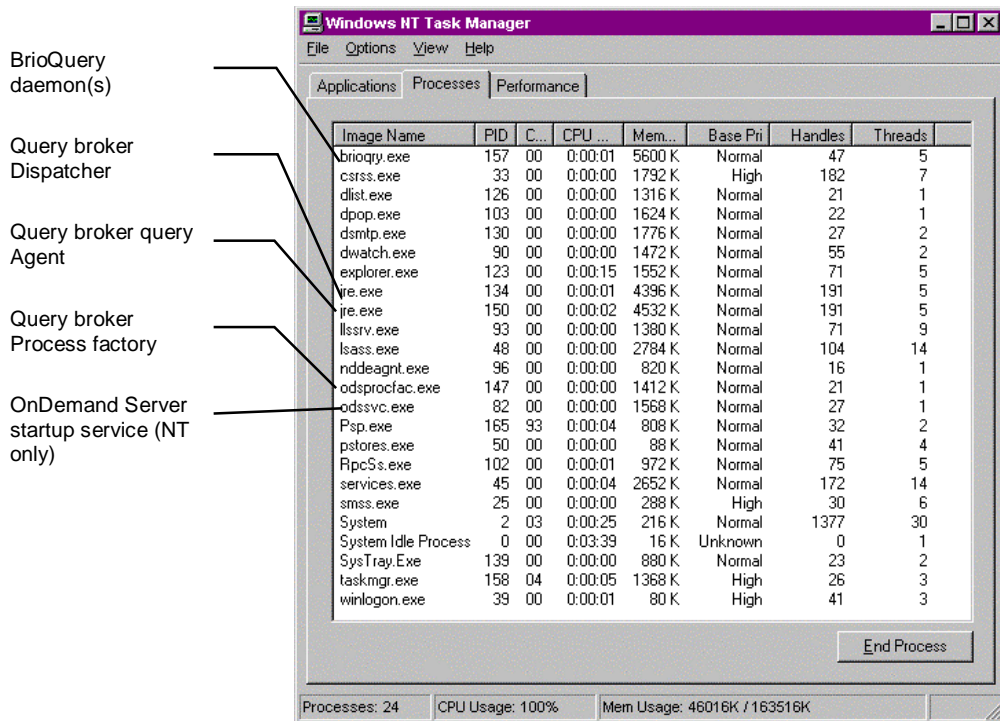
The BES Administrator can specify that zero, one, or more than one BrioQuery daemon run when the OnDemand Server starts. Each BrioQuery daemon handles one request at a time. If all BrioQuery daemons are busy running jobs, new requests are queued up at a Node until a currently running job completes.

The following screen shot calls out the various core applications used by the OnDemand Server.



OnDemand Server Startup service

The Windows NT Startup Service (or Unix shell script) makes sure that the Query broker components are running on the OnDemand Server computer. When the OnDemand Server service starts, it spawns the Query broker Process factory and Manager. The Manager, when it starts, asks the Process factory to spawn a Node. The Node, in turn, asks the Process factory to spawn at least one BrioQuery daemon. When the OnDemand Server is active, at least five separate processes are running, as shown below.

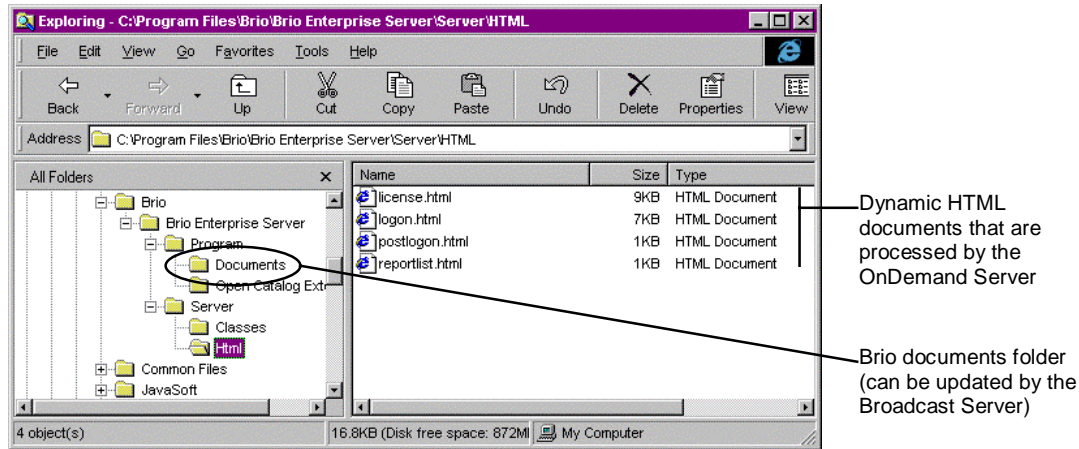


Dynamic HTML

These HTML documents contain special OnDemand Server tags in them. The OnDemand Server reads these files, interprets the Brio tags, and dynamically replaces the tags with user-specific information. Unlike the static HTML documents that reside on the Web server, these HTML documents can return different information with each user request. For example, the special OnDemand Server tag <ODSDOCListTag>, that resides in the reportlist.html file, is replaced by the Query broker with the list of files a particular user has access to.

The logon.html and postlogon.html pages also define the HTML frame structure for the user interface within the Web browser. These frames are populated by the static HTML documents (see page 9) that reside on the Web server.

The following screen shot shows both the dynamic HTML documents that are populated with data from the Brio Repository, plus the location of the Brio documents folder if the Administrator chooses to save shared documents on the server's file system.



Brio Enterprise Server Administrator

The Brio Enterprise Server (BES) Administrator is used to:

- Configure the OnDemand Server.
- Define user and group access to the OnDemand Server.
- Register documents to the OnDemand Server.
- Monitor active BrioQuery daemons.

The initial configuration must be performed locally on the OnDemand Server computer. After that, you can keep BES Administrator on the OnDemand Server computer or install it on another computer to administer one or more instances of OnDemand Server remotely.

The BES Administrator is also used to configure the Broadcast Server. When used together, the Broadcast Server and OnDemand Server are collectively referred to as the Brio Enterprise Server.

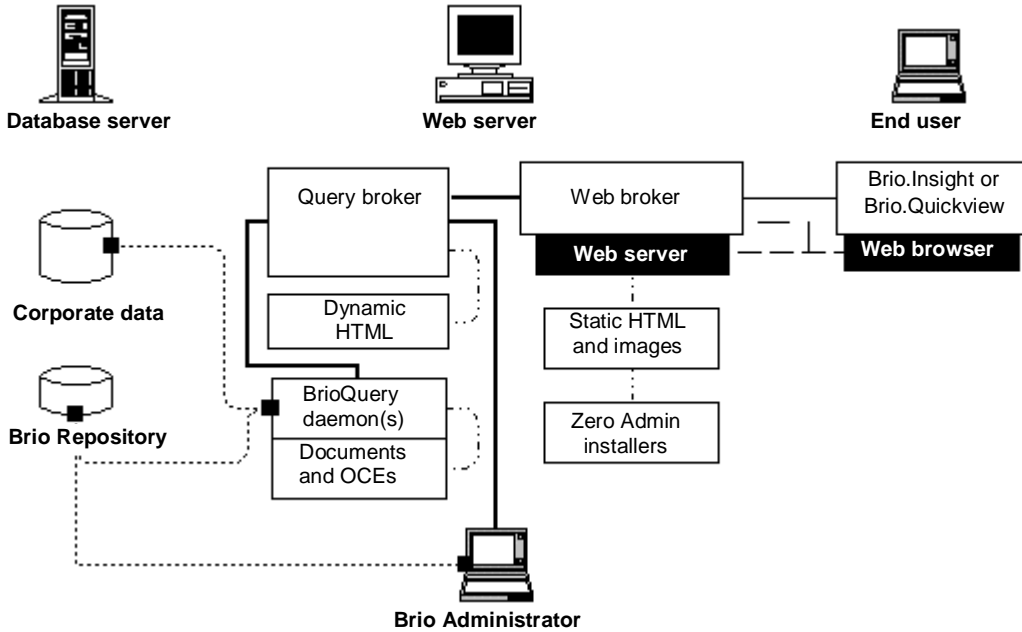
Database server

Brio recommends that enterprises organize their data stores into a data warehouse or architected data marts in order to take advantage of the clean schemas these databases provide. Regardless of your data storage choice, Brio Enterprise can access and extract data from almost any database that is accessible from the OnDemand Server computer. Refer to Appendix A: System Requirements on page 29 for a list of connectivity APIs that can be used to connect to a database.

The Brio Repository is created as a set of database tables in a relational database. It stores Brio reports and OnDemand Server configuration information, such as user and group privileges, server location, user names, and so on. When you first configure the OnDemand Server with BES Administrator, it prompts you to create Repository tables for that server. If you have multiple OnDemand Servers, you can either allow them to use their own Brio Repository, which works well for distinct user groups, or implement a load balancing scheme, whereby all OnDemand Servers use a single Brio Repository. You can learn how to do this by referring to the Brio technical note, "Load Balancing the OnDemand Server," located on the Brio Web site at <http://www.brio.com/support/bq5501.html>.

COMMUNICATION PROTOCOLS

The OnDemand Server components communicate with each other via proprietary and standard Web and client/server communication protocols on top of TCP/IP. The diagram and table that follows summarize which protocols each computer uses to communicate with other OnDemand Server components.



<i>This line:</i>	<i>Indicates this protocol:</i>	<i>And connects these components:</i>
-----	HTTP or HTTPS	Web browser to the Web broker or Web server for login, Adaptive Report lists, and initial document retrieval.
————	HTTP	Brio Web client (Brio.Insight or Brio.Quickview) to the Web broker for query processing and data refresh. <i>Note: though the connection technically is with the Web server, all communication from the Brio Web client and the Web server is directed specifically to the Web broker, which runs as a CGI, ISAPI, or NSAPI module. Therefore the HTTP line in the diagram connects the Web client to the Web broker to distinguish it from other Web server functions, such as retrieving HTML pages and images.</i>
.....	File system read/write	HTML file reads on the Web server, and Brio document reads and writes by the BrioQuery daemon on the OnDemand Server.
————	Brio proprietary API over TCP/IP sockets	Web broker (usually installed on the Web server) to the Query broker components (Manager, Node, and Process Factory) and BrioQuery daemon(s).
■-----■	Database connectivity middleware API.	BrioQuery daemon(s) and the BES Administrator to the database server. Connectivity API software, such as OLE DB, ODBC, SQL*Net, Open Client, and so on, must be installed on both sides of the connection.

Each of the communication protocols is described in greater detail in the following pages. Within each section, we provide information on three rare, but salient, communication issues:

- HTTPS support when processing queries.
- proxy support when processing queries.
- Firewalls between the Web server and the OnDemand Server computer.

HTTP and HTTPS

HTTP (hypertext transfer protocol) and HTTPS (secure hypertext transfer protocol, using SSL) are standard, built-in connection protocols used for communications between Web browsers and Web servers.

All client-side OnDemand Server actions that are run within the Web browser itself, such as the initial OnDemand Server logon, running Zero Administration scripts, presenting the list of authorized Adaptive Reports, and retrieving documents, run automatically over HTTP or HTTPS, depending upon the Web server's configuration.

HTTP and Brio Web Clients

Like a Web browser, the Brio Web clients, Brio.Insight and Brio.Quickview, has the HTTP protocol built into its plug-in code. This allows the Brio Web clients to send non-textual data streams to the Web broker, such as when submitting queries.

Once Brio.Insight or Brio.Quickview loads a document, any requests for *new or additional data into that document*, such as processing a query, performing a show values on a limit, or retrieving remarks for a topic or item, is initiated by the Brio client's *own* HTTP connection to the Web broker. As mentioned in the previous section, retrieving documents can occur over HTTPS.

This use of HTTP works very well within an intranet, which is the most common environment for business intelligence. If you are deploying over an extranet or Internet, however, the following issues need to be addressed.

Limited HTTPS support

Though most communications can occur using the Web browser's HTTP or HTTPS connection, the Brio Web clients currently do not support the HTTPS protocol. This means that if your company uses a secure SSL server *for business intelligence*, users will receive an error if they click the Process button to refresh a data set or run a new query.

If this is a concern, the first thing to consider is whether communication occurs within a secure intranet or not. If so, there usually isn't a need for an SSL server. Other considerations are that the OnDemand Server transmits passwords in encrypted format on its own, and unlike many other Web products, which send data in the clear, Brio data is packaged and compressed in a binary format that is only readable by the Brio Web clients.

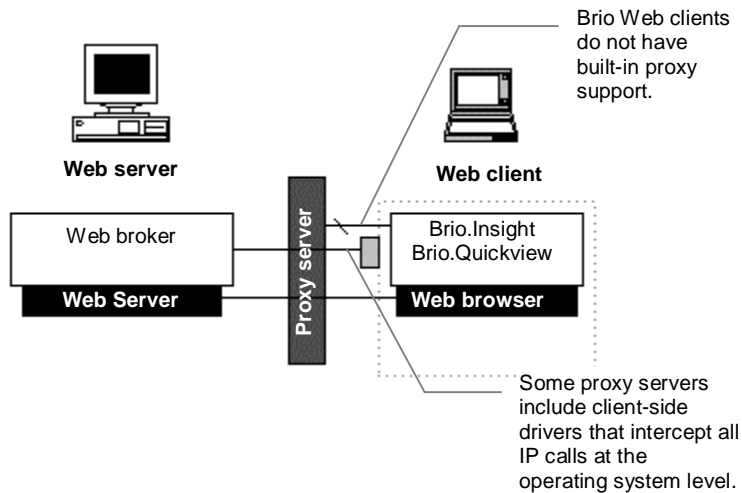
If you decide that you still need a secure Web server, you can turn off data refresh privileges (Adaptive Report modes 2, 4, and 5), and use the Broadcast Server to regularly refresh Brio documents and save them to the Brio Documents folder on the OnDemand Server computer. (See Table 4 on page 11 for a list of OnDemand Server components.) This ensures that the data saved with the reports is always fresh, which eliminates the need for end-users to process.

Full support for HTTPS communication with SSL servers is a future direction for Brio Enterprise.

Limited proxy server support

Though not dependent on HTTP per se, the plug-in's use of its own data connection might present a problem if a proxy server sits between the client and the Web server. Unlike the Web browser, the plug-in currently does not have built-in proxy server support. You can work around this if you use products such as Microsoft's proxy Server, which includes a client-side module that intercepts all IP communications at the operating system level. proxy servers are generally not required on intranets.

In the diagram below, the first line with the break in it shows that communication between the Brio Web client doesn't make it to the proxy server. The line at the bottom shows that the Web browser can be configured to recognize the proxy server, so communication can pass through the network. The line in the middle originates from a client-side proxy driver, such as Microsoft's WinSock proxy client, which intercepts all IP calls from the operating system and forwards them to the correct proxy server.



Integrated proxy server configuration support is a future direction for Brio Enterprise.

File system read and write

The OnDemand Server stores HTML files, Zero Administration client installers, and OCEs (connectivity instructions for the BrioQuery daemon) on server file systems. Brio shared documents can be stored either in the Brio Repository, which resides in a relational database, or in a server's file system.

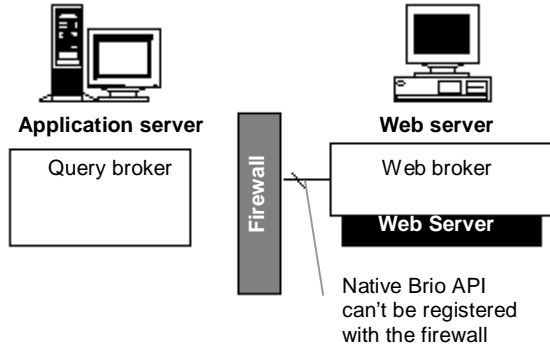
Brio calls over TCP/IP sockets

The OnDemand Server components make proprietary Brio calls over TCP/IP sockets to communicate with and send documents to each other. The Web broker, running as a CGI, ISAPI, or NSAPI module on the Web server, makes a TCP/IP socket connection to the Query broker Manager, running on the OnDemand Server computer. Communications between all Query broker components (Manager, Node, and Process factory), and the BrioQuery daemons all occur using proprietary Brio calls over standard TCP/IP sockets.

Limited firewall support

Though this is rare, some companies host their intranet Web server at an Internet Service Provider (ISP), located outside the company and beyond the bounds of the corporate firewall. In this case, establishing

communication between the Web server and the OnDemand Server might be difficult, because the Brio protocol cannot be registered with the firewall. Contact Brio Technical Support if you think this might be an issue for your company. (Note that the OnDemand Server software could optionally be installed on the Web server, so that all Brio API calls are made on the same machine and never make it to a firewall.)



Database connectivity API

A traditional client/server database connectivity API, such as OLE DB, SQL*Net, ODBC, and Open Client, is used by the BrioQuery daemon to access the database which hosts the Brio Repository, and to process client-side queries.

SEQUENCES OF EVENTS

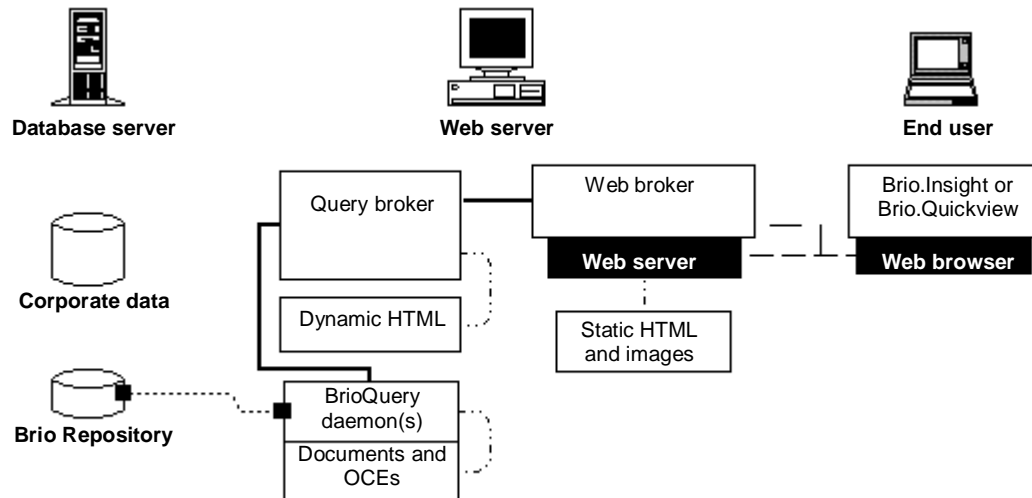
This section explains in detail exactly how the OnDemand Server components work together to bring business intelligence capabilities to the Web.

Viewing the logon page

Communication protocols: HTTP or HTTPS

The user launches the Web browser and connects to the corporate intranet.

1. The user either types a URL or clicks a hyperlink that contains the URL to the OnDemand Server's Web broker. The URL itself will indicate whether the Web broker runs as a CGI, ISAPI, or NSAPI module.
2. The Web broker forwards the logon page request to the Query broker, which may reside either on the Web server or on its own server (see Computing Infrastructure on page 4).
3. The Query broker retrieves the logon.html file from the OnDemand Server HTML directory, scanning the file for the <ODSLogonMethod> HTML tag where it substitutes a URL for the tag. This URL is used to direct the form to the correct Web broker module (CGI, ISAPI, or NSAPI) when the client clicks the Login button. The HTML is then sent back to the client's Web browser.
4. The Web browser receives the logon.html file. Since the logon.html file does not yet include all of the UI elements, such as images, it resolves the references to these files, retrieves them from the static HTML directory on the Web server, and populates any empty frames.



Client-side logon and Adaptive Report list

Communication protocols: HTTP or HTTPS

The end user is viewing the OnDemand Server logon screen.

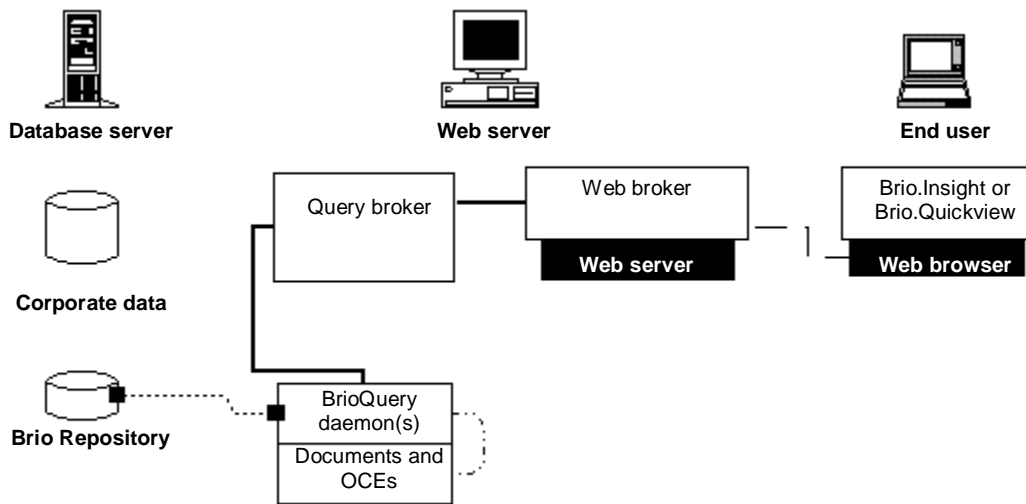
1. The page displayed to the user contains an HTML form. The user enters a user name and password and clicks the Login button. If using OnDemand Server authentication (instead of database or

JavaBean, described in step 4), there is also a button to change the user's password, which follows a similar procedure.

2. The Logon form posts an action to the OnDemand Server (the proper URL to post the message to was substituted in step 3 in "Viewing the logon page," above). The post message sends the user name and password as part of the request to execute the logon method of the OnDemand Server.
3. The post message is sent to the Web broker, which forwards it to the Query broker.
4. The Query broker accepts the user name and password and begins the process of authenticating the user. The authentication process is performed in one of three manners:
 - OnDemand Server Authentication – the Query broker asks a BrioQuery daemon to run a SQL statement to look up the user name and password in the Brio Repository tables. The BrioQuery daemon connects to the database using a pre-defined database user name and password combination, specifically created for this purpose. It returns the encrypted password from the Repository, where it is compared by the Query broker to the password entered by the user. If they match, the user is authenticated; if they do not match, the Web broker returns a message indicating that the logon failed, and is given the opportunity to re-enter.
 - Database Server Authentication – the Query broker asks a BrioQuery daemon to establish a connection to the database, using the user name and password entered on the HTML form as the database login parameters. If the database accepts the logon, the user is authenticated; if the database rejects the login, the Web broker returns a message indicating that the login failed and is given the opportunity to re-enter.
 - Custom JavaBean Authentication – the Query broker passes the user name and password to a custom JavaBean, which is indicated by the BES Administrator when setting up the OnDemand Server. This JavaBean is custom-written user-exit by each site to perform any type of authentication process required by the site. Details about implementing the JavaBean are in the Technical Note, "User Authentication with a JavaBean," available on the Brio Web site at <http://www.brio.com/support/bq5503.html>. The JavaBean ultimately returns a boolean expression back to the Query broker, where 1 indicates that authentication was successful, and 0 indicates that it failed.
5. Once the user has been successfully authenticated, the Query broker instructs the Web broker to set two cookies on the client's computer: the user name and encrypted password are then sent back to the Web browser as *session cookies*. Session cookies are held in memory and are destroyed when you quit the Web browser. This information is stored as cookies because HTTP communication is stateless, meaning that it knows nothing of previous transactions. By using session cookies, the OnDemand Server can re-use them as necessary during the browser session, such as when refreshing the list of Adaptive Reports, without re-prompting the user to continuously log in.
6. Once the user has been successfully authenticated, the Query broker asks a BrioQuery daemon to retrieve a list of Adaptive Reports for the user. This involves sending several SQL statements to look up information in the OnDemand Server Repository – the SQL determines what groups the user belongs to, and then retrieves the names of the reports and the Adaptive Report mode for each report.
7. The Adaptive Report information is returned to the Query broker (from the BrioQuery daemon that was

enlisted in step 6, above), where it is merged into the OnDemand Server HTML files. The Query broker retrieves several files from the OnDemand Server HTML directory, replacing the special OnDemand Server tags with the correct information. The postlogon.html page is the main HTML file that defines a series of frames that are populated by information retrieved from the Brio Repository, or other static HTML documents. One of the frames is used to display the file, reportlist.html. The Query broker scans the reportlist.html file, looking for the <ODSDocListTag> tag. This tag is replaced with the HTML tags to display the Adaptive Report List as a set of hyperlinks.

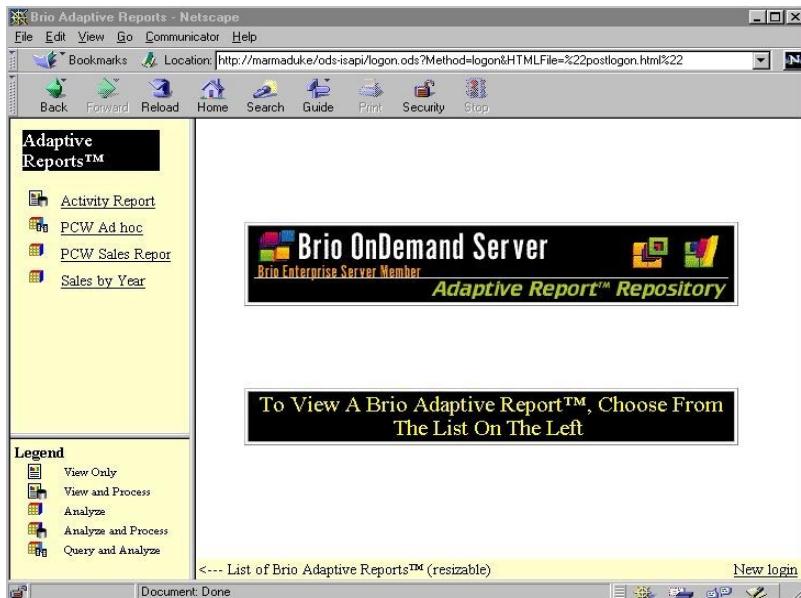
8. The postlogon.html file, the reportlist.html file, and the other files that are referenced as frames are sent by the Query broker through the Web broker, to the client's Web browser.
9. On the client, the web browser receives the HTML files from the Web server, resolves the frame and image references and displays the page to the user.



Selecting a report

Communication protocols: HTTP or HTTPS

A page similar to the following is displayed to the user. Though the BES Administrator or Webmaster can change the appearance of the screen, the common component will be a list of report names displayed as hyperlinks. Refer to the Brio technical note, "Customizing the OnDemand Server," (available at <http://www.brio.com/support/bq5500.html>) for information on how to customize the appearance and functionality of the default Brio HTML pages.



The following procedures occur when a user selects a report from the Adaptive Report list.

1. The Adaptive Report hyperlink sends a message to the Web broker asking for a document to be retrieved. This message is sent as a URL to the Web broker with a method—or argument—for the Query broker to act on.
2. The Web broker forwards the request, the GetDocument method, to the Query broker.
3. The Query broker asks the Web broker to retrieve the username and password cookies from the client's Web browser. The Query broker needs to verify that the user is still authorized to work with the document they have selected.
4. The Web broker gets the cookies from the browser, and sends them to the Query broker. The Query broker calls the BrioQuery daemon to send the SQL required to verify that the access privileges for the document are still valid for the user.
5. If the privileges are good, the BrioQuery daemon returns an Adaptive Report mode level (1-5), along with the location of the physical document, which can reside in the OnDemand Server's file system or in the Brio Repository.
 - File System: If the physical document is stored on the local file system, it is available to the Query broker just by reading it from local disk.
 - OnDemand Server Repository: If the document selected is not stored on the local file system, it is stored in the Brio Repository on the database server. The BrioQuery daemon runs the SQL statements to retrieve the document from the database tables, and saves it to a temporary file. The name of this temporary file is returned back to the Query broker.
6. The document is collected by the Query broker, which asks the BrioQuery daemon to add the "homing" information (the OnDemand Server's hostname and port number, referred to on page 11) and Adaptive

Report token to the document's header.

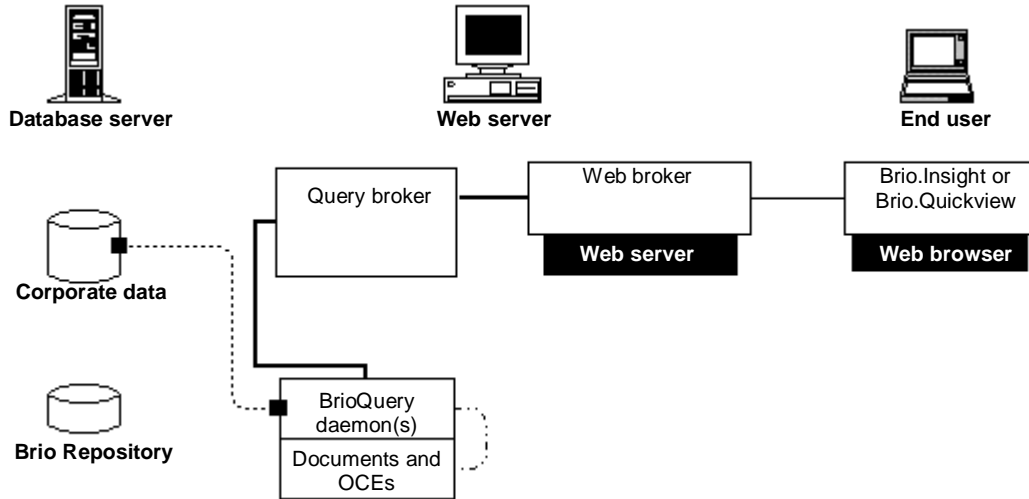
7. The Query broker forwards the document to the Web broker, which tags the document with the Brio MIME type so it can be launched by Brio.Insight or Brio.Quickview. The Web broker then hands off the file to the Web server, which streams it to the Web browser via HTTP or HTTPS
8. The Web browser receives the data stream, interprets the MIME type information as a Brio document, and launches the version of Brio.Insight or Brio.Quickview resident with the browser.
9. As the document is read by the Brio Web client, it reads the header information, which indicates the correct Adaptive Report mode for the document. The Brio Web client dynamically hides or exposes only the proper functionality for that adaptive state.

Query over the Web

Communication protocols: HTTP only

If the Adaptive Report mode for a document allows re-querying of the database (Adaptive Report modes 2, 4, and 5), the following actions are performed when a user clicks the Process button.

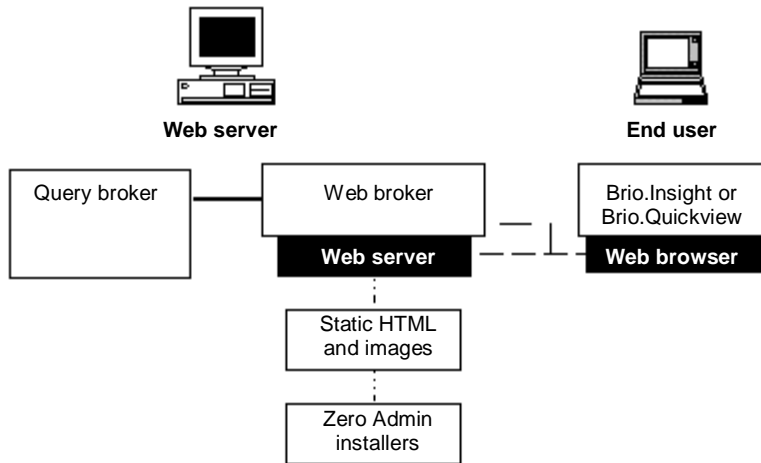
1. Brio.Insight/Brio.Quickview sends part of the document to the Web broker with the request to run the query. The parts of the document not required to process a query are not sent back to the OnDemand Server. This means only the minimum shell of a document with the query definition is sent to the Web broker.
2. The Web broker forwards the request to the Query broker, which saves the document as a temporary file in the OnDemand Server working directory.
3. The Query broker sends a message to the BrioQuery daemon to run the SQL saved in the temporary document, providing it the name of the proper OCE to use. If the BES Administrator requires that the user log in to the database (this is an extra security option discussed in the documentation that accompanies the Brio Enterprise Server), the user name and password are also forwarded to the BrioQuery daemon.
4. The BrioQuery daemon connects to the database and executes the query, saving the new results back into the temporary document.
5. The BrioQuery daemon notifies the Query broker that it has completed the query.
6. The Query broker picks up the temporary document and sends it back to the Web broker, which returns it to the client's Web browser.
7. At the browser, the currently open Brio.Insight/Brio.Quickveiw receives the stream of new data. The original document remains open, with the new result set returned from the OnDemand Server replacing the existing result set. This replacement of results also triggers the client document to refresh existing reports with the new data.



Zero Administration

Communication protocols: HTTP or HTTPS

This diagram shows the location of key components in the architecture for executing Zero Administration. The Zero Administration client installers and the HTML pages with the Zero Administration code are stored on the Web server.



There are two sides to Zero Administration: identifying the version number of the most up-to-date plug-in on the server, and downloading and updating the plug-ins on each client.

Server Requirements

One of the HTML documents that populates the postlogon.html file, workarea.html, includes a reference to a JavaScript file that includes all of the Zero Administration logic. This JavaScript file, ZeroAdmin.js, includes the most recent version of the Brio Web clients in a parameter line. The Brio installer sets this number when

the OnDemand Server is first installed. If the BES Administrator replaces the Brio Web client installers with newer files, he or she needs to manually change the version parameter in ZeroAdmin.js.

Client Processing

1. At each user logon to the OnDemand Server, the Web browser retrieves and parses the HTML documents from the Web server. The JavaScript logic for Zero Administration, which is included in these HTML files, runs in the client's Web browser.
2. It checks the version number in the ZeroAdmin.js file (downloaded from the Web server), and compares it with the version number on the client's computer. The client's version number can be found in one of three locations, depending upon the operating system and the Web browser.
 - Some browsers store the current plug-in (or Helper application) version in a persistent cookie file.
 - Netscape browsers store the plug-in version in the Netscape registry. You can view the registry by choosing "About Plugins" from the Netscape browser's Help menu.
 - Windows browsers can interrogate the plug-in itself to find out what version it is. You can view this information yourself by locating the Brio .dll files (npbqs32.dll for Brio.Insight or npbqv32.dll for Brio.Quickview on Windows 95/98/NT, for example) and displaying their file Properties.See "Appendix B: Zero Administration Components," for more information.
3. After the version numbers are compared, a number of things can happen.
 - If the numbers are equal (meaning the client has the same version number as the ZeroAdmin.js file), or if the client version is greater than the ZeroAdmin.js version, the JavaScript exits and nothing is displayed back to the user.
 - If the version number on the client is less than the version in the ZeroAdmin.js file, the client is prompted to install or upgrade their product.
4. Most popular Web browsers allow automatic download and installation and provide a digital certificate for an extra layer of security. Macintosh, UNIX, and older Windows-based browsers may require the user to first download the Web client installer, then run a setup program. See "Appendix B: " for more information.

CONCLUSION

Using a combination of Java, JavaScript, HTML, and plug-ins, Brio Enterprise is architected to deliver rich, interactive reports within a Web environment. In this white paper, we discussed how Brio Enterprise fits within your current computing infrastructure, taking advantage of Web servers, application servers, database servers, and client computers that run a wide variety of operating systems. We specified exactly what components run on each type of computer, then detailed which communication protocols are used to transfer data between them. Following this discussion, we offer tips on how to deal with some rare issues that may arise in non-intranet Web environments, and then sketch out the sequences of events that occur for each type of user request.

We hope that this paper gives you all the technical information you need to make an informed assessment of how easy it is to deploy Brio Enterprise over the Web, and how Brio's "no-compromise" approach towards Web-based business intelligence is clearly more acceptable to end-users. If you have any additional questions, do not hesitate to contact your Brio representative, or refer to other technical notes and white papers on our Web site.

APPENDICES

Appendix A: System Requirements

Broadcast Server has been certified for the following hardware and software configurations:

Platform	O/S	Hard Disk	RAM	Connectivity API	Optional
Win NT on an Intel-based PC	Win NT 4.0 or later.	8-11 MB	32 MB	32-bit Essbase, 32-bit Metacube, ODBC, OLE DB, OLE DB for OLAP, Open Client, Oracle Express (SNAPI), SQL*Net	MS Exchange, MAPI 1.0-compliant or SMTP-based e-mail. Standard FTP client and server.
Sun Sparc UNIX	Solaris 2.5.1 and 2.6	32 MB	32 MB	ODBC, Open Client, SQL*Net	SMTP-based e-mail
IBM RS/6000 UNIX	AIX 4.1.5 or 4.2	32 MB	32 MB	ODBC, Open Client, SQL*Net	SMTP-based e-mail
HP 9000 700/800 UNIX	HP-UX 10.2	32 MB	32 MB	ODBC, Open Client, SQL*Net	SMTP-based e-mail

OnDemand Server has been certified for the following hardware and software configurations:

Platform	O/S	Hard Disk	RAM	Connectivity API	Optional
Win NT on an Intel-based PC	Win NT 4.0 or later.	9—43 MB	42 MB	32-bit Essbase, 32-bit Metacube, ODBC, OLE DB, OLE DB for OLAP, Open Client, Oracle Express (SNAPI), SQL*Net	MS Exchange, MAPI 1.0-compliant or SMTP-based e-mail. Standard FTP client and server.
Sun Sparc UNIX*	Solaris 2.5.1 and 2.6	45—120 MB	74 MB**	ODBC, Open Client, SQL*Net	
IBM RS/6000 UNIX*	AIX 4.1.5, 4.2	45—120 MB	74 MB**	ODBC, Open Client, SQL*Net	
HP 9000 700/800 UNIX*	HP-UX 10.2	45—120 MB	74 MB**	ODBC, Open Client, SQL*Net	

Note: A Microsoft IIS, Netscape FastTrack or Netscape Enterprise Internet/intranet Web server is required.

* Also requires Motif 1.2, Open Windows 3.2 or CDE 1.0 (or greater.)

** Includes RAM requirements for Java Runtime Environment.

The OnDemand Server requires the Java Runtime Environment (JRE.) The appropriate JRE is included on the Brio Enterprise CD.

The OnDemand Server supports CGI, ISAPI, and NSAPI. Though Brio tries to keep its products always up-to-date via new patches and versions, we can not assure customers that current Brio software will always be compatible with future Web server software.

OnDemand Server v. 5.5.4 has been certified for the following Web servers:

Name	Win NT	UNIX
Microsoft Internet Information Server 4.0	CGI, ISAPI	—
Microsoft Internet Information Server 3.0	CGI, ISAPI	—
Netscape Enterprise Server 3.5.1	CGI,	CGI
Netscape Enterprise Server 3.0	CGI, NSAPI	CGI
Netscape Enterprise Server 2.0	CGI, NSAPI	CGI, NSAPI
Netscape Enterprise Server 3.0	CGI, NSAPI	CGI, NSAPI
Netscape Enterprise Server 3.0.1	CGI, NSAPI	CGI, NSAPI
Netscape Enterprise Server 2.0	CGI, NSAPI	CGI, NSAPI

Brio.Insight & Brio.Quickview has been certified for the following hardware and software configurations:

Platform	O/S	Hard Disk	RAM	Monitor	Browser Extension	Browser
Windows	Win 3.1+ (16-bit), Win 95, Win 98, or Win NT 4.0 or later.	3—4 MB	8 MB	Color or gray-scale	Browser Extension & Stand-alone	Netscape Navigator 3.+ , 4.+. Microsoft Explorer 3.02 with Authenticode, Microsoft Explorer 4.0+
PowerMac	System 7, 8	3.5—5.5 MB	6 MB application RAM	Color or gray-scale	Stand-alone	Netscape Navigator 3.+ , 4.+. Microsoft Explorer 3.02, Microsoft Explorer 4.0+
Macintosh 680x0	System 7	3.5—5.5 MB	6 MB application RAM	Color or gray-scale	Stand-alone	Netscape Navigator 3.+ , 4.+. Microsoft Explorer 3.02, Microsoft Explorer 4.0+
Sun Sparc UNIX	Solaris 2.5.1 and 2.6	14 MB	16 MB	Color or gray-scale	Stand-alone	Netscape Navigator 4.+
IBM RS/6000 UNIX	AIX 4.1.5, 4.2	13 MB	16 MB	Color or gray-scale	Stand-alone	Netscape Navigator 4.+
HP 9000 700/800 UNIX	HP-UX 10.2	16 MB	16 MB	Color or gray-scale	Stand-alone	Netscape Navigator 4.+

Note: JavaScript must be enabled in the Web browser for Zero-Administration.

Browser extensions are plug-ins to your Web browser. Stand-alone applications can be used as Helper applications within a Web browser or they can run on their own.

BrioQuery (Designer, Explorer, and Navigator) have been certified for the following hardware and software configurations:

Platform	O/S	Hard Disk	RAM	Monitor	Connectivity API
Windows	Win 3.1+ (16-bit)	3—17 MB	8 MB	Color or gray-scale	DAL, EDA Link, Essbase, Local Oracle, MDI Gateway, ODBC, Open Client, Sequelink, SQL*Net
Windows	Win 95, Win 98, or Win NT 4.0 or later.	3—17 MB	8 MB	Color or gray-scale	32-bit Essbase, 32-bit Metacube, ODBC, OLE DB, OLE DB for OLAP, Open Client, Oracle Express (SNAPI) (NT only), SQL*Net
PowerMac	System 7, 8	3.5—5.5 MB	6 MB application RAM	Color or gray-scale	32-bit ODBC, Open Client, Sequelink, SQL*Net
Macintosh 680x0	System 7	3.5—5.5 MB	6 MB application RAM	Color or gray-scale	DAL, EDA Link, Essbase, Local Oracle, MDI Gateway, ODBC, Open Client, Sequelink, SQL*Net
Sun Sparc UNIX*	Solaris 2.5.1 and 2.6	14 MB	16 MB	Color or gray-scale	ODBC, Open Client, SQL*Net
IBM RS/6000 UNIX*	AIX 4.1.5, 4.2	13 MB	16 MB	Color or gray-scale	ODBC, Open Client, SQL*Net
HP 9000 700/800 UNIX*	HP-UX 10.2	16 MB	16 MB	Color or gray-scale	ODBC, Open Client, SQL*Net

* Also requires Motif 1.2, Open Windows 3.2, or CDE 1.0 (or greater).

Appendix B: Zero Administration Components

Zero Administration relies upon knowing the exact version of the Brio Web client plug-in installed on the end-user's computer, and then knowing which plug-in (or Helper application) to install, based on the user's operating system and Web browser. There are several ways to discover the version of the Brio Web client, and several ways to install the Web client, some of which have extra security advantages.

Table 5 indicates which techniques are used for all supported platforms. The following tables will assist you if you are unfamiliar with a term.

The Web client's version number can be stored in one of three locations where the Zero Administration JavaScripts can read it.

Version # storage	Indication
Netscape registry	This is a binary file, stored in the Windows directory as nsreg.dat.
Cookies	Small (less than 4K) text file(s). Cookies can reside together in a single file, named cookies.txt, or each cookie can have its own text file in the Windows directory. Persistent cookie files are usually follow the naming convention, user@domain.txt.
Plug-in Inquiry	The plug-ins themselves include version information. Some browser/platform combinations can read this information directly from the plug-in, preventing the need to save the information in a registry or text file.

Platform-specific software archiving software are used to install the Brio Web client. The installer may or may not include a digital certificate, depending upon whether the Web browser and operating system support it.

ZAC Installer	Indication	Digitally signed?
JAR	Java Archive, used for modern Netscape browsers on Windows 32-bit and UNIX platforms.	Yes
CAB	"Cabinet" file, used for Microsoft Web browsers on Windows 32-bit platforms.	Yes
Bin-Hex	Binary archive that has been converted from hexadecimal to ensure portability across platforms. Used for Macintosh files.	No
SE-EXE	Self-extracting executable, used for Windows 3.x platforms.	No

Table 5: Zero Administration Specifications

		Netscape 2.0	Netscape 3.0	Netscape 4.0	MS Internet Explorer 3.02	MS Internet Explorer 4.0
AIX	ZAC Installer	Not supported	Not supported	JAR	N/A	N/A
	Version # storage	Not supported	Not supported	Netscape Registry	N/A	N/A
HP-UX v.10	ZAC Installer	Not supported	Not supported	JAR	N/A	N/A
	Version # storage	Not supported	Not supported	Netscape Registry	N/A	N/A
SunOS 5.4 X86	ZAC Installer	Not supported	Not supported	Not Supported	N/A	N/A
	Version # storage	Not supported	Not supported	Netscape Registry	N/A	N/A
SunOS 5.4 Sparc	ZAC Installer	Not supported	Not supported	JAR	N/A	N/A
	Version # storage	Not supported	Not supported	Netscape Registry	N/A	N/A
Mac 68K	ZAC Installer	Bin - Hex	Bin - Hex	Bin - Hex	Bin - Hex	Bin - Hex
	Version # storage	Cookies	Cookies	Cookies	Cookies	Cookies
Mac PPC	ZAC Installer	Bin - Hex	Bin - Hex	Bin - Hex	Bin - Hex	Bin - Hex
	Version # storage	Cookies	Cookies	Cookies	Cookies	Cookies
Win 3.1	ZAC Installer	SE - Exe	SE - Exe	JAR	CAB	CAB
	Version # storage	Plug-in Inquiry	Plug-in Inquiry	Netscape Registry	Cookies	Cookies
Win95	ZAC Installer	SE - Exe	SE - Exe	JAR	CAB	CAB
	Version # storage	Plug-in Inquiry	Plug-in Inquiry	Netscape Registry	Cookies	Cookies
Win NT	ZAC Package	SE - Exe	SE - Exe	JAR	CAB	CAB
	Info storage	Plug-in Inquiry	Plug-in Inquiry	Netscape Registry	Cookies	Cookies

Appendix C: About Brio Enterprise 5.5

Whether Web-based or connected via client/server, Brio Enterprise client solutions share the same powerful functionality and intuitive interface. Designed for end-user, but built for IT, Brio Enterprise products focus on both simplicity and power. The result is a unified, leading-edge and easy-to-navigate product family that guides users from query, to analysis, to finished report quickly and efficiently. By providing complete product integration, Brio Enterprise reduces IT administration and lowers overall corporate costs.

Client/Server Products

Brio Enterprise's client/server products provide desktop users with direct access to all of their data sources. Depending on the end users needs, reports and analysis are handled in one of three different client/server Brio products. BrioQuery Designer is the most powerful of these products, and it is used to design the queries, reports, and Data Models that are distributed with Brio's Web products.

BrioQuery Designer™ – Query, analysis and reporting with database administration functionality, security, auditing, and Repository setup (for IT departments).

BrioQuery Explorer™ – Query, analysis and reporting with direct access to database tables and the Brio Repository of pre-defined data models and reports (for power users).

BrioQuery Navigator™ – Query, analysis and reporting with access to repository of pre-defined data models and reports (for active analysts).

Brio Enterprise Web Based Products

Brio Enterprise's Web product line consists of two main areas: client Web browser products and Web server products. The core of Brio's enterprise Web solution is the OnDemand Server.

Brio offers flexible choices for IT planners by shipping versions of the OnDemand Server on four operating systems: Windows NT, Sun Solaris, HP-UX and IBM AIX. This means that you can choose from a wide range of hardware to massively scale a system, if need be. Finally, the OnDemand Server supports remote administration so that administrators can manage users, groups, documents, privileges and the server configuration from the comfort of their own desks.

Web Clients

Brio.Insight™ – Browser-based query, analysis and reporting with five varying levels of functionality based on report information and user security (for active analysts and report users).

Brio.Quickview™ – browser-based report viewing and refreshing of data views (for report viewers).

Server Products

Broadcast Server™ – A query server that schedules and automates query processing and report distribution via email, networks, printers, FTP and the Web.

OnDemand Server™ – A Web application server that enables querying over the Web, zero administration clients, report level security and Adaptive Reports.

INDEX

A

adaptive report · 3, 6, 7, 8, 9, 12, 13, 17, 18, 21, 22, 24, 25, 34
administration · 5, 34
 remote · 5, 34

B

Brio products
 Brio Enterprise Server · 5, 15, 25
 Brio.Insight · 1, 6, 7, 8, 17, 18, 25, 27, 30, 34
 Brio.Quickview · 1, 6, 7, 8, 17, 18, 25, 27, 30, 34
 Broadcast Server · 12, 18, 29, 34
 OnDemand Server · 1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 27, 29, 30, 34
 Server Administrator · 5, 6, 8, 10, 11, 12, 13, 15, 16, 17, 22, 23, 25, 27
BrioQuery daemon · 12, 13, 14, 17, 19, 20, 22, 24, 25
broker · 8, 9, 11, 12, 13, 14, 15, 17, 19, 21, 22, 23, 24, 25
 Query broker · 8, 9, 11, 12, 13, 14, 15, 17, 19, 21, 22, 23, 24, 25
 Web broker · 9, 11, 13, 17, 19, 21, 22, 23, 24, 25

C

CAB · 9, 10, 32, 33
CGI · 5, 9, 19, 21
cookies · 6, 7, 22, 24, 32

D

daemon · 12, 13, 20
database · 3, 4, 5, 8, 11, 12, 13, 16, 17, 19, 20, 21, 22, 24, 25, 34
 API · 3, 12, 16, 20
 server · 4, 12, 17, 24
database server · 5, 16, 24
dispatcher · 11, 13, 14, 17, 19

E

extranet · 4, 18

F

factory · 13, 14, 19

firewall · 19

H

HTML
 dynamic HTML · 9, 15
 static HTML · 6, 9, 10, 11, 12, 15, 21, 23
HTML tags · 9, 12, 13, 15, 23, 25
HTTP · 3, 6, 8, 17, 18, 19, 21, 22, 23, 25, 26
HTTPS · 6, 17, 18, 19, 21, 23, 25, 26

I

Internet · 6, 18, 19, 29, 30, 33
intranet · 3, 4, 18, 19, 21, 29
ISAPI · 5, 9, 19, 21

J

JAR · 9, 10, 32, 33
Java · 3, 5, 11, 13, 29, 32
 JRE · 11, 13, 29
JavaScript · 5, 6, 7, 9, 10, 12, 26, 27, 31

L

logon.html · 6, 12, 15, 21, 23, 26

M

Macintosh · 7, 27, 30, 31, 32
method · 22, 24
MIME type · 25

N

NSAPI · 5, 9, 19, 21
NT service · 11, 14

O

OCE · 8, 12, 25

P

plug-in · 6, 7, 18, 19, 26, 27, 32
post · 6, 12, 15, 22, 23, 26
postlogon.html · 6, 12, 15, 23, 26
process factory · 13

Q

query Agent · 11, 13, 14, 17, 19
Query broker · 8, 9, 11, 12, 13, 14, 15, 17, 19, 21,
22, 23, 24, 25
 Manager · 11, 13, 14, 17, 19
 Node · 11, 13, 14, 17, 19
 Process factory · 13, 14, 19
query processor · 19

R

registry · 7, 27, 32
 Netscape registry · 7, 27, 32
repository · 22, 34

S

security · 3, 5, 8, 10, 12, 25, 27, 32, 34
server
 application server · 4, 5, 11, 34
 database server · 4, 12, 17, 24
 proxy server · 3, 19
 Web server · 4, 5, 6, 8, 9, 10, 11, 12, 15, 17,
18, 19, 21, 23, 25, 26, 27, 29, 30, 34
cookie · 22
SSL · 18, 19

T

TCP/IP · 17, 19
 sockets · 18
 TCP/IP sockets · 17, 18, 19

U

UNIX · 6, 7, 11, 14, 27, 29, 30, 31, 32, 33, 34

W

CGI
 CGI · 3, 5, 9, 17, 19, 21, 30
Web broker · 5, 9, 11, 13, 17, 18, 19, 21, 22, 23,
24, 25
 ISAPI · 5, 9, 17, 19, 21, 30
 NSAPI · 5, 9, 17, 19, 21, 30

Z

Zero Administration · 3, 5, 6, 7, 9, 10, 11, 12, 13,
18, 19, 26, 27, 32, 33, 34
ZeroAdmin.js
 ZeroAdmin.js · 26, 27
zeroadmin.os
 zeroadmin.os · 27