Brio Intelligence Server Guide

Version 6.6

BRIO SOFTWARE™

4980 Great America Parkway
Santa Clara, CA 95054 USA
+1(408)496-7400

Brio Intelligence Server Guide

Part Number 1209095

Brio Software
4980 Great America Parkway
Santa Clara, CA 95054
+1(408)496-7400

support@Brio.com
sales@Brio.com
www.Brio.com

Contents

Figures

Tables

About This Book

The *Brio Intelligence Server Guide* provides architecture concepts and administration instructions for Brio Intelligence Server—Brio Broadcast Server and Brio OnDemand Server. It includes:

- Descriptions of the components that make up the Brio Intelligence Server
- Configuration and administrative procedures for the Broadcast Server and the OnDemand Server
- Types of connection (OCE) files and how to create them
- Information on customizing the OnDemand Server Web site including Zero Administration
- Reference information on the database tables
- Troubleshooting information listed by error code messages

This guide provides deployment information for Brio.Insight and Brio.Quickview (through Zero Administration). These Web client applications are not discussed from the end-user perspective.

For a list of supported hardware refer to the readme file on the CD.

Note Most screen shots presented in this book were taken on a Windows NT system. Actual screens may appear different according to your system.

Audience

This guide is intended for database and IT administrators. It assumes familiarity with database applications and Brio products.

Help

If you need help with Brio products or their documentation *and* you have a current Brio Technical Support agreement, call Brio Technical Support at 800.337.6324 (within North America). You may also send an email message to support@brio.com. Please be prepared to provide your valid customer number and company name. You will also need to know the version of Brio Intelligence Server you are using, your operating system, and all data source names and versions. If you wish to execute a Brio Technical Support agreement, call toll free at 877.289.2746.

1 Brio Intelligence Server: Providing Easy Access to Reports

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 the *Brio Intelligence Server*, which provides database connectivity without the installation of database client software. The Brio Intelligence Server works on behalf of the Brio Web clients to deliver authorized reports and data refreshes whenever a client requests new information. Brio distributes processing among client computers and servers, as shown in Figure 1-1. Brio Intelligence 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.

Though you can install all Brio Intelligence Server components on a single server for testing or demonstration purposes, the components should be distributed between client computers and two or more servers for optimal scalability and reliability.

This chapter contains the following sections:

- Brio Intelligence Server Components
- Security
- How OnDemand Server Components Work Together

Brio Intelligence Server Components

Brio Intelligence Server works on behalf of the user to deliver authorized reports and data refreshes whenever the user requests new information. Its two main components are *Broadcast Server* and *OnDemand Server*.

Figure 1-1 shows Web browsers on client machines, a Web server, OnDemand Server components on multiple servers, and a database server. Using Brio's drag and drop interface, a client requests a list of documents and views the list from a Web browser. Brio Intelligence Server components manage the scheduling of the request (for printing, email notification, and so on), the routing of the request to the database and the return of the document list to the client's Web browser.

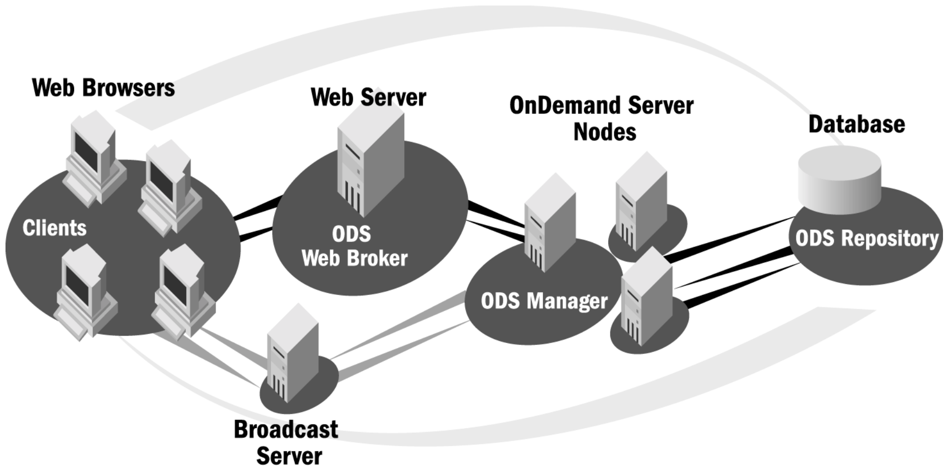


Figure 1-1 Brio Intelligence Server Components

Broadcast Server

Broadcast Server automates scheduled query processing and report notification, and distributes the results through email, the Web, FTP, and internet file servers, or printers. Users schedule the processing of reports based on date, time, or event—such as a database update. Reports—with or without saved result sets—are published to specified users or groups of users who can then perform additional analysis without ever connecting to the database.

Broadcast Server Components

Broadcast Server consists of the following components (see Figure 1-2):

- Database Repository – Stores documents and scheduling information (alternative storage is a file system on local computer).
- BC Listener – Java-based application that manages client requests to the BCServer machine's file system-based job storage.
- BCServer – Scheduling agent that polls and updates a queue of scheduled jobs, routes them to Brio daemon (process), and distributes results through network resources. Broadcast Server can simultaneously poll job queues stored in multiple job repositories.
- Brio Daemon – Component that connects to databases and processes queries at the request of BCServer. Also referred to in this guide as BQ daemon or BQ process.

Scheduled Queries

Broadcast Server runs scheduled queries automatically, at appointed times and at intervals, keeping published reports up to date. Frequently-run queries off-loaded to Broadcast Server reduce transactions with the database and conserve network bandwidth during peak periods.

Broadcast Server can run scheduled documents in multiple cycles. Each run will retrieve data sets based on specified requirements and constrained by access privileges.

Documents that require wide distribution can be scheduled once and deliver the correct information to diverse audiences from different regions or divisions.

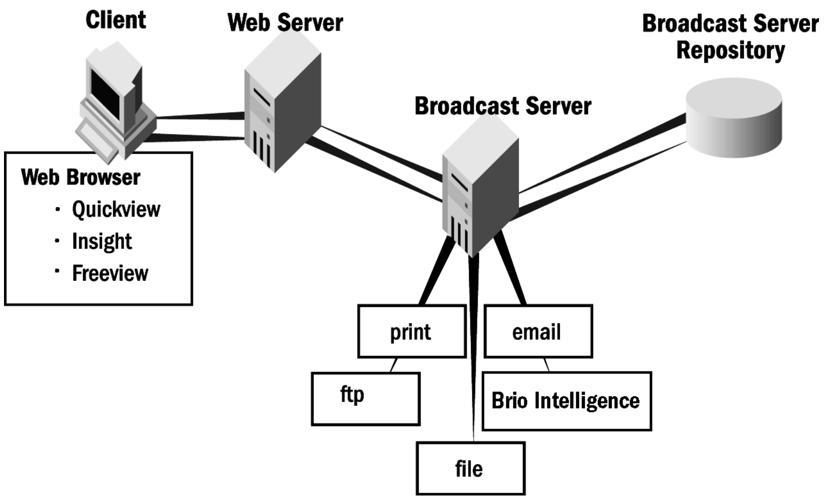


Figure 1-2 Brio Broadcast Server

OnDemand Server

The OnDemand Server is a Web-based application server that allows a user to view and select from a list of available documents. These documents can be either regular reports or they can contain components that allow users to build their own queries and reports.

Access to a document depends on the privilege level of the group to which a user belongs and the groups assigned to that document. When a document is registered, the administrator can assign different privilege levels that then supersedes default group permission. (See “Security” on page 1-17 for more information.)

OnDemand Server Components

The OnDemand Server is composed of modules that perform authentication, document retrieval, and document processing requests. Core components of the OnDemand Server 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 UNIX.

The OnDemand Server has the following modules:

- Web Broker – Resides in the system's Web server – one Web server per *cluster*. A cluster consists of one ODS manager and one or more ODS nodes.
- ODS Manager – Generally resides on a separate server – one ODS manager per cluster.
- ODS Node (s) – Generally resides on a separate server – a cluster can have multiple nodes
- ODS Repository (ies) – Resides on the database server. (See Appendix B, "Database Table Reference," for more information on the ODS Repository.)

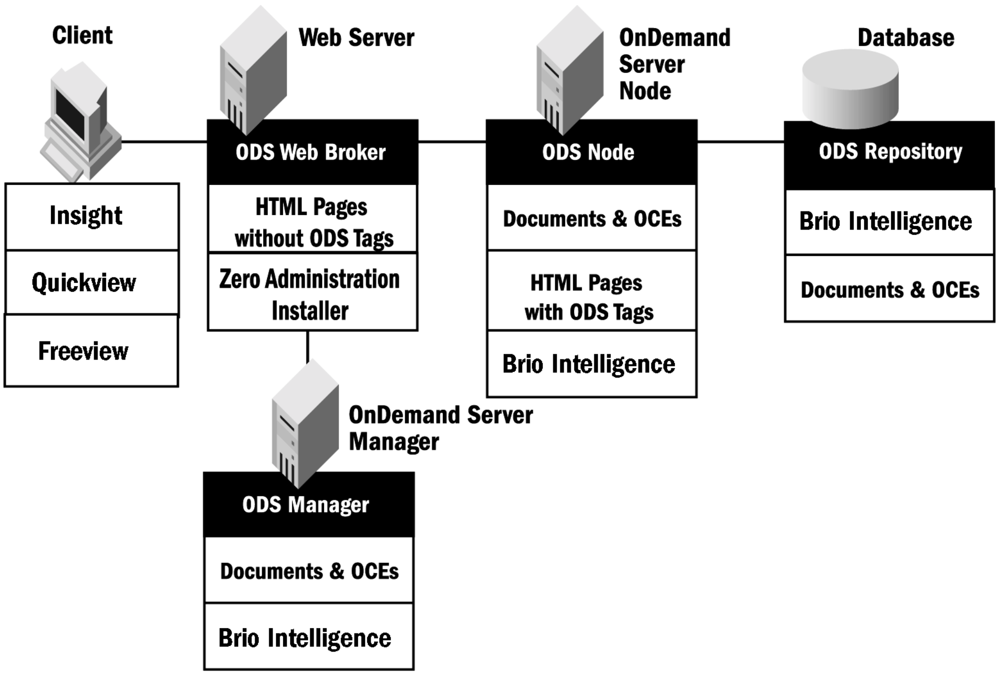


Figure 1-3 OnDemand Server Components

Web Broker

The Web Broker is a small communications module that transfers requests and information between the Web server and the OnDemand Server. The Web Broker is available in three types:

- Internet Server API (ISAPI) – Uses the Microsoft IIS Web server's native, proprietary application programming interface (API).
- Netscape Server API (NSAPI) – Uses the Netscape Web server's native, proprietary application programming interface (API).
- Common Gateway Interface (CGI) – A generic Web server interface that can run on IIS, Netscape Enterprise or iPlanet, and Apache.

The native Web Brokers perform better (faster). They are re-entrant, meaning that only one instance of the application needs to be loaded into memory regardless of the number of users or connections. With CGI, which is a generic Web Broker, every new connection spawns another instance of the CGI executable. If you are planning on having a large number of named users (500+), Brio recommends that you use one of the native web brokers in order to avoid a bottleneck at the Web server level.

ODS-related components in the Web server:

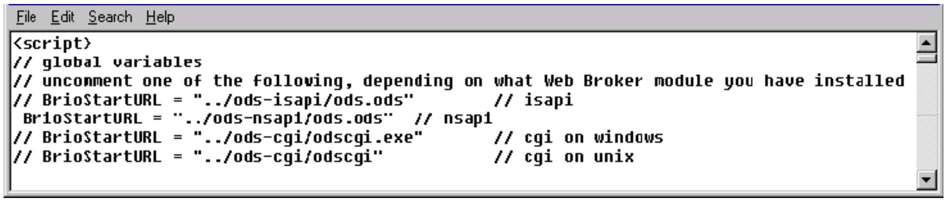
- HTML files without ODS tags – The images and instructional text that users see when they log in reside as HTML and GIF files on the Web server. These files display the opening screen and user preferences. These files do not need to be processed or modified in any way and are referred to in older documentation as “static” files.

  In Windows the HTML files without ODS tags are stored in `C:\ Program Files\ Brio\ Brio Enterprise Server\ Server\ HTML.` In UNIX these files are stored in `$BRIOHOME/ SERVER/ HTML.`

  These 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.

You can change the appearance and behavior of the OnDemand Server by modifying these HTML files. For example, to change a Web server type, you would “uncomment’ the new server type in Brio.html (shown in Figure 1-4.)

For the latest information on customizing the OnDemand Server HTML pages see “Customizing the OnDemand Server” on page 4-2.



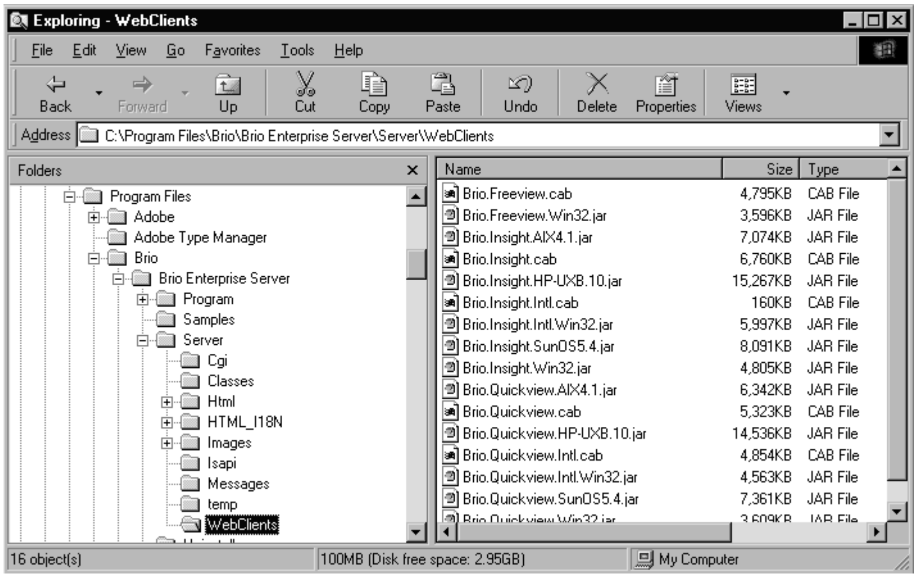
```
<script>
// global variables
// uncomment one of the following, depending on what Web Broker module you have installed
// BrioStartURL = "../ods-isapi/ods.ods"        // isapi
 BrioStartURL = "../ods-nsapi/ods.ods"  // nsapi
// BrioStartURL = "../ods-cgi/odscgi.exe"       // cgi on windows
// BrioStartURL = "../ods-cgi/odscgi"           // cgi on unix
```

Figure 1-4 Web Server Type

- Zero-Administration Installers – JavaScript file that compares the version of the Brio Web client on the user’s machine with that on the Web server. If the version on the Web server is newer, the JavaScript prompts the user to download the newer version of the Brio Web client.

  If using a Windows’ Web browser, the JAR and CAB files include the installer, plus a digital certificate for extra security.

  In UNIX the Zero-Administration Installers are stored in `$BRIOHOME/SERVER/WebClients.`



Address C:\Program Files\Brio\Brio Enterprise Server\Server\WebClients

- Program Files
  - Adobe
  - Adobe Type Manager
  - Brio
    - Brio Enterprise Server
      - Program
      - Samples
      - Server
        - Cgi
        - Classes
        - Html
        - HTML_I18N
        - Images
        - Isapi
        - Messages
        - temp
        - WebClients

Name	Size	Type
Brio.Freeview.cab	4,795KB	CAB File
Brio.Freeview.Win32.jar	3,596KB	JAR File
Brio.Insight.AIX4.1.jar	7,074KB	JAR File
Brio.Insight.cab	6,760KB	CAB File
Brio.Insight.HP-UXB.10.jar	15,267KB	JAR File
Brio.Insight.Intl.cab	160KB	CAB File
Brio.Insight.Intl.Win32.jar	5,997KB	JAR File
Brio.Insight.SunOS5.4.jar	8,091KB	JAR File
Brio.Insight.Win32.jar	4,805KB	JAR File
Brio.Quickview.AIX4.1.jar	6,342KB	JAR File
Brio.Quickview.cab	5,323KB	CAB File
Brio.Quickview.HP-UXB.10.jar	14,536KB	JAR File
Brio.Quickview.Intl.cab	4,854KB	CAB File
Brio.Quickview.Intl.Win32.jar	4,563KB	JAR File
Brio.Quickview.SunOS5.4.jar	7,361KB	JAR File
Brio.Quickview.Win32.jar	3,609KB	JAR File

16 object(s) | 100MB (Disk free space: 2.95GB) | My Computer

Figure 1-5 Zero Administration Web Client Installers

Note In a BrioONE deployment, Brio.Portal provides the user interface instead of the HTML files. Brio.Portal also invokes the Zero-Administration Installers.

Table 1-1 Web Server Components

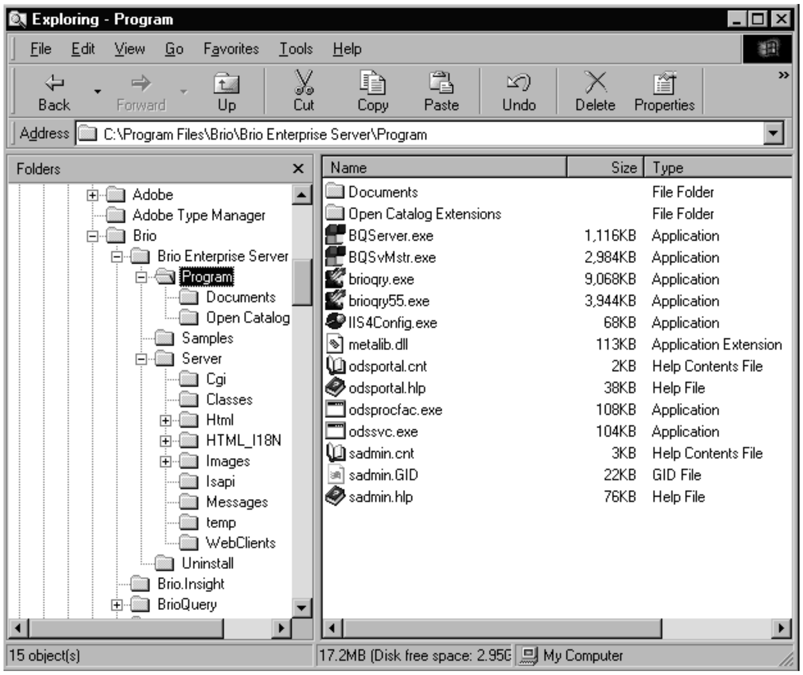
Web server component	Technologies used	Purpose	Notes
Web Broker	ISAPI, NSAPI, or CGI	Communications module that transfers client requests and information between the Web server and the OnDemand Server.	If using NSAPI or ISAPI, the Web broker is loaded by the Web server at startup as a DLL, where it runs inside the Web server's address space; if using CGI, the Web broker is loaded dynamically with each request in its own memory space.
HTML files without ODS tags	HTML documents and GIF image files. Also includes the JavaScript code for Zero Administration.	User interface elements for the logon, document list, and work area pages.	These HTML and GIF files populate frames defined by the HTML files with ODS tags 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.

OnDemand Server Manager

The ODS Manager is the module that controls work-flow. Each server that hosts the ODS Manager or Node has an ODS *Process Factory* that spawns, or launches, these components.

The ODS Manager responds to requests for the address of an available node sent by the Web Broker. If multiple requests are being processed, the ODS Manager performs queue management.

The Manager keeps track of the port number for the active Node (for example, 5501: increment the Manager port number by 1). Only one ODS Manager runs in an ODS cluster. As client requests come in from the Web broker, the Manager returns the port number of the active Node to the Web Broker.



Name	Size	Type
Documents		File Folder
Open Catalog Extensions		File Folder
BQServer.exe	1,116KB	Application
BQSvMstr.exe	2,984KB	Application
brioqry.exe	9,068KB	Application
brioqry55.exe	3,944KB	Application
IIS4Config.exe	68KB	Application
metalib.dll	113KB	Application Extension
odsportal.cnt	2KB	Help Contents File
odsportal.hlp	38KB	Help File
odsprocfac.exe	108KB	Application
odssvc.exe	104KB	Application
sadmin.cnt	3KB	Help Contents File
sadmin.GID	22KB	GID File
sadmin.hlp	76KB	Help File

15 object(s) | 17.2MB (Disk free space: 2.95G | My Computer

Figure 1-6 Core Applications Used By OnDemand Server

ODS-related components that reside with the ODS Manager:

- ODS Process Factory – A continuously running process that spawns other processes for the OnDemand Server.
- Documents – The documents are report-related files, such as shared queries, reports, and analyses, created by Brio Intelligence tools.

  Documents are stored in a directory accessible to the ODS Manager and Node(s). If the directory is on the server that hosts the Manager, a copy is maintained (mirrored) in each server that hosts a Node.
- Open Catalog Extension files (*OCE files*) – Binary files that define which database connectivity API and other login information the BQ daemon uses to connect to specific data sources.

  OCEs are stored in a directory that is copied and maintained (mirrored) on each server that has an ODS node.
- OnDemand Server NT Service or UNIX shell script – When the Manager and Node reside on the same computer, `ODS.start` starts the Manager and a ODS Process factory. The Manager JRE, when it starts, requests the Process factory to start a Node. If the Node is configured to pre-spawn BQ processes it asks the Process factory for them. When a Node is not on the same computer as the Manager, `ods.start` starts a Process factory and the node JRE.

Table 1-2 OnDemand Server Manager Components

Component	Technologies used	Purpose	Notes
ODS Manager	100% Java module that runs within its own Java Runtime Environment (JRE).	The ODS manager responds to the Web Broker's request for an ODS node port number.	The Manager runs on port 5500 (or whichever port was selected by the administrator during installation).
ODS Process Factory	Brio-specific compiled C++ code.	Spawns other processes for the OnDemand Server.	
Brio Documents	Documents (`.bqy`) created by Brio Intelligence client/server tools.	Shared queries, reports, and analyses.	Brio documents are stored in the Documents directory on the server hosting the ODS Manager using the server's file system and mirrored in the server(s) hosting the Node(s). The Broadcast Server can be used to run reports on a scheduled basis, and save them in the Documents folder for shared use.

Table 1-2 OnDemand Server Manager Components *(Continued)*

Component	Technologies used	Purpose	Notes
Open Catalog Extensions (OCE)	Proprietary Brio format	Define which database connectivity API and other login information the BQ daemon uses to connect to specific data sources.	The BES Administrator or client/server Brio Intelligence user creates OCEs.
ODS Windows NT service or UNIX shell script	Windows NT Service or UNIX shell script.	Starts ODS core components.	At least two JREs and the Process Factory run simultaneously when ODS is active. One or more BQ daemons can run, as well.

OnDemand Server Node

An ODS Node works as the middleman between the Web broker and BQ daemons. A Node is a Java server application that, like the Manager, runs in its own JRE. On each server that hosts an ODS Node, there resides an ODS Process Factory that launches the ODS Node.

The ODS Node instructs the BQ daemon to authenticate the user, obtain the document list from the ODS repository, and send it back to the Web Broker for display on the user's browser.

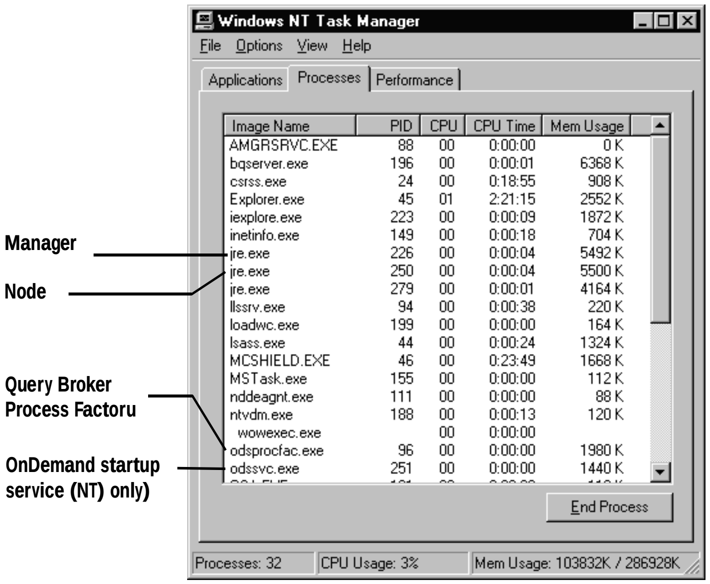


Image Name	PID	CPU	CPU Time	Mem Usage
AMGRSRVC.EXE	88	00	0:00:00	0 K
bqserver.exe	196	00	0:00:01	6368 K
csrss.exe	24	00	0:18:55	908 K
Explorer.exe	45	01	2:21:15	2552 K
iexplore.exe	223	00	0:00:09	1872 K
inetinfo.exe	149	00	0:00:18	704 K
jre.exe	226	00	0:00:04	5492 K
jre.exe	250	00	0:00:04	5500 K
jre.exe	279	00	0:00:01	4164 K
llssrv.exe	94	00	0:00:38	220 K
loadwc.exe	199	00	0:00:00	164 K
lsass.exe	44	00	0:00:24	1324 K
MCSHIELD.EXE	46	00	0:23:49	1668 K
MSTask.exe	155	00	0:00:00	112 K
nddeagnt.exe	111	00	0:00:00	88 K
ntvdm.exe	188	00	0:00:13	120 K
wowexec.exe		00	0:00:00	
odsprocfac.exe	96	00	0:00:00	1980 K
odssvc.exe	251	00	0:00:00	1440 K

Processes: 32 CPU Usage: 3% Mem Usage: 103832K / 286928K

Figure 1-7 Processes Running When ODS Is Running

ODS-related components that reside with the ODS node:

- ODS Process Factory – A continuously running process that spawns other processes for the OnDemand Server.
- BQ Daemon – Services all ODS transactions with a database. Each user login to the OnDemand Server triggers a database transaction that authenticates the user. The node instructs the BQ daemon to get the document list. User requests to process a document for new information are also passed to a BQ daemon.

  The OnDemand Server allows the administrator to specify the maximum number of BQ daemons that can run at one time. Any requests in excess of this maximum are queued until another currently running process is completed.
- HTML files with ODS Tags – These HTML files contain special OnDemand Server tags. The OnDemand Server reads the files, interprets the special tags, and replaces them with user-specific information. For example, the server dynamically replaces the special ODS <`ODSDOCListTag`> tag with the list of files a particular user is allowed to access. The OnDemand Server performs this replacement as it returns the HTML page to the user's browser. These files were referred to in older documentation as "dynamic" files.

Table 1-3 OnDemand Server Node Components

Component	Technologies used	Purpose	Notes
ODS Node	100% Java module that runs within its own JRE.	The ODS node(s) handles all user requests, forwarding them as appropriate to the database.	
ODS Process Factory	Brio-specific compiled C++ code.	Spawns other processes for the OnDemand Server.	Each server that hosts an ODS component (Manager, Node) has an installed ODS Process Factory.
BQ daemon (process)	Brio-specific compiled C++ code.	Queries a variety of data sources on behalf of Web clients. It is also used to retrieve authentication and user privileges from the ODS Repository, which is stored in a relational database.	Connects to a variety of relational (RDBMS) or multidimensional (MDD) databases. The BES Administrator can specify that several (or zero) BQ daemons pre-launch when the OnDemand Server starts.
HTML files with ODS Tags	HTML with custom ODS tags and JavaScript	These HTML files include custom ODS tags that are dynamically processed by the Node to initiate users' authentication and to populate the list of Reports. These HTML files also include frames that define the display structure for the HTML files without ODS tags that reside on the Web server.	Includes logon.html, postlogon.html, and reportlist.html. These pages can be customized by an administrator with an understanding of HTML, JavaScript, and the custom ODS tags. For the latest information on customizing the OnDemand Server HTML pages see `www.brio.com/support _services/technical _support/technical _notes.html`.

OnDemand Server Repository

The OnDemand Server Repository is created as a set of databases tables in a relational database. It stores Brio reports and OnDemand Server configuration information, such as user and group privileges, server location, and user names. See Appendix B, "Database Table Reference" for more information on the ODS Repository.

☆ **Tip** The simplest configurations place the Web server, OnDemand Server, and Broadcast Server on the same physical machine. A better solution, for performance reasons, is to run the Web server on one machine, Broadcast Server on another, and place the OnDemand Server manager on a third machine and the OnDemand Server node(s) on separate machines.

As a rule, the OnDemand Server modules should be installed on a different physical machine from the database server that hosts the Repository. In addition, for performance and security reasons, the Repository should not be on the database server as the enterprise data.

OnDemand Server as a BrioONE Service

Brio Intelligence can be deployed as part of BrioONE —an integrated analytic infrastructure for e-business that includes Brio.Portal. OnDemand Server plays an important role in the integration, automatically publishing its documents into Brio.Portal and delegating user authentication to Brio.Portal.

How the OnDemand Server Communicates

The ODS components communicate with each other through three different communications protocols:

- HTTP or HTTPS – Between the Web browser and the Web server
- TCP/IP Sockets – Between OnDemand Server components. The Web server uses built-in mechanisms to pass and receive information from the CGI/ISAPI/NSAPI module, which in turn makes a TCP/IP socket connection to the ODS manager. Communications between the manager, the node, the process factory, and the BQ daemons all use TCP/IP sockets.
- Database connectivity API, such as SQL*Net, ODBC, or Open Client – For communication between the BQ daemon and the repository tables on the database. APIs are used to connect to databases, run queries, and to refresh data.

Figure 1-8 and Table 1-4 summarize which protocols each computer uses to communicate with other OnDemand Server components.

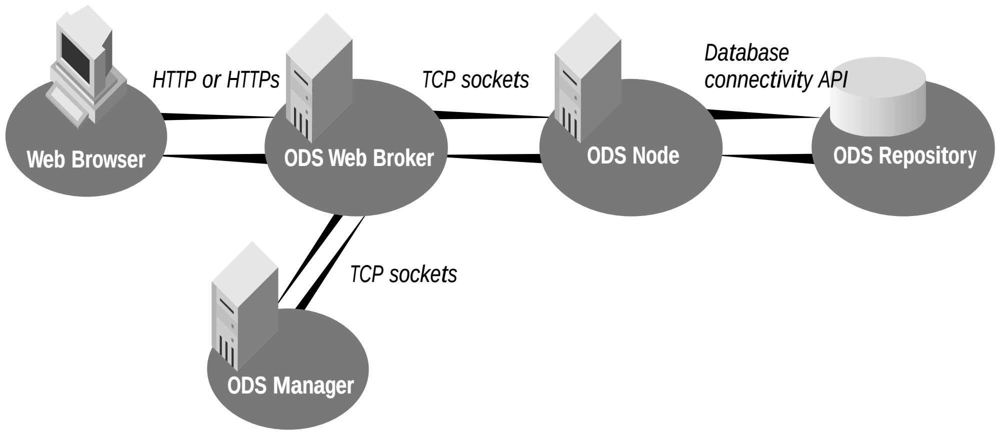


Figure 1-8 Communications Protocols

Table 1-4 Communication Protocols

Protocol	Description
HTTP or HTTPS	Web browser to the Web broker or Web server for login, document lists, and initial document retrieval.
HTTP	Brio Web client (Insight or Quickview) to the Web broker for query processing and data refresh. ODS includes proxy server support. **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 files and images.
File system read/write	HTML file reads on the Web server, and Brio document reads and writes by the BQ daemon on the OnDemand Server.
Brio proprietary API over TCP/IP sockets	Web broker (installed on the Web server) to the Manager, Node, and BQ daemons.
Database connectivity middleware API	Connectivity API software, such as OLE DB, ODBC, SQL*NET, Open Client, and so on, must be installed on both sides of the connection.

Note In a BrioONE deployment, the OnDemand Server manager communicates with Brio.Portal over TCP/IP sockets using the ONE/Integrator API when publishing a document. When retrieving or processing a document the OnDemand Server node communicates using ONE/API.

Security

Brio Intelligence Servers provide two levels of security, Adaptive Reporting and Row Level Security.

Adaptive Reporting

Adaptive reporting is a security mechanism that allows the Brio Intelligence Administrator to authorize groups of users to a specific level of access to specific reports. The Brio Intelligence Server Administrator uses this security mechanism to assign default privileges to a group. The privileges, described in Table 1-5, are called adaptive reporting states.

Table 1-5 Adaptive Reporting States

States	Description
1 – View	View and print reports.
2 – View and process	View and print reports and refresh results.
3 – Analyze	View and print reports and perform offline analysis.
4 – Analyze and process	Perform offline analysis and refresh results.
5 – Query and analyze	Perform offline analysis and refresh results plus query new data and change existing queries.
6 – Data model and analyze	Modify existing data models, add new query sections and create data models based on the document's pre-existing database connections (.oce files). An administrator can publish an empty document that has all of the data connections available for specific user types. User can open the document and create data models (by accessing the table catalog), queries, results, charts, pivots, and reports based on this data.

The group's default privileges can be overridden by privileges assigned with the document. For example, the Marketing group might be granted data modeling, analysis and query capabilities (6) for data pertaining to lead generation, but may have view only (1) or no access at all to corporate financial data. For a particular document containing financial data, the document, when registered, might assign analyze (3) privilege.

When Insight or Quickview opens a document or report, it reads a token that the OnDemand Server inserts into the document. This token instructs the Brio Web client to dynamically adapt its user interface, access to toolbars, menus, and other features, to one of the six adaptive states.

Quickview is a report viewer that can adapt between the first two modes, while Insight is capable of full report building, interactive analysis, ad hoc querying, and data modeling.

BES Administrator stores Adaptive Report privileges in the ODS Repository. When a client application, Insight or Quickview, opens a document or report, it reads a token that the OnDemand Server inserts into the document itself. This token instructs Insight (or other Brio Web client applications) to dynamically *adapt* its user interface, access to toolbars, menus, and other features, to one of the six modes listed in Table 1-5.

Row Level Security

While adaptive reporting allows users different access levels to reports, row level security. Row level security is implemented by means of tables within the server's repository. For more information see "Introduction to Row Level Security", which is available from Brio Systems Consultants.

Load Balancing and Failover

The OnDemand Server supports load balancing and failover across a cluster of servers. Each OnDemand Server cluster is comprised of a *manager* and one or more *nodes*. Transaction requests are all received by the manager, which distributes the request to one of the nodes in the cluster. The distribution is conducted *round-robin* fashion across all the active nodes in the cluster. If a node fails while performing a transaction, the manager automatically re-forwards the request to a different node in the cluster.

The configuration of hardware in the cluster is scalable to allow virtually unlimited expansion. The physical server where the manager is installed may also serve as a node. Additional nodes must each reside on separate physical servers, one node per server. Managers and nodes may be any combination of server operating systems—mixing a Windows NT manager with NT and UNIX nodes is fully supported.

When you install OnDemand Server on each node machine, you are prompted to supply it with the IP address of the manager machine. When an OnDemand Server node starts, it establishes a TCP/IP socket connection to the manager, alerting the manager that the node is alive and ready. The manager then is able to distribute the load by cycling user requests for service across all the nodes in the cluster.

The OnDemand Server node software communicates using TCP/IP sockets, starting a socket listener on the port number one greater than the port number the OnDemand Server manager is using. For example, if the default OnDemand Server port number is 5500, the node will use port number 5501.

If a client (Server Administrator or CGI) discovers that a node process is not responding, it informs the manager by re-requesting a connection.

The manager also maintains a monitor thread that looks for any service that has not recently sent a message indicating it is alive. Thus, both manager and nodes take responsibility for maintaining their connection.

If a node fails to respond, the manager contacts the process factory, and the service is restarted. If the manager fails to respond, then the various node servers will notice and tell their process factories to re-register themselves.

All administration functions are performed by connecting to the manager. When the node starts up, all configuration information and document/connection files are automatically replicated from the manager to all nodes in the cluster. The file synchronization is tracked by entries in the *ODSFileVersion* file that resides on both the manager and nodes. The files are monitored by version number, with only changed file replicated out to the nodes.

How OnDemand Server Components Work Together

This section explains how the OnDemand Server components work together. Figure 1-9 illustrates the top level sequence of events that occur when a request is entered from a Web client such as Insight or Quickview. The following four sections provide the details on how the OnDemand Server works:

- Viewing the Logon Page
- User Authentication
- Selecting a Report
- Scheduling a Job that Posts to the Web

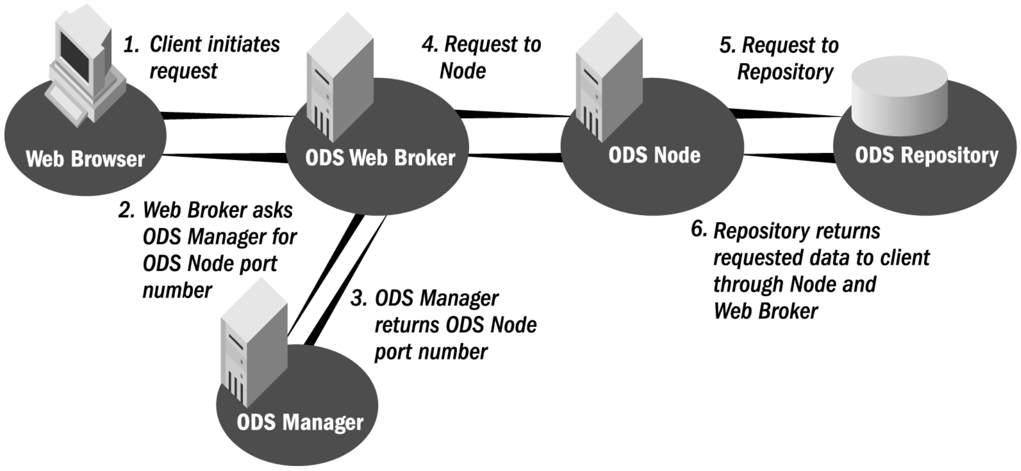

1. Client initiates request
4. Request to Node
5. Request to Repository
Web Browser
ODS Web Broker
ODS Node
ODS Repository
2. Web Broker asks ODS Manager for ODS Node port number
6. Repository returns requested data to client through Node and Web Broker
3. ODS Manager returns ODS Node port number
ODS Manager

Figure 1-9 OnDemand Server Sequence of Events

Viewing the Logon Page

The user launches the Web browser and connects to the corporate intranet: this transaction uses HTTP or HTTPS.

1. The user either types a URL or clicks a hyperlink from Brio.html or an equivalent customized link that contains the URL to the OnDemand Server's Web Broker. The URL itself will indicate whether the Web Broker runs as an ISAPI, NSAPI, CGI module.
2. The Web Broker forwards the logon page request to the Node.

3. The Node 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 HTML directory on the Web server, and populates any empty frames.

Note In a BrioONE deployment, Brio.Portal provides the logon page.

User Authentication

After a user enters a user name and password in the form displayed, the logon information goes to the Web Broker, which forwards it to the ODS Node. the Node accepts the user name and password and begins the process of authenticating the user, in one of the following ways:

- OnDemand Server Authentication – The ODS Node asks a BQ daemon to run an SQL statement to look up the user name and password in the ODS Repository tables. The BQ 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 Node 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 ODS Node asks a BQ 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 ODS Node passes the user name and password to a custom JavaBean, which is indicated by the BES Administrator when configuring the OnDemand Server. This JavaBean is

custom-written at each site to perform any type of authentication process required by the site. Details about writing and implementing the JavaBean are in Appendix D, “Custom JavaBean Authentication”. The JavaBean ultimately returns a boolean value back to the Node, where 1 indicates that authentication was successful, and 0 indicates that it failed.

Note In a BrioONE deployment, Brio.Portal authenticates the user. A single sign-on is implemented so that the user can use OnDemand Server without having to enter their password again.

Selecting a Report

When a user selects a document to retrieve, the request goes to the Web Broker and then to the ODS Node. After verifying that the user is authorized to work with the selected document, the BQ daemon returns the Adaptive report mode (1-6), along with the location of the document, which can reside in the ODS file system or in the Repository:

- File System – If the physical document is stored on the local file system, it is available to the OnDemand Server 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 BQ 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 to the Node.

The Node collects the document, which asks the BQ daemon to add:

- Node’s hostname and port number
- Adaptive Report token to the document’s header

The Node forwards the document to the Web broker, which tags the document with the Brio MIME type so it can be launched by Insight or Quickview. The Web Broker then hands off the file to the Web server, which streams it to the Web browser via HTTP or HTTPS. The Web browser receives the data stream, interprets the MIME type information as a Brio document, and launches the version of Insight or Quickview resident with the browser. The document dynamically hides or exposes only the proper functionality for that adaptive state.

Note The ODS does not compress or expand documents. Documents are compressed by Brio Intelligence when Brio Intelligence is directed to compress a document. The ODS simply picks up the bits and streams them. If a document was saved compressed then it is streamed compressed.

Scheduling a Job That Posts to the Web

The procedure described in this section assumes that OnDemand Server and Broadcast Server are installed in the same database schema and the same database server. Installation and configuration instructions for Broadcast Server and OnDemand Server are on the Brio Intelligence distribution CD.

Note The user account for the BCS Repository is the same as that for the ODS Repository.

From Insight or QuickView, the user builds a job and selects the scheduling option from the file menu. From the Actions tab, the user sets the job to be saved, and specifies a directory

The user then marks the option to *Register to the OnDemand Server*, clicks the *Register* button, and fills in the tabs, completes scheduling the job, and checks the output from the job log.

Behind the scenes, the Broadcast Server (BCS) posts a job to the OnDemand Server. The job is written to the tables, including the command to register to the web. The Broadcast Server picks up the job. The job is returned to the user and the query is completed.

Once the file is completed, the BCS logs into ODS using a user name and password that are the same as those of an ODS user with rights to register to the Web. The BCS sends the ODS the job itself, and the adaptive state assigned to it when the user scheduled the job.

Note ADR (Automatic Distributed Refresh) does not work when using the Web clients, Freeview, Quickview, and Insight.

Summary

Using a combination of Java, JavaScript, HTML, and plug-ins, Brio Intelligence is designed to deliver rich, interactive reports within a Web environment. Within your current computing infrastructure, Brio Intelligence takes advantage of Web servers, database servers, and client computers that run a wide variety of operating systems. This chapter specifies what components run on each type of computer, details which communication protocols are used to transfer data between them, and describes how OnDemand Server components work together.

2 Broadcast Server Configuration and Administration

This chapter describes the *Brio Intelligence Server Administrator*, lists the configuration and administration tasks for Broadcast Server, and provides detailed instructions on how to use the Server Administrator to accomplish the administration tasks.

For complete configuration instructions, see the *Brio Intelligence Server Installation Guide* provided on the Brio Intelligence CD.

This chapter contains the following sections:

- Brio Intelligence Server Administrator
- Broadcast Server Configuration Quick Guide
- Broadcast Server Administration Procedures
- Using the Job.log File

Brio Intelligence Server Administrator

Brio Intelligence Server Administrator is a configuration and administration tool that you use to configure and administer both the Broadcast Server and the OnDemand Server. The Brio Intelligence Server Administrator is installed with the OnDemand Server and the Broadcast Server executable files, but it can also be installed separately.

The Brio Intelligence Server Administrator uses a conventional window-based graphical interface. The hierarchical structure or Administrative Tree Control allows the user to navigate to any item for either the Broadcast Server or the OnDemand Server. Once you select an item in the Administrative Tree Control (such as Processing Connections or Groups), you can add new options underneath the headings, modify existing options, or remove the option using the controls in the options panel on the right.

For help on basic windowing procedures or window-based computing environments, please consult on-line help topics in your Macintosh® or Windows™ operating system help files.

Remote Administration

Once each local machine is configured, OnDemand Server administration tasks can be accomplished remotely through the Server Administrator. Complex tasks such as OCE creation or modifying server configuration require that all drive mappings, directory names and locations, and Brio INI files are the same between the two boxes.

To remotely administer an NT-based ODS, install a remote access product such as PC Anywhere, Timbuktu, etc. and run the Server Admin utility that resides on the ODS box itself.

To remotely administer a Unix-based ODS, create a login session using a graphical telnet product such as Hummingbird.

However, because many BCS settings are stored locally in the BQServ1.ini file, you cannot use the Server Administrator to administer Broadcast Server remotely. For this reason, you may want to keep one copy of the Brio Intelligence Server Administrator on each Broadcast Server machine.

Broadcast Server Configuration Quick Guide

Initial configuration consists of setting the preferences, creating repository tables, and creating or adding a polling connection.

Table 2-1 provides an overview of the Broadcast Server configuration tasks. For complete configuration instructions, see the *Brio Intelligence Server Installation Guide* provided on the Brio Intelligence CD.

Table 2-1 Broadcast Server Configuration Quick Guide

Configuration Tasks	Purpose	More Information	Select From the Top Bar
Set File Locations	Stores default file locations for the Brio Intelligence connection, the directory for working on Brio Intelligence objects, the brioqry.exe, the job.log file, and the bqserver.exe	These default file locations may be changed	Edit→Preferences→File Locations tab
Set Date Formats	Determines the date format and timestamp format according to the locale selected	May be changed to any desired format	Edit→Preferences→ Formats tab
Set Service Settings	Displays the name of the server, sets the maximum concurrent jobs, and has a toggle switch to add/remove the Broadcast Server service	The higher the maximum number of jobs set, the more of your resources will be allocated to Brio Daemon(s)	Edit→Preferences→ Service tab
Set Email Options	For NT systems, allows selection of the email protocol: None, MS Exchange, MAPI 1.0 compliant, or Internet Mail	UNIX systems use the Internet mail (SMTP) protocol	Edit→Preferences→Mail tab

Table 2-1 Broadcast Server Configuration Quick Guide *(Continued)*

Configuration Tasks	Purpose	More Information	Select From the Top Bar
Set Transfer Settings	Sets the IP address and port number of the machine you are running the BCS on	Allows you to store jobs on the file system of the local machine or network drive	Edit→Preferences→ Transfer Settings tab
Add Polling Connections	Connects the BCS to databases in order to poll for pending jobs and to process queries	Add a polling connection for each database you plan to use as a job repository	From the Tree Control select: Polling Connections→Add Connection
Create Job Repository Tables	Creates job repository tables where a queue of jobs is stored	You are prompted to create job repository tables when you add a polling connection to a database with no repository tables	From the Tree Control select: Polling Connections→Add Connection→Create Repository

Note The maximum concurrent jobs is not a limit setting. It controls how much memory Brio should set aside for the results sets when it starts each concurrent job.

Broadcast Server Administration Quick Guide

Table 2-2 provides an overview of the Broadcast Server administration tasks. Perform the following administration tasks when necessary for your system.

Table 2-2 Broadcast Server Administrative Quick Guide

Administration Task	Purpose	More information	Administrative Tree Control Location
Create Database Connections	Creates new database connections	Create either repository or processing connections	(see "Creating an OCE File" on page 5-3)
Add Processing Connections	Connects the BCS to databases in order to process scheduled documents	Use processing connections to process documents on the database the job was created on	Polling Connections→ Name of Database→ Processing Connections (see page 2-7)
Create User Groups	Creates user groups that are associated with specific users, directories, and printers	Scheduling privileges are determined by group assignment	Polling Connections→ Name of Database→ Groups (see page 2-8)
Assign Users	Adds users to a specific user group	Do not add users if you are not using user groups	Polling Connections→ Name of Database→ Groups→Group Name→Users (see page 2-10)
Specify Output Directories	Selects output directories and/or FTP directories for use by a specific user group	Enables group members to save or export scheduled files to specific local or network directories	Polling Connections→ Name of Database→ Group→ Group Name→ Directories (see page 2-11)
Specify Output Printers	Selects printers for use by a specific user group	Enables group members to schedule reports for printing to specific network or local printers	Polling Connections→ Name of Database→ Group→Group Name→Printers (see page 2-13)

Table 2-2 Broadcast Server Administrative Quick Guide *(Continued)*

Administration Task	Purpose	More information	Administrative Tree Control Location
Create Custom Calendars	Creates calendars based upon the fiscal or other internal or organizational calendars	Jobs scheduled with a custom calendar resolves dates and variable date limits against the quarterly and monthly dates specified in the custom calendar, rather than the real calendar	Polling Connections→ Name of Database→ Custom Calendars (see page 2-14)
Create Event Triggers	Enables users to schedule jobs based on the completion of a defined event	If an event is disabled, jobs whose schedules are based on the event will not run	Polling Connections→ Database→Event Triggers (see page 2-15)
Monitor Job Lists	Provides run times and status details for each job submitted to a job repository	Logon using an Administrators' logon account to monitor the job lists	Polling Connections upon logon or From the Top Bar Select - Administration→Job List→Database Name (see page 2-21)

Broadcast Server Administration Procedures

After initial configuration use these instructions to administer your Broadcast Server.

Creating Database Connections

A connection file is a file that retains all the information necessary to logon to a specific configuration of database and connection API software. In addition, a connection file retains DBMS-specific connection preferences and even specifications for automatic access to metadata (Brio Intelligence connection files only).

The Database Connection Wizard steps you, the administrator, through the process of creating a connection file. The connection file makes the logon process easy by capturing connection parameters in a small file and allowing connection to a data source. The connection file enables a stable connection to be set up once, and then distributed and reused painlessly.

See page 5-3 for directions on using the Database Connections Wizard.

Adding Processing Connections

Broadcast Server uses processing connections to connect to databases and process scheduled documents. You can authorize many processing connections under each polling connection to permit users to schedule documents to a centralized job repository even when their documents are processed against different databases.

Broadcast Server connects to the processing database with a valid database login that is supplied when the job is scheduled, thereby enforcing database security.

Note Broadcast Server must have at least one connection to the database processing a scheduled document, or else the job will not run. Make sure you create or copy all the connection (OCE) files that Broadcast Server needs before adding processing connections.

To add a processing connection:

1. Select the **Processing Connections** heading in the Administrative Tree Control.
2. Click **Add Processing Connection**.
3. Navigate to the desired connection (OCE) file and click **Open**.

This procedure adds the processing connection to the Administrative Tree Control under the current polling connection.

To delete a processing connection select the processing connection you want to delete from the Administrative Tree Control and click **Remove Processing Connection** from the right panel.

Creating User Groups

Broadcast Server's User Groups feature associates lists of users, output directories, and printers with each user group. For each polling connection, Broadcast Server adds two default user groups to the Administrative Tree Control: Administrators Group and Public Group.

You can use these and additional custom groups to differentiate network privileges based on departmental mission, geographical location, or other criteria. Once assigned to a group based on the database user ID, a given user's group membership determines the output directories and printers available when scheduling a job.

Note The user groups feature does not enforce database security.

Administrators Group

Users added to the Administrators group can access all Broadcast Server resources and options when scheduling a Broadcast Server job. In addition, Administrators can monitor all jobs in a given job list. (Users view only the jobs scheduled under their own database user ID.)

The Administrators group should contain Broadcast Server's dedicated database user ID for the polling database as well as user IDs for those administrative or IT personnel who belong in the Administrators group.

Custom Groups

When a user schedules a Brio document, only the resources available to that user's group appear as scheduling options.To assign specific privileges to a group of users, add a custom group for them. You can add as many custom groups as you like.

Note You cannot assign a user to more than one Broadcast Server group. If a user requires unique privileges, you must create a separate group for that person.

Public Group

Users not assigned to a custom group can view resources and schedule documents with the privileges of the Public group.

- If you do not want to establish different levels of privilege, use the Public group.
- If you do not want to provide general access to Broadcast Server, delete the Public group. This will create secured access by ensuring that only administrators and members of a custom group have access to Broadcast Server.

To create a group:

1. Select the **Groups** heading in the Administrative Tree Control and click the **Add Group** button.
2. Enter a **name for the group** in the Group Name field.
3. Click the **Groups** header to add another group.

The new group appears in the Administrative Tree Control.

You can select existing groups and rename them using the same procedure.

To delete a user group select the group you want to delete from the Administrative Tree Control and click **Remove Group** on the right panel.

Assigning Users to a Group

For each group (including the Administrators group), you need to assign user members who match the group's resources.

Note This step assumes you are implementing the User Groups feature. If you are not adding custom groups, skip this step and add default directories for the Public group.

To assign a user to a group:

1. Select the **Users** heading in the Administrative Tree Control under the user group to which you want to add this user.
2. Click **Add User** in the right panel.

   The Database logonID and User Name fields appear.
3. Enter a **database user ID** and optional **descriptive name.**
4. Click the **Users** heading or the **Apply All button** at the bottom of the right panel.

   The user appears in the Administrative Tree Control under the specified group.

Note User names and passwords are case-sensitive for certain databases.

Specifying Output Directories

Directory picklists enable group members to save or export scheduled files to local or network directories. You can also use this feature to provide access to intranet or FTP directories.

Note Be sure to grant the BEServer account network write access to all destination directories.

To specify an output directory:

1. Select the **Directories** heading under a group in the Administrative Tree Control.
2. Click **Add Directory** in the right panel.
3. Enter a **descriptive alias** for the end user in the Directory Alias field and a complete **network path** to the directory in the Directory Path field.
4. Click the **Directories** heading in the Administrative Tree Control or click the **Apply All** button on the lower part of the right panel to add the directory.

   The directory appears in the Administrative Tree Control under the group specified.

   You can select existing directories and change specifications using the same procedure.

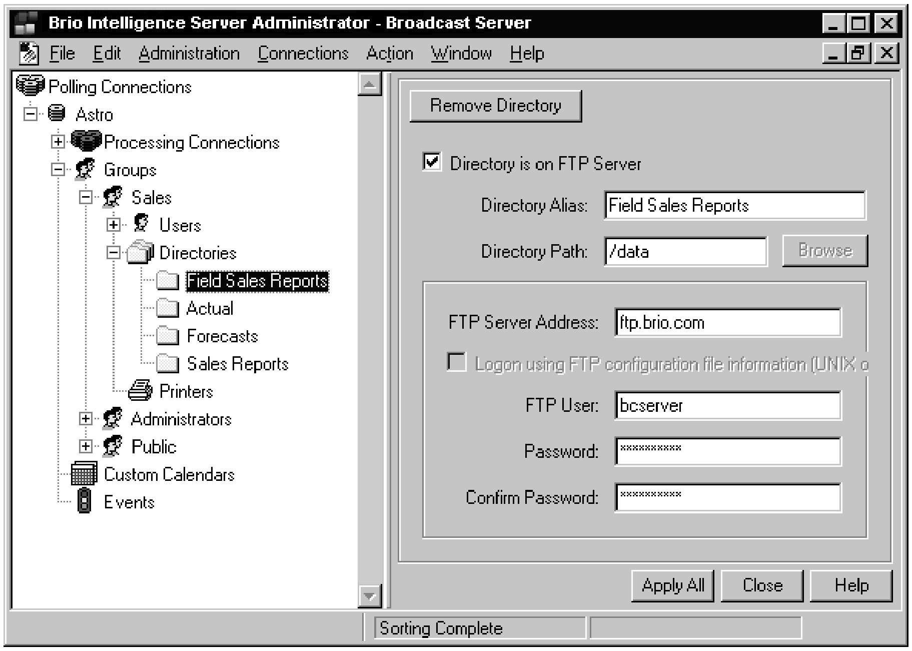
FTP Directories

You can also add FTP server directories as output options for Broadcast Server.

To add an FTP directory:

1. Select the **Directories** heading under a group in the Administrative Tree Control.
2. Click **Add Directory** in the right panel.

3 Click **Add a Directory** and click **Directory is on FTP Server** to activate the options in the FTP area.



☑ Directory is on FTP Server

Directory Alias: Field Sales Reports

Directory Path: /data

FTP Server Address: ftp.brio.com

☐ Logon using FTP configuration file information (UNIX o

FTP User: bcserver

Password: ***********

Confirm Password: ***********

4 Enter a **descriptive alias** in the Directory Alias field and a **pathname** in the Directory Path field.

Note On a FTP server, the pathname is relative, not absolute, so do not include the drive name in the path. Separate each directory in the path with a slash '/' rather than a backslash '\'.

5 Enter the **FTP Server Address**.

6 Enter a **FTP User** and **Password** for Broadcast Server to connect to the FTP server. Make sure the FTP logon account you use has write access to the target FTP directory.

Check the job.log file for FTP processing errors. See "Job.log File" on page 2-25.

To check the FTP connection:

1 Place the following commands in a script file:

```
open (ftp server name)
user
(userid) (password)
cd (remote ftp directory name)
lcd (Broadcast Server's working directory)
verbose
binary
prompt
put (name of file from BCS working directory)
quit
```

2 Run the file from the command line using the following syntax:

```
ftp -n -s:<script filename>
```

Specifying Output Printers

Printer picklists enable group members to schedule reports for printing to network or local printers.

Note **NT Users Only** – Make sure that drivers for each output printer are installed on the NT server machine and that each output printer is installed to the NT Printers Control Panel.

Note **UNIX Users Only** - Make sure that it is possible to print from the machine where BCS is installed. If the print server is not accessible add this capability to the machine, for example, use the "admintool" (as "root" in Solaris).

To specify a printer:

1. Select the **Printers** heading under a group in the Administrative Tree Control.
2. Select the printers you want to add from the Printers right panel.

   If no printers appear, make sure the printers are configured through the Control Panel (Windows NT).
3. Click **Add Printer** to add them as job output destinations.

Creating Custom Calendars

You can create custom calendars to schedule jobs based on fiscal or other internal or organizational calendars. Jobs scheduled with a custom calendar resolve dates and variable date limits against the quarterly and monthly dates specified in the custom calendar, rather than against the real calendar. You can add multiple year versions of a calendar, such as Fiscal 1999 and Fiscal 2000, or create different, overlapping calendars for departmental schedules.

Note Brio Intelligence Server assumes that dates with a year value of less than 30 are in the twenty-first century; thus, the 6-digit date 06/01/01 is translated to the 8-digit date 06/01/2001. If you have set default date formats to read the year in a different position, such as 00/11/31, the date is translated to 2000/11/31.
If this method is not suitable for your site, you have the option of entering all dates in 8-digit format instead of 6-digit format.

To create a custom calendar:

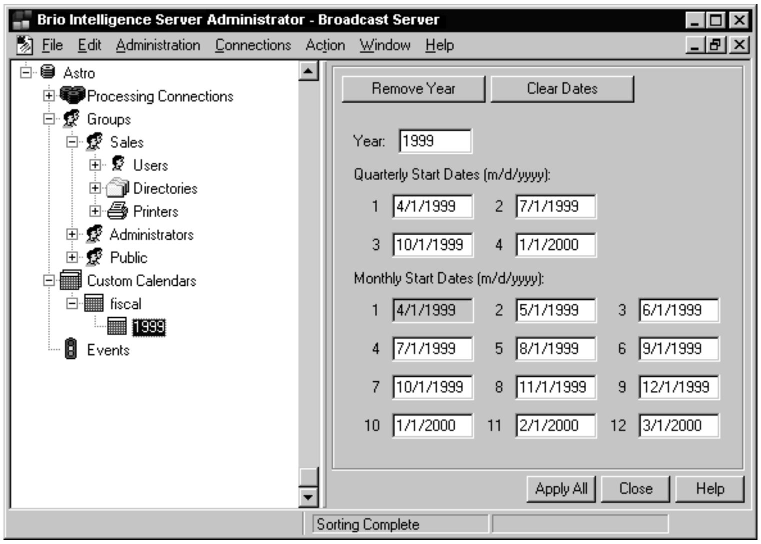
1. Select the **Custom Calendars** heading in the Administrative Tree Control and click **Add Calendar**.
2. Enter a name for a calendar type (for example, fiscal or model year) in the Custom Calendar field and click **Add Year**.

   Empty date fields for a calendar appear in the panel.
3. Enter a start date for Quarter 1 using month/day/year format, such as 06/01/99 or 03/15/00 (or 06/01/1999 or 03/15/2000, if you prefer), then click in another field.

   Server Administrator automatically adds the remaining dates to the calendar.

4 If necessary, edit any of the date fields to fine-tune the calendar.

5 Click the **name of a calendar** or the **Custom Calendar** heading to add more calendars.



Year: 1999

Quarterly Start Dates (m/d/yyyy):

1	4/1/1999	2	7/1/1999
3	10/1/1999	4	1/1/2000

Monthly Start Dates (m/d/yyyy):

1	4/1/1999	2	5/1/1999	3	6/1/1999
4	7/1/1999	5	8/1/1999	6	9/1/1999
7	10/1/1999	8	11/1/1999	9	12/1/1999
10	1/1/2000	11	2/1/2000	12	3/1/2000

Sorting Complete

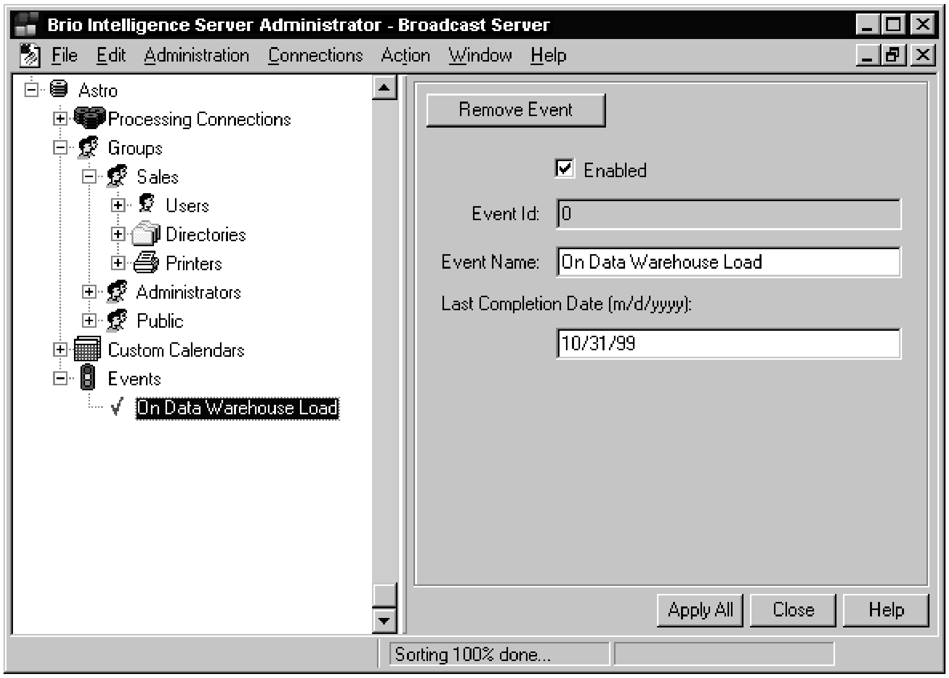
Creating Event Triggers

You can also define event triggers, which enable users to run jobs based on calendar dates.

When scheduled to run based on an event, the Broadcast Server checks the job repository to see if the event has completed since the most recent job run. If so, the job runs either on the next poll or at a date and time specified by the user, depending on what the user has specified.

To provide event-driven scheduling for end-users:

- Define each event in Server Administrator.
- Update the completion date in the BRIOEVNT table for all jobs triggered by the event each time the event occurs.



☑ Enabled

Event Id: 0

Event Name: On Data Warehouse Load

Last Completion Date (m/d/yyyy): 10/31/99

You can send or automatically trigger SQL UPDATE statements, or you can use Server Administrator to enter dates manually in the COMPLETION_DATE field as they occur.

To add an event trigger:

1 Select the Events heading in the Administrative Tree Control and click **Add Event.**

2 Enter a **descriptive name** for the event in the Event Name field.

3 Enter a **last completion date** if you want to trigger the event.

You can also return to Server Administrator at any time to enter a completion date manually, or to disable an event trigger.

4 Click the **Event heading** to add further events.

To update the status of an event manually once it has completed, change the last completion date.

Note If an event is disabled, users do not see it when they schedule jobs, and jobs with schedules based on a disabled event will not run, even if the Last Completion Date is updated.

Starting Broadcast Server on Windows NT Systems

Before the preceding administration modifications can take effect, you need to stop the Broadcast Server service, use the Windows NT Services Control Panel to log on and start Broadcast Server. You can then begin to poll the job repository tables.

To start the Broadcast Server:

1. Open the Windows NT Services Control Panel.

   The control panel appears, listing the Broadcast Server service.

2. Select the **Broadcast Server** service and click Startup.

   The Service dialog appears.

3. Under Startup Type, select **Automatic**, to start the NT service automatically when the NT server machine is booted.

4. Log on as **This Account**, using the same domain user account (such as BEServer) and password you set up, and click **OK**.

Note Do not use the System Account to log onto Broadcast Server.

5. Click **Start** in the Services control panel to start the NT service.

6. Close the NT Services Control Panel and use the File Manager to move to the Broadcast Server working directory. Open the service log file (BQServer.log) and check to ensure that the Broadcast Server has started and is polling the job repository.

Note If Broadcast Server fails to connect and poll, make sure the database connection software is included in either the system path environment variable or of the BEServer account's path environment variable. Refer to *Adjust network and application settings* in the *Before Installing Intelligence Server* section of the *Brio Intelligence Server Installation Guide* provided on the Brio Intelligence CD.

Starting Broadcast Server on UNIX Systems

Before the preceding administration modifications can take effect to start Broadcast Server and begin polling the job repository tables.

To start the Broadcast Server:

1 Move to the Broadcast Server working directory.

```
# cd <server_install_directory>/bin
```

2 Run the Brio Broadcast Server Startup script.

```
% bqs (or optional script % bqs.start)
```

3 Open the service log file, `bqserver.log,` (located in the *working* directory) and confirm that the Broadcast Server has started and is polling the job repository.

UNIX Startup Scripts for the Broadcast Server

Three start up scripts for UNIX users.

Sample Startup Script for HPUX

Create a script similar to the following in the `/sbin/rc3.d` directory.

```
-----------------------------

#!/bin/sh

# Brio Startup Script for HP

# set -x # Uncomment for debugging

# Setup Environment (Make site-specific changes here)

UNIX95=true
export UNIX95
USER=brio6x
```

```
PATH=.:/usr/home/brio6x/bin:/bin:$PATH
ORACLE_HOME=/local/oracle8
SHLIB_PATH=/usr/home/brio6x/lib:/usr/lib
HOME=/usr/home/brio6x

export DISPLAY USER PATH SHLIB_PATH HOME ORACLE_HOME
# Start the X Virtual Framebuffer (Xvfb)
if [ -x /usr/X11R6/Xvfb ]; then
echo "Starting up the Virtual Frame Buffer on Screen 1 ..."
/usr/X11R6/Xvfb :1 -fbdir /tmp >> /var/adm/Xvfb.log &
DISPLAY=‘hostname‘:1.0
export DISPLAY
fi

# Start the Brio Broadcast Server
if [ -x $HOME/610/bin/bqs ]; then
echo "Starting up Brio BCS using Xvfb Screen 1 ..."
# Uncomment the following line if $USER runs the c-shell
#su $USER -c "cd $HOME/bin; setenv HOME $HOME; bqs.start >
/dev/null"
# Uncomment the following line $USER uses any other shell
su $USER -c "cd $HOME/bin; HOME=$HOME; export HOME; bqs.start >
/dev/null"
fi
exit 0
```

Sample Startup Script for Solaris

Create a script similar to the following in the `/etc/rc3.d` directory.

```
-----------------------------

#!/bin/sh

# Example Brio Startup Script for Solaris

# Setup Environment (Make site-specific changes here)

XvfbPath=/usr/X11R6
USER=brio6x
ORACLE_HOME=/oracle8
HOME=/usr/users/brio6x

export XvfbPath USER ORACLE_HOME HOME

# Start the X Virtual Framebuffer (Xvfb)
if [ -x $XvfbPath/Xvfb ]; then
echo "Starting up the Virtual Frame Buffer on Screen 1"
$XvfbPath/Xvfb :1 -fbdir /tmp >> /var/adm/Xvfb.log &
DISPLAY=‘hostname‘:1.0
export DISPLAY
fi

# Start the Brio Broadcast Server
```

```
if [ -x $HOME/bin/bqs.start ]; then
echo "Starting up Brio Broadcast Server using Xvfb Screen 1"
# Execute bqs.start as the Brio User, not as root
su $USER -c "cd $HOME/bin > /dev/null
./bqs.start > /dev/null"
fi
```

Sample Startup Script for AIX

Create a script similar to the following in the `/etc/` directory. After you create the script, add the following line to the end of the `/etc/inittab` file

```
brio:2:wait:/etc/rc.brio > /tmp/brio.out 2>&1

-----------------------------

Brio startup script for AIX

cat /etc/rc.brio

#!/bin/sh

# Brio Startup Script

DISPLAY=sunbird:0.0
export DISPLAY

# Start the Apache Web Server
echo "Starting the Apache Web Server"
/external/apache/bin/apachectl start >/dev/null 2>&1

USER=brio6x
HOME=/external/home/brio6x
BRIOHOME=/external/home/brio6x/610
IBPATH=/external/home/brio/lib

export USER HOME BRIOHOME

# Start Broadcast Server
if [ -x $BRIOHOME/bin/bqs.start ]; then
echo "Starting up Brio Broadcast Server"
# Execute bqs.start as the Brio User, not as root
su $USER -c "cd $BRIOHOME/bin > /dev/null
./bqs.start > /dev/null"
fi
exit 0
```

Monitoring Job Lists

The Job List provides run times and status details to catalog each job submitted to a job repository.

The job list appears when you connect to a job repository through a polling connection in Brio Intelligence Server Administrator or when you click on Administration/Job List/polling connection from the top bar.

Modify Jobs

You can modify the general or schedule details of a job through the job list.

To modify a job:

1. Click **Administration** from the top bar. Select **Job List** from the drop-down list. Then click on a polling connection.

   The Job List screen for that polling connection appears. See page 2-22 for the field definitions of the Job List Monitor.

2. Double click on the job you want to modify or single click on the job and click **Modify Job**.

   The Job Detail panel appears showing the General tab. See page 2-22 for the field definitions on the General tab. Change the fields as necessary for your job.

3. Click the Schedule tab to modify the schedule or to set priority and click **OK**.

   See page 2-23 for job scheduling details. The priority field is used only when more than one job is scheduled for the same time.

Run Job at Next Poll

You can run a job when needed without changing the job's regular schedule. (This feature is not available for jobs with ASAP specified as the run interval.)

➤ To run a job at next poll click Run Next Poll.

The "Completion Status" of the job will change to Run ASAP. As the job runs the completion status will change to what is appropiate for the moment.

To refresh a Job Information List click **Refresh**.

Table 2-3 Job List Field Definitions

Job List Fields	Definitions
Job ID	A unique job code assigned by Brio Intelligence. Each job code remains the same for the life of the job.
Enabled	Indicates status of a job. A disabled job is discontinued (but remains in a job list) until re-enabled.
Priority	Shows the rank of the scheduled job based on a numerical value. This selection is only available for jobs scheduled to run simultaneously and takes precedence over the job identification number. A job assigned with a high priority number (for example, 100) is run before a job with a low priority number (for example, 50). If two jobs are run simultaneously and share the same priority column number, then Brio Intelligence will rank according to each job’s unique job id.
User Name	The database user name of the user scheduling the job.
Job Name	Descriptive name for the job.
Last Date	Date of the previous run.
Completion Status	Status of the previous (or current) job run, for example, 'successful,' ‘failure,’ or ‘running’.
Next Date	Date of the next iterative run.
Execution Time	Scheduled run time for each iteration.
Job Interval	Time between iterative runs.
Repetitions	Number of iterative run cycles assigned.

Table 2-4 General Job Field Definitions

General Job Fields	Definitions
Enabled	Toggles the job. If this box is unchecked, the job is disabled and will not run until it is re-enabled. A disabled job remains in the job list.
Job Id	A unique job code assigned by the server. A job code remains the same for the life of the job.
User Name	The name of the user that scheduled the job.
Job Name	A descriptive job name which describes the job in the Job List.

Table 2-4 General Job Field Definitions

General Job Fields	Definitions
Document	The name of the Brio Intelligence document (.bqy) that is scheduled.
Calendar	A custom-created calendar for use in scheduling the job. Calendars are provided by the Broadcast server administrator, and can be used to schedule a job by alternate or fiscal calendars.
Send email notification on completion?	Requests an email confirmation on completion of each job run. If client and server run different email systems, the email address string entered in the field must match the mail protocol used by the server, not the addressee.
Address	An email address to send the log file to.
Address Book	Displays email address book (MAPI-compliant email systems only).
Check Name	Pings the email address to check a connection. There is no reply if the address checks. (MAPI-compliant email systems only).

Table 2-5 Job Schedule Details

Job Schedule Details	Parameters
Daily	Check a box for each day the report should run.
Weekly	Select the day on which to run the report.
Monthly or Quarterly	Select an ordinal dates from the first pop-up. Select Nth and enter a numeric date to run the report on any day other than the first or last day of the period. Specify a weekday or the generic "day" to indicate the interval.
Every	Enter a starting date and incremental value in the Every text field. Choose an incremental period unit from the popup.
Event-Based	Select an event-trigger. Enter a specific date on which to run the job once the event is completed.
ASAP	Select ASAP (as soon as possible) to execute the job at the next poll.

Table 2-5 Job Schedule Details *(Continued)*

Job Schedule Details	Parameters
Priority	Enables you to rank scheduled jobs based on a numerical value representing the rank priority of the job. This selection is only available for jobs scheduled to run simultaneously and takes precedence over the job identification number. A job assigned with a high priority number (for example, 100) is run before a job with a low priority number (for example, 50). If two jobs are run simultaneously and share the same priority column number, then Brio Intelligence will rank according to each job's unique job id.
Time to Execute (HH:MM:AM)	Enter a run time (colon delimited) and a time-of-day to execute the job.
Upon completion of	If you run a job based on an event, check this field and select the event from the pull-down list. When a job is scheduled to an event, Broadcast Server checks to see if the event was completed after the most recent job run, and if so, the job runs at a date and time specified by the user.
Number of Executions/ Infinite	Enter the number of times the job should run or Infinite to run the job indefinitely.
Time Threshold and Always Run	The time threshold is a grace period that follows the scheduled run time. If capacity processing postpones the job beyond the threshold, the job is shelved until the next scheduled run. Click Always Run to run the job regardless of how long it is postponed.

Job.log File

The Broadcast Server job.log file collects information about a job as it is running. The location of the job.log file, in the working directory, is set in the Brio Intelligence Server Administrator by selecting `Edit-->Preferences-->File Locations` tab. The job.log file contains the following information:

- Time and Date the job started processing
- Limits set on the query(ies)
- OCE(s) the server connected to process the query(ies)
- SQL statement(s) the server sent to the database(s) for each query in the BQY document
- Number of rows of results returned from the database for each query
- Each individual action that the server performed for the job
  - Section the action was performed on
  - Destination of the file(s)
  - When the action started and ended
- Error Messages from parts of the job that failed
- Time and date the job completed
- If the job completed with or without errors.

Note If the notification option is turned on the job.log file gets notified.

Summary

This chapter summarizes the configuration and administration tasks and describes how to do the administration tasks using the Brio Intelligence Server Administrator for the Broadcast Server.

3

OnDemand Server Configuration and Administration

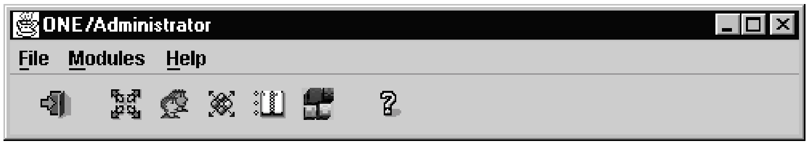
This chapter lists the configuration and administration tasks for the OnDemand Server and provides detailed instructions on how to accomplish the administration tasks either through the Brio Server Administrator (for a description of the Brio Server Administrator, see page 2-2) or by modifying the ODS.ini file.

For complete configuration instructions, see the *Brio Intelligence Server Installation Guide* provided on the Brio Intelligence CD.

This chapter contains the following sections:

- OnDemand Server Configuration Quick Guide
- OnDemand Server Administration Quick Guide
- OnDemand Server Administration Procedures
- ODS.ini File
- ODSCGI.ini File for UNIX Systems
- Log Files
- Process Multiplier
- Multiple GET Requests from Internet Explorer

Note In a BrioONE deployment, the OnDemand Server Administrator is launched from ONE/Administrator.



Click to launch the Brio OnDemand Server Administrator.

OnDemand Server Configuration Quick Guide

During initial configuration set the basic settings and *load balancing*. You can modify these settings later as necessary.

The OnDemand Server configuration tasks are listed in Table 3-1. For complete configuration instructions, see the *Brio Intelligence Server Installation Guide* provided on the Brio Intelligence CD.

Once each local machine is configured, OnDemand Server administration tasks can be accomplished remotely through the Server Administrator. See page 2-2 for information on OnDemand Server remote administration.

Table 3-1 OnDemand Server Configuration Quick Guide

Configuration Task	Purpose	More information
Set Load Balancing	Distributes processing demands across all available network resources	This is not necessary if your ODS is installed on one machine
Define a Repository Connection	Defines the OCE file	
Define a Connection to ODS	Defines the connection to ODS	
Create Repository Tables	Creates repository tables for the current database	Will be prompted to create repository tables when trying to connect to a database that does not have repository tables
Add a New Server	Adds an instance of the ODS	
Configure Basic Settings	Configures the number of Brio daemons to use and the maximum amount of RAM to allocate to processing	If a query runs out of available RAM, it will use swap space on the local machine's RAM disk

Table 3-1 OnDemand Server Configuration Quick Guide *(Continued)*

Configuration Task	Purpose	More information
Select an Authentication mode	Determines which authentication mode to use for authorizing users to logon to ODS	You must be connected to ODS to set or modify the authentication mode
Add Directories	Stores default file locations for the directories of OnDemand registered documents file, Brio daemon executable file, temporary files, and Brio Connection files	You must be connected to ODS to set or modify the default file directories
Store Server Repository Information	Identifies connection, password and attribute information for the repository database	You must be connected to ODS in order to set or modify the server repository information

OnDemand Server Administration Quick Guide

The following administration tasks can be done whenever an addition or modification is necessary.

Table 3-2 OnDemand Server Administration Quick Guide

Administration Task	Purpose	More Information	Administrative Tree Control Location
Create Database Connections	Creates new database connections	Create either repository or processing connections	Servers→Add→ Create→OCE Wizard (see page 5-3)
Add Processing Connections	Enables the ODS to connect to databases	Multiple processing connections can be authorized under each ODS repository connection	Server name→ Processing Connections→ Add (see page 3-6)
Authorize Users	Authorizes new users to use OnDemand Server, or authorizes existing users of the Brio Intelligence Repository to use OnDemand Server	Authorize new users according to the authentication mode selected for this instance of ODS	Server Name→Modify→ Authentication tab or User (see page 3-7)
Assign Group Privileges	Assign users to groups, assigns groups a default privilege level, and then associates groups with registered documents and Repository Models		Server Name→ Groups→ Add (see page 3-9)
Add Folders	Adds additional folders	You can add nested folders as necessary to organize documents	Server Name→Document→New Folder (see page 3-11)
Register Documents and Repository Models	Registers standard Brio Intelligence documents and repository objects for access through ODS	Registers documents and repository models to groups	Server Name→Add Document (see page 3-11)
Access your OnDemand Server	Logs on to the OnDemand Server		Server Name→Logon (see page 3-15)

OnDemand Server Administration Procedures

After initial configuration, use these instructions to administer your OnDemand Server.

Creating Database Connections

A connection file is a file that retains all the information necessary to logon to a specific configuration of database and connection API software. In addition, a connection file retains DBMS-specific connection preferences and even specifications for automatic access to metadata (Brio Intelligence connection files only).

The Database Connection Wizard steps you, the administrator, through the process of creating a connection file. The connection file makes the logon process easy by capturing connection parameters in a small file and allowing connection to a data source. The connection file enables a stable connection to be set up once, and then distributed and reused.

See page 5-3 for directions on using the Database Connections Wizard.

Adding Processing Connections

OnDemand Server uses its processing connections to connect to databases and refresh documents as directed by the user. Multiple processing connections can be authorized under OnDemand Server's repository connection, enabling all documents to be stored in a centralized repository even when they are processed against different databases.

Note Brio recommends that you use two different connection files, one for processing documents and one for connecting to the repository tables. See Chapter 5, "Connection or OCE Files".

To add a processing connection:

1 Right-click the **Processing Connections** heading in the Administrative Tree Control, and click **Add**.

A dialog box that allows you to select a connection from OnDemand Server's Open Catalog Extension directory appears. These connections are stored on the ODS machine. (Create these connections using the Database Connection Wizard. See page 5-3 for instructions on using the Database Connection Wizard.)

2 Navigate to the desired connection file (.oce) and click **Open**.

The Add dialog box appears.

3 Enter the **user name and password** to access your data on this enterprise database.

The processing connection is now available for use by Brio Intelligence (.bqy) documents on the Add Document-Connections tab.

Authorizing Users

There are three different user authentication modes: OnDemand Server Authentication, Database Authentication, and Custom JavaBean Authentication. See page 1-21 for a complete description of the authentication modes.

OnDemand Server or JavaBean Authentication

If you are using the OnDemand Server for Authentication—or if you are using JavaBean authentication—you need to specify the user names individually. User names and passwords are case-sensitive for certain databases.

See Appendix D, "Custom JavaBean Authentication" for instructions on how to write a JavaBean.

To authorize users:

1 Right-click on **Users** in the Administrative Tree Control, and select **Add**.

2 Enter the **username** and **password** for the person you wish to add in the dialog box and then click **OK**.

The new user is added to the Users Administrative Tree Control.

Note In a BrioOne deployment, users are defined using ONE/Users and the Authentication tab is not available.

Note Be sure the *Allow access to OnDemand Server* checkbox is checked; turning that option off gives the user access to the Brio Intelligence Repository without enabling access to the larger OnDemand Server Repository.

Database Authentication

Users are authenticated based on their database passwords.

To add new users:

1 Select **Users** under the server where you want to add users and then click **Add**.

The Configure Users dialog appears.

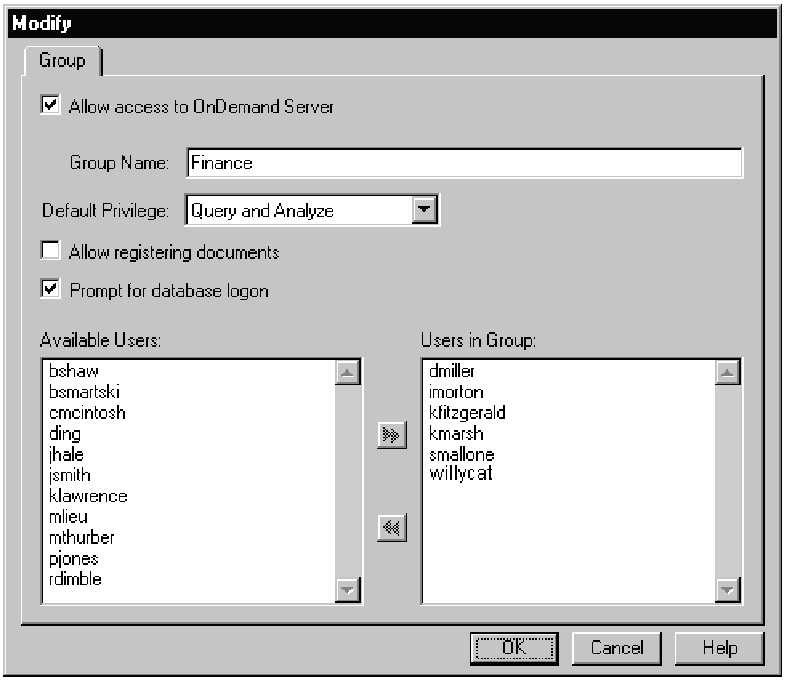
2 Select **users** on the Database Users list to transfer them to the Selected Users list and click **OK** when all users are transferred.

Assign Group Privileges

Individual documents are registered to OnDemand Server groups, each of which contains one or more users. Each OnDemand Server has a PUBLIC group, which automatically contains all users.

You can assign users to groups, assign groups a default privilege level (adaptive state), and then associate groups with registered documents.

To add additional groups, select **Groups** in the Administrative Tree Control, and click **Add**.


Modify

Group

☑ Allow access to OnDemand Server

Group Name: Finance

Default Privilege: Query and Analyze

☐ Allow registering documents

☑ Prompt for database logon

Available Users: bshaw, bsmartski, cmcintosh, ding, jhale, jsmith, klawrence, mlieu, mthurber, pjones, rdimble

Users in Group: dmiller, imorton, kfitzgerald, kmarsh, smallone, willycat

Note In a BrioONE deployment, users are assigned to groups using ONE/Users and this screen is not available. The default privilege for all groups is “View Only”.

Elements in the Modify or Add a Group dialog box:

Allow access to OnDemand Server	This checkbox identifies the group as having access to the ODS. Generally this checkbox should be left enabled.
Group Name	Any descriptive name you give the group. It should be 30 characters or less.
Default Privilege	Use this popup to define the default privilege level for documents registered to this group. Members of this group are authorized to see each document in the group unless you specifically override the privilege level for one or more individual documents.
Allow Registering Documents	Use this checkbox to allow members of the group to register documents to the OnDemand Server from within Brio Intelligence. Generally, only groups of administrators or power users should have this capability.
Prompt for Database Logon	Use this checkbox to define the default behavior for documents registered to this group. If prompting is enabled, users are prompted to enter a database username and password for each request to process a document. If prompting is disabled, when using Database authentication, the ODS passes the user's username and password to the database as logon parameters. When using ODS authentication, the ODS automatically logs on to the database using the username and password that the administrator used when originally creating the processing connection.
Available Users and Users in Group	Use these list boxes to identify which users are part of this Group. Double-click usernames to move them back and forth between the two lists.

Adding Folders

You can nest an unlimited number of OnDemand Server documents into folders and present them to users in a hierarchical format.

To create a nested folder:

1 Right-click on the **folder** under the Documents heading in the Administrative Tree Control where you want to nest the new folder, and select **Add Folder**.

2 Enter a **unique name** and **description** for the folder.

To create additional folders within folders, repeat these steps.

Note In BrioOne, folders are created in One/Publisher.

Registering Documents and Repository Models

OnDemand Server serves both repository models and local documents through the Web.

When users connect to OnDemand Server with a browser, they choose from a catalog of documents and repository models that the administrator provides.

- Documents are standard Brio Intelligence documents registered for use with OnDemand Server. A document may contain a full results set as well as a Data Model and reports. Documents are stored in a documents directory and located through pointers placed in the repository.
- Repository models are stored repository objects registered for use with OnDemand Server. OnDemand Server can serve only those repository models stored in its own Brio Intelligence Repository. Repository models are physically stored in the repository.

To register documents and repository models:

1. Right-click on a **folder** in the Administrative Tree Control and choose **Add Document**. To register the object, click the appropriate button.

   You are prompted to register either a local document or a repository model from the Repository.

2. For repository models, choose an available model to register, and skip the next two steps.

3. For documents, you are prompted to specify the source drive.

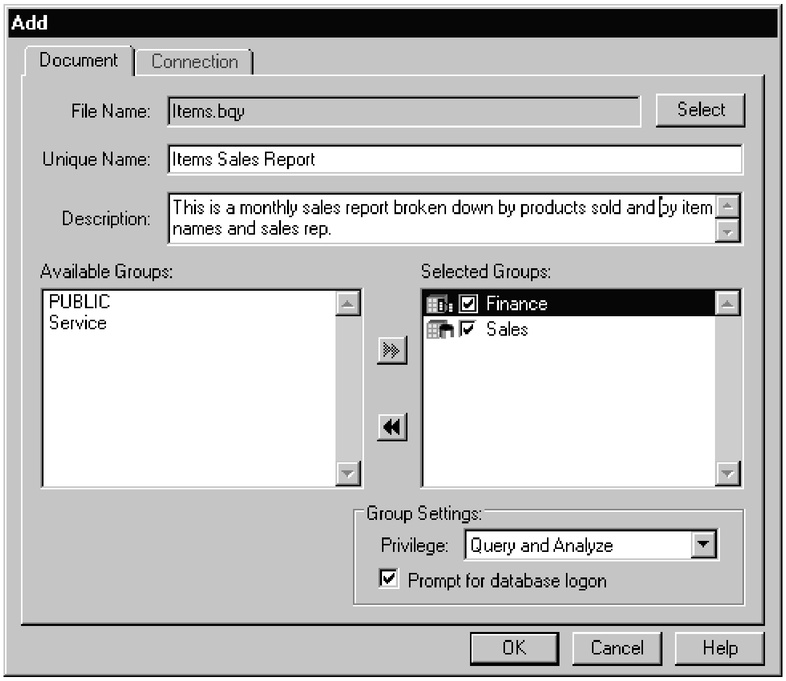
4. Select the appropriate **source drive**:

   - Choose From OnDemand Server to select a document from the Documents directory on the OnDemand Server machine. Use this option to register any documents you have gathered and placed in the directory.
   - Choose From Local Disk to select from a network drive. Use the File Open dialog box to browse to specific Brio Intelligence documents you would like to register. Once you select a file, Brio Enterprise Server Administrator transfers it using TCP/IP to the OnDemand Server Document directory.

5. Enter **preferences** for the registered object:

   - For documents, enter a unique display name for users to identify the document in the registry. This identifier must be unique across the entire system.
   - Enter a description to identify the contents of the document.

6 Add or remove groups from the **Selected Groups** panel to grant access to the object. Each selected group appears in the panel with an icon that indicates the default access privilege that members have to the object.



Add

File Name: Items.bqy

Unique Name: Items Sales Report

Description: This is a monthly sales report broken down by products sold and by item names and sales rep.

Available Groups: PUBLIC, Service

Selected Groups: ☑ Finance, ☑ Sales

Group Settings:

Privilege: Query and Analyze

☑ Prompt for database logon

7 To override default settings for the current document only (if desired):

- Select a group or groups.
- Select an override privilege for the object using the Privilege popup.
- When users of the group view this object, their normal default privileges will be changed to reflect the override privilege.
- Click Prompt for database logon to require that users supply a database user name and password to log on to the database when refreshing data or running a query.

8 Move to the **Connections Tab** to specify which processing connections to use to process the document.

9 For each query or data model section in the document, select a **processing connection** from the connection list.

If the processing connection is not available, create or copy a connection file to the default connections directory, and add it as a processing connection.

10 Repeat as necessary to register all the documents and repository models you want to make available.

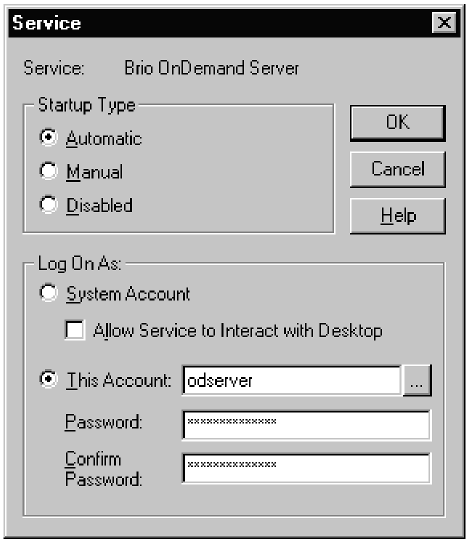
Note Removing documents deletes them from the registry. Removing a repository model un-registers the model. To remove the model from the Brio Intelligence Repository, first run Brio Intelligence, then choose Tools–>Administer Repository, and delete the model.

Configuring the NT Service for Brio OnDemand Server

On Windows NT systems, you must configure the service to run as the BEServer account before these modifications can take effect.

To configure the NT service:

1 Open the Windows NT Services Control Panel.



Service: Brio OnDemand Server

Startup Type: [✓ Automatic] Manual Disabled

Log On As: System Account

Allow Service to Interact with Desktop

[✓ This Account:] odserver

Password: ***************

Confirm Password: ***************

2 Select the **OnDemand Server** service and click **Startup**.

3 On the Service dialog, select the **Automatic Startup Type** setting to start the NT service automatically when the NT server machine is booted.Click **Startup**.

4 Log on as **This Account**, using the BEServer domain account name and password you set up earlier, and click **OK** (see Create a Network Account for the BEServer on the CD).

5 Click **Start** in the Services control panel to start the NT service.

Note Always use the BEServer account (or the domain user account you have set up for this purpose if you have given it a different name). Running server components with the System account will cause problems.

Testing the OnDemand Server Web Site

Your OnDemand Server is now ready to deliver documents to your end users.

Windows NT

To access your OnDemand Server:

1 Enter the following URL in a Web browser:

```
http://<ondemand_server_hostname>:<port>/odshtml/brio.html
```

This displays the OnDemand Server startup page.

2 Click the **logon icon** on the top center of the page to bring you to a dialog box to enter your **username and password**.

Once you have logged on, a document list appears.

Sample Startup Script for HPUX

Create a script similar to the following in the `/sbin/rc3.d` directory.

```
-----------------------------

#!/bin/sh

# Brio Startup Script for HP

# set -x # Uncomment for debugging

# Setup Environment (Make site-specific changes here)

UNIX95=true
export UNIX95
USER=brio6x
PATH=.:/usr/home/brio6x/bin:/bin:$PATH
ORACLE_HOME=/local/oracle8
SHLIB_PATH=/usr/home/brio6x/lib:/usr/lib
HOME=/usr/home/brio6x

export DISPLAY USER PATH SHLIB_PATH HOME ORACLE_HOME
# Start the X Virtual Framebuffer (Xvfb)
if [ -x /usr/X11R6/Xvfb ]; then
echo "Starting up the Virtual Frame Buffer on Screen 1 ..."
/usr/X11R6/Xvfb :1 -fbdir /tmp >> /var/adm/Xvfb.log &
DISPLAY=‘hostname‘:1.0
export DISPLAY
fi

# Start the Brio OnDemand Server
if [ -x $HOME/server/ods.start ]; then
echo "Starting up Brio ODS ..."
su $USER -c "cd $HOME/server; nohup ods.start > /dev/null"
fi
exit 0
```

Sample Startup Script for Solaris

Create a script similar to the following in the `/etc/rc3.d` directory.

```
-----------------------------

#!/bin/sh

# Example Brio Startup Script for Solaris

# Setup Environment (Make site-specific changes here)

XvfbPath=/usr/X11R6
USER=brio6x
ORACLE_HOME=/oracle8
HOME=/usr/users/brio6x
```

```
export XvfbPath USER ORACLE_HOME HOME

# Start the X Virtual Framebuffer (Xvfb)
if [ -x $XvfbPath/Xvfb ]; then
echo "Starting up the Virtual Frame Buffer on Screen 1"
$XvfbPath/Xvfb :1 -fbdir /tmp >> /var/adm/Xvfb.log &
DISPLAY=‘hostname‘:1.0
export DISPLAY
fi

# Start the Brio OnDemand Server
if [ -x $HOME/server/ods.start ]; then
echo "Starting up Brio OnDemand Server"
# Execute ods.start as the Brio User, not as root
su $USER -c " cd $HOME/server;ods.start > /dev/null"
fi
exit 0
```

Sample Startup Script for AIX

Create a script similar to the following in the `/etc/` directory. After you create the script, add the following line to the end of the `/etc/inittab` file

```
brio:2:wait:/etc/rc.brio > /tmp/brio.out 2>&1

-----------------------------

Brio startup script for AIX

cat /etc/rc.brio

#!/bin/sh

# Brio Startup Script

DISPLAY=sunbird:0.0
export DISPLAY

# Start the Apache Web Server
echo "Starting the Apache Web Server"
/external/apache/bin/apachectl start >/dev/null 2>&1

USER=brio6x
HOME=/external/home/brio6x
BRIOHOME=/external/home/brio6x/610
IBPATH=/external/home/brio/lib

export USER HOME BRIOHOME

# Start Broadcast Server
if [ -x $BRIOHOME/bin/bqs.start ]; then
echo "Starting up Brio Broadcast Server"
# Execute bqs.start as the Brio User, not as root
```

```
su $USER -c "cd $BRIOHOME/bin > /dev/null
./bqs.start > /dev/null"
fi

# Start the Brio OnDemand Server
# DISPLAY is not necessary for V6 ODS
if [ -x $BRIOHOME/server/ods.start ]; then
echo "Starting up Brio OnDemand Server"
# Execute ods.start as the Brio User, not as root
su $USER -c "cd $BRIOHOME/server; ods.start > /dev/null"
fi

exit 0
```

ODS.ini File

It is generally not a good idea to modify this file directly—the Server Administrator application handles most changes—but on occasion you may want to view or modify the INI file. The ODS.ini file (for Windows NT systems) is stored in the `\ods_install_dir\server\` directory. The ODS.ini file (for Unix systems) is stored in the `/ods_install_dir/server/` directory.

While Brio ONE/ODS stores document information in the repository, it stores important information about application startup, configuration, log files, and load balancing in the ODS.ini file on the local disk where the server is installed.

ODS.ini Sample File

Every ODS.ini file will not include all of the lines listed here as many of them are optional or used only for particular ODS configurations.

For Windows NT Systems

Note Some lines are too long to fit in this manual without line breaks; however, they each constitute one line in the ODS.ini file. Within the file, there is no space between the equal sign and the following characters.

```
BQ_ODS_URL=/ods-isapi/ods.ods
BQ_PORTAL_SERVICE=(true/false)
BQ_PORTAL_HOSTNAME
BQ_PORTAL_PORT_NUM
BQ_DOCUMENT_DIRECTORY=
   C:\Program Files\Brio\Brio Enterprise Server
   \Program\Documents
BQ_OCE_DIRECTORY=
   C:\Program Files\Brio\Brio Enterprise Server
   \Program\Open Catalog Extensions
BQ_EXEC_PATH=
   C:\Program Files\Brio\Brio Enterprise Server
   \Program\brioqry.exe
BQ_LOG_DIRECTORY=
   C:\Program Files\Brio\Brio Enterprise Server
   \Server
BQ_TEMP_DIRECTORY=
   C:\Program Files\Brio\Brio Enterprise Server
   \Server\temp
```

```
BQ_LANGUAGE=en
BQ_MAXIMUM_PROCESS=10
BQ_MAXIMUM_PROCESSING_MEMORY=1000
BQ_KEEP_TEMP_FILES=(true/false)
BQ_START_LOG=(off, standard, trace, debug)
BQ_TOTAL_PRESPAWNED_PROCESS=1
BQ_CONFIGURATION_MODE=(Single, Cluster_Files_Synched)
BQ_TEMP_DIRECTORY_CLEANUP_PERIOD=(in minutes)
BQ_FILE_SYNCHRONIZATION_PERIOD=(in minutes)
BQ_ADMIN_TABLE_OWNER=brio
BQ_ADMIN_USERNAME=system
BQ_ADMIN_PASSWORD=encrypted string
BQ_ADMIN_OCE=brio.oce
BQ_USER_AUTH_MODE=0
BQ_USER_AUTH_CLASS_NAME
BQ_USER_AUTH_HOST_NAME
BQ_USER_AUTH_OPT_STR1
BQ_USER_AUTH_OPT_STR2
BQ_PUBLIC_FOLDER_READ_PERMISSION=(true/false)
BQ_PORT_NUM=5500
BQ_NOSYNC_ATTRIBUTES
ODS_LOCAL_AUTH_ACCEPTED=(true/false)
ODS_NODE_BY_HOSTNAME=(true/false)
ODS_USERCACHE_POLLING_RATE
ODS_USERCACHE_MAX_SIZE
BQ_VERSION_CHECK=true
BQ_VERSION_MISMATCH_MSG
BQ_MAXIMUM_LOG_FILES
BQ_AUDIT_LOG=(on/off)
```

For Unix Systems

Every ODS.ini file will not include all of the lines listed here as many of them are optional or used only for particular ODS configurations.

```
BQ_ODS_URL=/ods-isapi/ods.ods
BQ_PORTAL_SERVICE=(true/false)
BQ_PORTAL_HOSTNAME
BQ_PORTAL_PORT_NUM
BQ_DOCUMENT_DIRECTORY=/export/brio/bes/documents
BQ_OCE_DIRECTORY=/export/brio/bes/opencat
BQ_EXEC_PATH=/export/brio/bes/bin/bq
BQ_LOG_DIRECTORY=/export/brio/bes/server
BQ_TEMP_DIRECTORY=/export/brio/bes/server/tmp
BQ_LANGUAGE=en
BQ_MAXIMUM_PROCESS=10
BQ_KEEP_TEMP_FILES=(true/false)
BQ_START_LOG=(off, standard, trace, debug)
BQ_TOTAL_PRESPAWNED_PROCESS=1
BQ_CONFIGURATION_MODE=(Single, Cluster_Files_Synched)
BQ_TEMP_DIRECTORY_CLEANUP_PERIOD=(in minutes)
BQ_FILE_SYNCHRONIZATION_PERIOD=(in minutes)
BQ_ADMIN_TABLE_OWNER=brio
BQ_ADMIN_USERNAME=system
```

```
BQ_ADMIN_PASSWORD=encrypted string
BQ_ADMIN_OCE=brio.oce
BQ_USER_AUTH_MODE=0
BQ_USER_AUTH_CLASS_NAME
BQ_USER_AUTH_HOST_NAME
BQ_USER_AUTH_OPT_STR1
BQ_USER_AUTH_OPT_STR2
BQ_PUBLIC_FOLDER_READ_PERMISSION=(true/false)
BQ_PORT_NUM=5500
BQ_NOSYNC_ATTRIBUTES
ODS_LOCAL_AUTH_ACCEPTED=(true/false)
ODS_NODE_BY_HOSTNAME=(true/false)
ODS_USERCACHE_POLLING_RATE
ODS_USERCACHE_MAX_SIZE
BQ_VERSION_CHECK=true
BQ_VERSION_MISMATCH_MSG
BQ_MAXIMUM_LOG_FILES
BQ_AUDIT_LOG=(on/off)
```

Table 3-3 ODS.ini File Line Descriptions

Line	Explanation
BQ_ODS_URL	Specifies the broker URL that facilitates communication between the web server and ONE/ODS (BQ_ODS_URL=/ods-isapi/ods.ods or [BQ_ODS_URL=/ods-nsapi/ods.ods] or [BQ_ODS_URL=/ods-cgi/odscgi.exe] for NT or [BQ_ODS_URL=/ods-cgi/odscgi] for UNIX. When Web components are on different machines this line must be manually configured.
BQ_LANGUAGE=en	Specifies ONE/ODS language setting option. The default is English.
BQ_MAXIMUM_PROCESS	Controls the maximum number of BQ processes that each ODS node may spawn. Adjusting this value allows control of machine resources.
BQ_MAXIMUM_ PROCESSING_MEMORY	Specifies the maximum amount of RAM allocated to process data returned from a database query. Increasing this value can help performance at the expense of requiring more physical memory. When the maximum value is reached, the ODS starts using temporary disk files to manage memory requirements.
BQ_KEEP_TEMP_FILES= (true /false)	Instructs the Server to either delete or keep the temp files it uses when processing documents. For debugging purposes, you may want to keep the temp files on the server to verify they are being created correctly. The default is false.

Table 3-3 ODS.ini File Line Descriptions *(Continued)*

Line	Explanation
BQ_START_LOG=(off, standard, trace, debug)	Instructs the Server to generate a specific type of log file: The *off* setting turns the logging service off. The *standard* setting generates standard error messages only. The *trace* setting writes all events that occur to the log file. The *debug* setting turns on logging for ODS Java.
BQ_CONFIGURATION_MODE	See "Load Balancing Configuration Notes" on page 3-24
BQ_FILE_SYNCHRONIZATION_PERIOD =(in minutes)	Specifies the number of minutes between the node synchronization process. The default is 60 minutes. Changing this value is rarely necesssary as normal administration activity takes care of synchronization.
BQ_ADMIN_TABLE_OWNER	Specifies the database owner of the OnDemand Server repository tables. This setting is blank if the database your repository is installed on does not require or support owner names. Oracle databases require all upper case characters for owner names.
BQ_USER_AUTH_MODE=0	Corresponds to the Authentication mode the OnDemand Server uses. A value of 0=OnDemand Server authentication, 1=Database authentication, 2=Brio.Portal authentication, 3=Custom Method authentication. Do not change this setting manually; always use the GUI.
BQ_NOSYNC_ATTRIBUTES	Lists user specified ods.ini file lines of attributes that are not to be synchronized.
ODS_LOCAL_AUTH_ACCEPTED=(true/false)	Allows or does not allow local authentication to be used. The default (false) is more secure because it does not accept the locally stored password. If authentication fails, the user is prompted for a password. For Windows systems, user profile information is locally stored in the registry (see page 4-25.)
ODS_NODE_BY_HOSTNAME =(true/false)	Supports the use of either a host name or an IP address for routing. The default (false) instructs the ODS to use IP addresses.
ODS_USERCACHE_POLLING _RATE	Specify in minutes how often the cache should be checked against the maximum size set in ODS_USERCACHE_MAX_SIZE. If the maximum size is met or exceeded the user's cache is cleared.
ODS_USERCACHE_MAX_SIZE	Specify in number of megabytes the maximum size allowed for the user's cache. If the maximum size is met or exceeded the user's cache is cleared.

Table 3-3 ODS.ini File Line Descriptions *(Continued)*

Line	Explanation
BQ_VERSION_CHECK= true/false	Checks the server version number with the client version number. Default is true, check version number. Returns the following error message: "ODS server version mismatch with client; y.y.y.y ODS expects client with version x.x.x.x and above. Please check your installation."
BQ_VERSION_ MISMATCH_MSG=error_ message	Appends an error message to the standard error message allowing users to provide contact information when the version numbers between the server and client do not match.
BQ_MAXIMUM_LOG_FILES	Controls the maximum number of back-up log files saved for the ODSNode.log and the ODSManager.log.
BQ_AUDIT_LOG= (on/off)	Instructs the Server to create an audit file or not. The default is set to on.

Clearing the Node's Cache

The ODSNode process caches the list of documents accessible to each user upon login. This improves performance on subsequent requests by the same user, however, the ODSNode process grows in size when many users are logging into the ODS. To improve performance, the cache can be cleared using the following ODS.ini parameters:

```
ODS_USERCACHE_POLLING_RATE (minutes)
ODS_USERCACHE_MAX_SIZE (megabytes)
```

These parameters are not in the ODS.ini file by default.

To clear the cache:

1. Set the following two parameters in the ODS.ini file:
   - `ODS_USERCACHE_POLLING_RATE (minutes) – This parameter sets the interval the` *`heap size`* `will be checked.`
   - `ODS_USERCACHE_MAX_SIZE (megabytes) –` The cache will be cleared when the used memory (but not the total memory) is equal to or greater than 80% of the user specified max cache size.
2. Restart the OnDemand Server.

Example

With the following settings, the OnDemand Server will create a process that checks the node's heap size every 10 minutes and clears the cache the used memory is equal to or greater than 52.2MB (64MB x .80).

```
ODS_USERCACHE_POLLING_RATE = 10
ODS_USERCACHE_MAX_SIZE = 64
```

Load Balancing Configuration Notes

In a load balancing configuration, all database configuration information needs to be replicated across all machines. Configuration information such as ODBC data source names, Oracle host name information, and Open Client host information needs to be replicated (*replication*) identically across all machines, since all machines use the same OCE files to connect, and this information is stored inside the OCE file.

The `INI` file entry `BQ_CONFIGURATION_MODE` controls load balancing configuration for each machine. There are two options:

- Single – Installation without load balancing
- Cluster_Files_Synched – Installation utilizing load balancing

  with the `Cluster_Files_Synched` option, all documents and OCEs are synchronized, from the manager to the nodes, at various times:

  - Upon node startup
  - At an interval (in minutes) defined by the `BQ_FILE_SYNCHRONIZATION_PERIOD` field in the ODS.ini file
  - Whenever a new document or OCE is registered to the manager through the Server Administrator or Brio Intelligence

  In the `Cluster_Files_Synched`þ load balancing configuration, parts of the ODS.ini file need to be synchronized across the multiple machines. Entries in the ODS.ini file fall into two categories:

  - Entries read from the local machine and not synchronized from manager to all nodes
  - Entries determined by the manager machine (and, therefore, synchronized)

Local Node Machine Entries

You cannot use a GUI (Graphical User Interface) to modify ODS.ini file entries local to the node machine; the Server Administrator communicates only with the manager and the ODS.ini file on the manager's system.

The following values are set with defaults or install parameters during installation. If they need to be modified, you have to do it by hand with a text editor. By default, these settings are local and will not be synchronized:

```
BQ_DOCUMENT_DIRECTORY
BQ_OCE_DIRECTORY
BQ_EXEC_PATH
BQ_LOG_DIRECTORY
BQ_TEMP_DIRECTORY
BQ_LANGUAGE
```

In addition, you can "localize" the following entries by including them in the `BQ_NOSYNC_ATTRIBUTES` setting, in which case, they will be set manually on the node machine and not synchronized with the manager. This allows for machine-specific settings in areas such as `BQ_MAXIMUM_PROCESS`. The following settings will be synchronized by default unless you specify otherwise in the `BQ_NOSYNC_ATTRIBUTES` setting:

```
BQ_MAXIMUM_PROCESS
BQ_KEEP_TEMP_FILES (true/false)
BQ_MAXIMUM_PROCESSING_MEMORY
BQ_START_LOG (off, standard, trace, debug)
BQ_TEMP_DIRECTORY_CLEANUP_PERIOD (in minutes)
BQ_TOTAL_PRESPAWNED_PROCESS
BQ_AUDIT_LOG (on/off)
```

Manager Machine Entries

Manager machine entries are synchronized when one of the following events occur:

- Changes are made by the Server Administrator
- Interval defined (in minutes) by the `BQ_FILE_SYNCHRONIZATION_PERIOD` ODS.ini file setting
- Node startup

```
BQ_CONFIGURATION_MODE (Single, Cluster_Files_Synched)
BQ_FILE_SYNCHRONIZATION_PERIOD (in minutes)
BQ_ADMIN_TABLE_OWNER
BQ_ADMIN_USERNAME
BQ_ADMIN_PASSWORD
BQ_ADMIN_OCE
BQ_USER_AUTH_MODE
```

```
BQ_USER_AUTH_CLASS_NAME
BQ_USER_AUTH_HOST_NAME
BQ_USER_AUTH_OPT_STR1
BQ_USER_AUTH_OPT_STR2
BQ_PUBLIC_FOLDER_READ_PERMISSION
BQ_PORT_NUM
BQ_NOSYNC_ATTRIBUTES
```

ODSCGI.ini File for UNIX Systems

ODSCGI.ini is used by the Web server to direct requests to the correct OnDemand Server port. The file is written (or updated, in an upgrade) by the installer when it runs the following script:

```
BRIO_HOME/server/ods_web_config
```

The install writes the ODSCGI.ini file in the /etc directory, or adds to it if this is not the first OnDemand Server on the system.

The following is a sample *ODSCGI.ini* file for a machine with multiple OnDemand Servers. The ":" and the "=" signs are the delimiters.

```
# cgi_directory:web_server_port=ods_host:ods_port
/ods-cgi:4003=duster:5500
/ods-cgi:4004=duster:5500
/ods-nsapi:90=duster:3300
```

The following is an explanation of the above sample file:

- The first column is the Web component type (CGI or NSAPI).
- The second column is Web server port. It must be unique on the machine. The Web server host name is not specified because it is implied by the machine that has the file. This port is used by the OnDemand Server for processing queries, logon, and downloading documents.
- The third column is ODS host name.
- The fourth column is the ODS port. The ODS port does not need to be unique on the machine, since you can configure two Web servers to access the same ODS. The second port number (5500 and 3300, in the example) is used by users to connect to the OnDemand Server from the Web.

In the `ODSCGI.ini` you can have multiple mappings from a virtual path to the OnDemand server, for example:

```
/ods-cgi:80=duster:5500
/ods-nsapi:80=duster:5500
```

Note The installer overwrites duplicate entries (where the Web server port is a duplicate).

The following line specifies in what language the error messages appear. (In the example, *_en* stands for *English*.)

```
ODSI18N=/brioone/ods/server/messages/odsi18n_en.properties
```

Always overwrite this when you do a second install on one machine. There should only be only one line of this type if all your ODS installations are in one language and are the same version.

Log Files

There are several options available for logging. You can enable logging with standard error messages only, or with a full trace of all events that occur.

ODS Java

For ODS Java, you can specify logging service by individual component, as described briefly in the following paragraphs:

Process Factory

To turn logging on for the process factory, use the following flag on a command line:

```
<process factory exec path> -d[0,1,2] <arguments>
```

where 0 = off, 1 = standard error messages, and 2 = full tracing.

ODS Logging

To start either an `ODSManager.log` or an `ODSNode.log`, change the line `BQ_STARTLOG` in the ODS.ini file.

```
BQ_START_LOG=[off, standard, trace, debug]
```

See Table 3-3 on page 3-21 for a description of the above ODS.ini line.

Note You must restart the OnDemand Server for these changes to take effect.

CGI for Windows NT Systems

Add the following entries to the registry, under *Broker*:

```
Log_Mode:[off, standard, trace]
Log_File_Name:<log file name>
```

The log file name will usually be something like þ`d:\temp\ODSCGI.log`

CGI for UNIX Systems

Add the following entries to the `/etc/ODSCGI.ini` file:

```
Log_Mode:[off, standard, trace]
Log_File_Name:<log file name>
```

The log file name will usually be something like `/tmp/ODSCGI.log`

Note You must restart the OnDemand Server for these changes to take effect.

Back-up Log Files

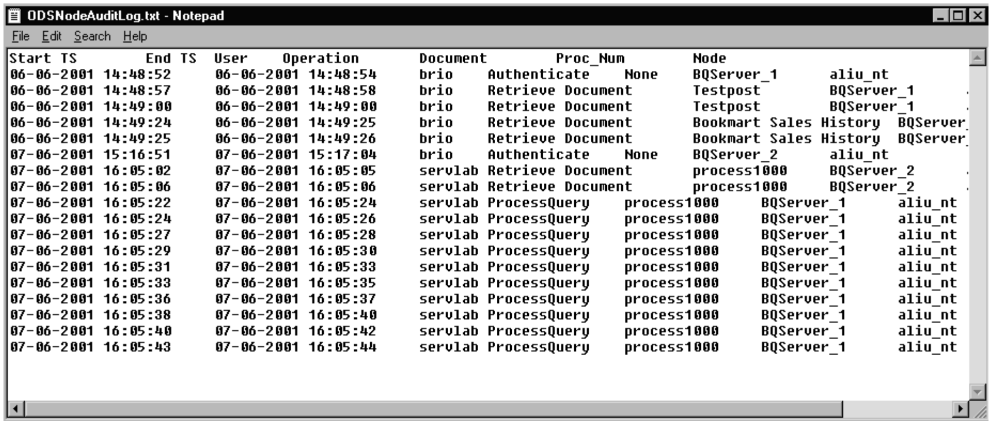
The OnDemand Server maintains the last 5 log files by default for the ODSNode.log, the ODSManager.log, and the ODStartup.log. Use the `BQ_MAXIMUM_LOG_FILES` parameter in the ODS.ini file to set the desired number (for the ODSNode.log or the ODSManager.log) of back-up log files. The back-up log file naming convention is: logFile.log (most recent), logFile_1.log, logFile_2.log, so on. When the maximum number of log files is exceeded the oldest log file is overwritten.

Audit Log File

The audit log file (called ODSNodeAuditLog.txt) is a new feature of OnDemand Server 6.5. This file captures three OnDemand Server operations (authentication or login, retrieving a document, or processing a query) in order to help you understand the usage and performance of your OnDemand Server. The fields in the log file are delimited by a tab. The audit log file (Figure 3-1) contains the following information for each operation:

- Start time - DD-MM-YYY HH:MM:SS
- End time - DD-MM-YYY HH:MM:SS
- Login ID of the current user - BRIOUSER cookie value

- Operation type - authenticate, retrieve document, or process query
- Document - document unique name
- Proc_Num - description of the local BQ daemon
- Node - hostname or IP address of the Brio OnDemand Server



```
Start TS        End TS  User    Operation       Document        Proc_Num        Node
06-06-2001 14:48:52     06-06-2001 14:48:54     brio    Authenticate    None    BQServer_1      aliu_nt
06-06-2001 14:48:57     06-06-2001 14:48:58     brio    Retrieve Document       Testpost        BQServer_1
06-06-2001 14:49:00     06-06-2001 14:49:00     brio    Retrieve Document       Testpost        BQServer_1
06-06-2001 14:49:24     06-06-2001 14:49:25     brio    Retrieve Document       Bookmart Sales History  BQServer_
06-06-2001 14:49:25     06-06-2001 14:49:26     brio    Retrieve Document       Bookmart Sales History  BQServer_
07-06-2001 15:16:51     07-06-2001 15:17:04     brio    Authenticate    None    BQServer_2      aliu_nt
07-06-2001 16:05:02     07-06-2001 16:05:05     servlab Retrieve Document       process1000     BQServer_2
07-06-2001 16:05:06     07-06-2001 16:05:06     servlab Retrieve Document       process1000     BQServer_2
07-06-2001 16:05:22     07-06-2001 16:05:24     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:24     07-06-2001 16:05:26     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:27     07-06-2001 16:05:28     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:29     07-06-2001 16:05:30     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:31     07-06-2001 16:05:33     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:33     07-06-2001 16:05:35     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:36     07-06-2001 16:05:37     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:38     07-06-2001 16:05:40     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:40     07-06-2001 16:05:42     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
07-06-2001 16:05:43     07-06-2001 16:05:44     servlab ProcessQuery    process1000     BQServer_1      aliu_nt
```

Figure 3-1 Sample ODSNodeAuditLog.txt File

Note In a BrioOne deployment the Login ID of the current user is the usrsrv value.

The `BQ_AUDIT_LOG` setting synchronizes in the ODS.ini from the ODS manager to the ODS nodes by default. You can override this default by listing `BQ_AUDIT_LOG` as an entry in the `BQ_NOSYNCH_ATTRIBUTES` list. `See` "Local Node Machine Entries" on page 3-25 for more information on the `BQ_NOSYNCH_ATTRIBUTES entry in the ODS.ini file.`

Because the log files are for individual nodes you need to combine the separate audit logs or load the data into a database before importing the data into Brio Intelligence for analysis.

The audit log file is created in the log directory specified in the `BQ_LOG_DIRECTORY` entry in the ODS.ini file.

Process Multiplier

The process multiplier provides the ability to simulate load testing. The OnDemand Server performs each request it receives as many times as the ODS_PROCESS_MULTIPLIER variable is set for. This action allows a single client machine to trigger a large number of processes. The success or failure of each process is logged in the ODSCGI.log file.

To set up the process multiplier:

For Windows NT, set the value (between 0 and 10,000) of the process multiplier in the registry: [HKEY_LOCAL_MACHINE\SOFTWARE\\Brio Enterprise Server\6.0.000\Broker]

ODS Process Multiplier=10

For UNIX, set the value (between 0 and 10,000) of the process multiplier in the ODSCGI.ini file:

```
ODS_PROCESS_MULTIPLIER=10
```

Multiple GET Requests from Internet Explorer

Internet Explorer makes multiple GET requests when a BQY document is loaded into the plug-in. To improve the performance of the OnDemand Server the Web broker code was changed to ignore these multiple requests. Ignoring these requests can cause problems in some versions of Internet Explorer. If you have problems loading documents from your plug-in set the following parameter to true (the default is false):

For Windows in the registry:

[HKEY_LOCAL_MACHINE\SOFTWARE\Brio Enterprise Server\6.0.000\Broker] Enable_Multiple_Reques
ts=true

For UNIX in the ODSCGI.ini:

Enable_Multiple_Requests=true

Note In BrioONE mode, when accessing documents registered prior to 6.5, you may see multiple GET requests from Internet Explorer. To increase the performance, you can either re-register the old documents, or update them (for example, by changing the users/groups that access the document) from the ODS Administrator.

Summary

This chapter summarizes the configuration and administration tasks for the OnDemand Server, and describes in detail the administration tasks using the Brio Intelligence Server Administrator or the ODS.ini file.

4 OnDemand Server Web Site

Users connect to OnDemand Server through an HTML directory, similar to a Web site, which authenticates their access privileges and displays the contents of the OnDemand Server registry (see “Security” on page 1-17).

The HTML directory also contains optional Zero Administration update technology for automatically installing and upgrading Brio client software.

This chapter contains the following sections:

- OnDemand Server Web Site
- Zero Administration

OnDemand Server Web Site

The OnDemand Server Web site consists of a series of HTML pages distributed across the OnDemand Server and the Web server. You can add content or customize the appearance of any of these pages as long as the basic structure of the pages is not changed.

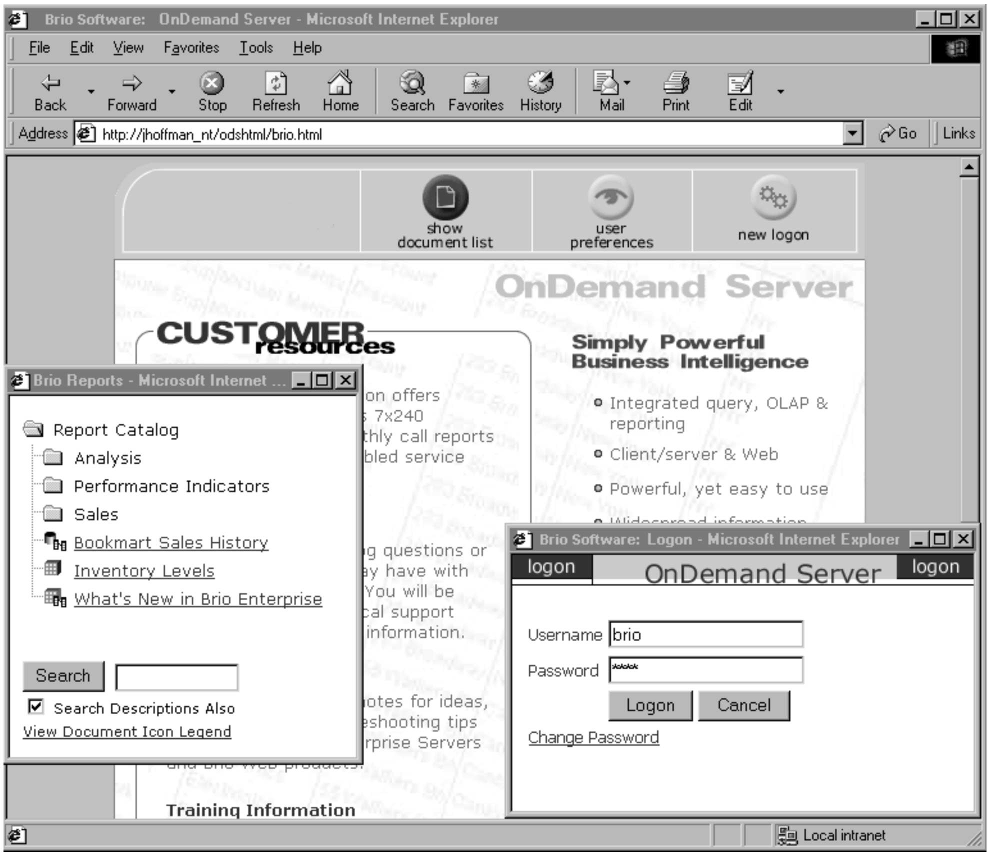
☆Tip It is recommended that you use a plain text editor for Web page modification. GUI-based Web authoring tools may have adverse affects on the JavaScript code. You should only change these files if you are experienced at HTML coding.

Customizing the OnDemand Server

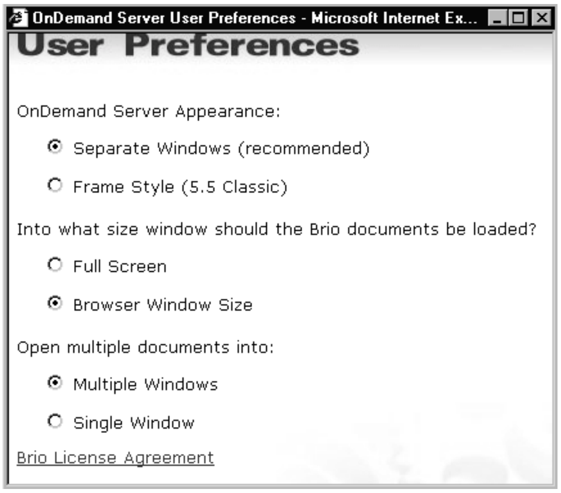
The Brio OnDemand Server uses HTML files with JavaScript to control the login process and to make authorized documents available to the user. The appearance and behavior of the OnDemand Server can be changed by modifying the default or the sample HTML files, or by developing your own HTML files.

OnDemand Server Default User Interface

The default interface discussed in this section is that of the stand-alone version of Brio Intelligence. Three HTML pages comprise the user interface of the OnDemand Server. The pages are the start page (`brio.html`), the logon page (`changepwd1.html`), and the document list page (`reportlist.html`).


Brio Software: OnDemand Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address http://jhoffman_nt/odshtml/brio.html
Go Links
show document list
user preferences
new logon
OnDemand Server
CUSTOMER resources
Simply Powerful Business Intelligence
Integrated query, OLAP & reporting
Client/server & Web
Powerful, yet easy to use
Training Information
Brio Reports - Microsoft Internet ...
Report Catalog
Analysis
Performance Indicators
Sales
Bookmart Sales History
Inventory Levels
What's New in Brio Enterprise
Search
Search Descriptions Also
View Document Icon Legend
Brio Software: Logon - Microsoft Internet Explorer
logon
OnDemand Server
logon
Username brio
Password ****
Logon Cancel
Change Password
Local intranet

A fourth page, which is optional, is the user preferences page (`user.prefs.html`).



OnDemand Server User Preferences - Microsoft Internet Ex...

User Preferences

OnDemand Server Appearance:

[✓ Separate Windows (recommended)]

Frame Style (5.5 Classic)

Into what size window should the Brio documents be loaded?

Full Screen

[✓ Browser Window Size]

Open multiple documents into:

[✓ Multiple Windows]

Single Window

Brio License Agreement

Customizing the HTML Pages

There are several ways to customize the HTML pages to meet your company's needs.Very simple changes can be made to the default start page (see the following section for instructions). For more extensive modifications use one of the three sample HTML code files as a template (see page 4-5 for instructions). Alternately, you can develop your HTML pages from scratch (see page 4-12 for guidelines).

OnDemand Server Default Start Page

The Show Document List and Logon buttons, used to access Brio Intelligence documents through the OnDemand Server, are the only required elements on the start page. While you cannot remove these buttons, you can customize their appearance. All other elements can be removed or modified. For example, you can insert your company logo, replace the Brio content with your company content, and select the correct Web broker type.

To customize the default start page (`\\HTML\brio.html`):

1. Copy the `\\HTML\brio.html` into `\\HTML\newstart.html.`

   Making your changes in a different file safeguards your changes when you update the OnDemand Server.

2. To change the logo and its link, edit this line of code in the `\\HTML\newstart.html` file.

   ```
   <td width="586"><nobr><a href="http://www.brio.com"><img
        SRC="/odsimage/start.page/logo_left.gif" WIDTH="189"
   HEIGHT="75" BORDER="0" alt="www.brio.com"></a><img
   ```

3. To replace the Brio content with your content, modify or delete the code between these lines:

   ```
   <tr> <!--left side gray border of document space. Creates 3D
   effect -->
   .
   .
   .
   <!--Left hand gray border. Gives data area a 3D effect -->
   ```

   See page 27 for a complete copy of the `\\default\brio.html\` file.

4. To select the Web broker type, uncomment the appropiate code line.

   ```
   // global variables
   // uncomment one of the following, depending on what Web Broker
   module you have installed
   BrioStartURL = "../ods-isapi/ods.ods" // isapi
   // BrioStartURL = "../ods-nsapi/ods.ods"// nsapi
   // BrioStartURL = "../ods-cgi/odscgi.exe" // cgi on windows
   // BrioStartURL = "../ods-cgi/odscgi"// cgi on unix
   ```

5. Save `\\html\newstart.html`.

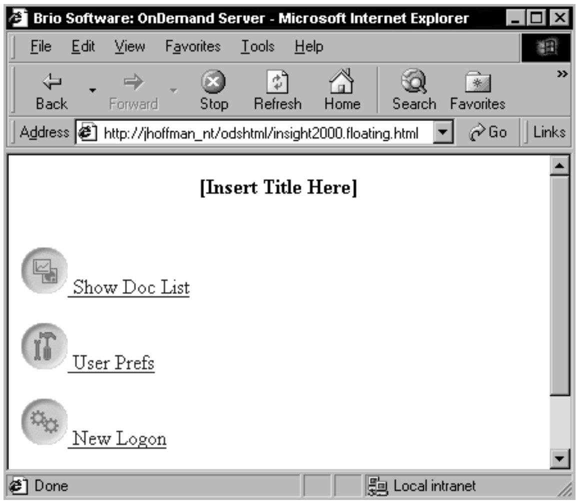
6. To run the OnDemand Server, replace `brio.html` in the URL with your new file, `newstart.html`.

   ```
   http://<ondemand_server_hostname>:port_number/odshtml/newstart.
   html
   ```

Sample HTML Pages

Brio provides three different sample HTML code files for you to use as templates for your customizing effort. Each sample code file, floating window, framed window, and simple list, provides a different look for the OnDemand Server while including all the elements, the required buttons, and the necessary links between the files to use the OnDemand Server.

The characteristics of the sample code files are listed in Table 4-1.


Brio Software: OnDemand Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://jhoffman_nt/odshtml/insight2000.floating.html
Go Links
Done
Local intranet

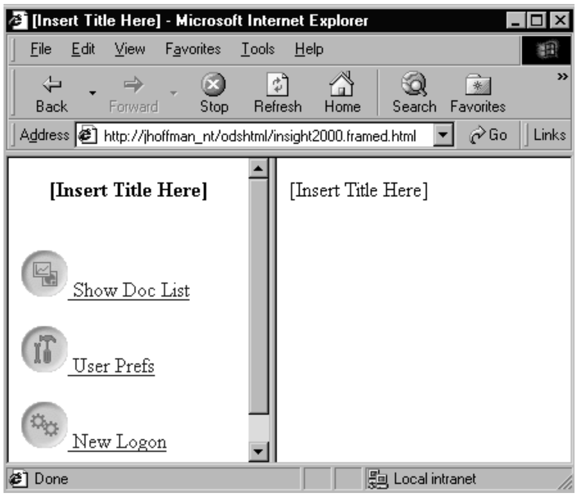
[Insert Title Here]

Show Doc List

User Prefs

New Logon

Figure 4-1 Floating Window Sample Start Page


[Insert Title Here] - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://jhoffman_nt/odshtml/insight2000.framed.html
Go Links
Done
Local intranet

[Insert Title Here]

Show Doc List

User Prefs

New Logon

[Insert Title Here]

Figure 4-2 Framed Window Sample Start Page

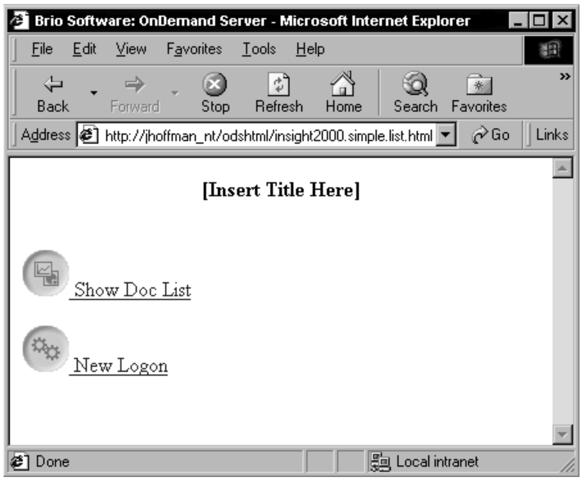

Brio Software: OnDemand Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address http://jhoffman_nt/odshtml/insight2000.simple.list.html Go Links
[Insert Title Here]
Show Doc List
New Logon
Done
Local intranet

Figure 4-3 Simple List Sample Start Page

Table 4-1 Characteristics of the Sample HTML Files

Sample File Name	Characteristics
Insight2000floating	Document list uses nested folders Provides button for user preferences May use a single window or multiple windows to display documents
Insight2000framed	Document list uses nested folders Provides button for user preferences May use a single window or multiple windows to display documents
Insight2000simple.list	Document list has no nested folders Document replaces the document list in the open window Does not provide button for user preferences

Sample HTML Pages: Example

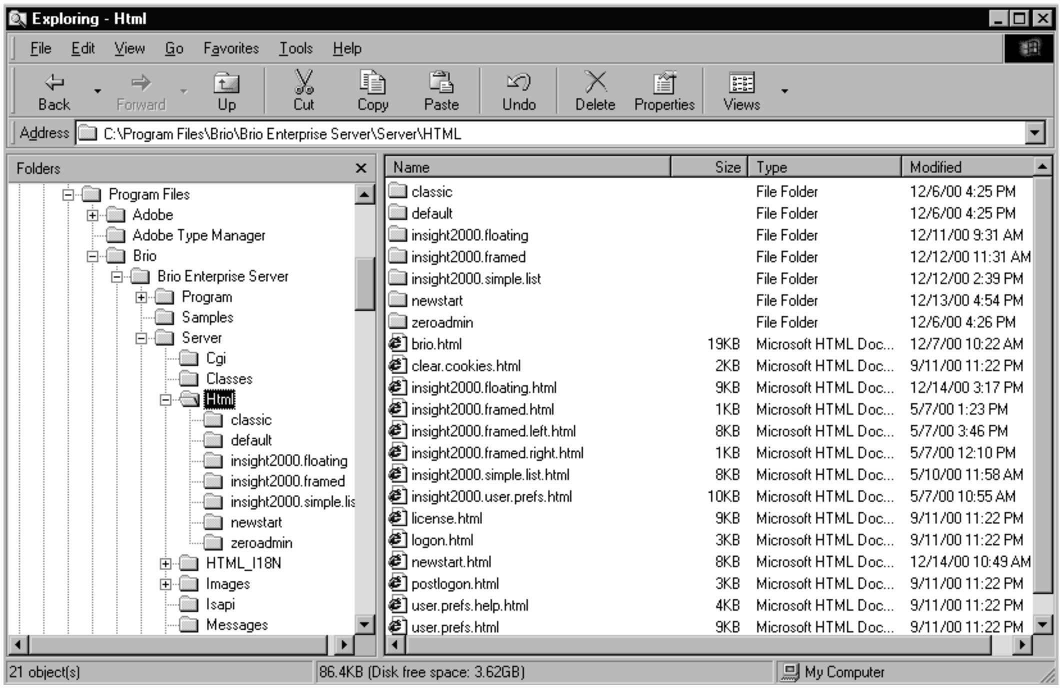
To customize the look and feel of the OnDemand server using the `insight2000.floating` pages:

1 Download the sample code file, `Insight 2000 - Customizing the ODS - sample files.zip`, located at `ftp://ftp.brio.com/pub/insight2000`.

2 Back up your existing files in the `\\Brio Enterprise Server\server\html` directory.

3 Extract the files in `Insight2000.zip`.

Choose `C:\` and the files will automatically unzip in the `\\Brio Enterprise Server\server\html directory.`



Address: C:\Program Files\Brio\Brio Enterprise Server\Server\HTML

```
Program Files
  Adobe
  Adobe Type Manager
  Brio
    Brio Enterprise Server
      Program
      Samples
      Server
        Cgi
        Classes
        Html
          classic
          default
          insight2000.floating
          insight2000.framed
          insight2000.simple.lis
          newstart
          zeroadmin
        HTML_I18N
        Images
        Isapi
        Messages
```

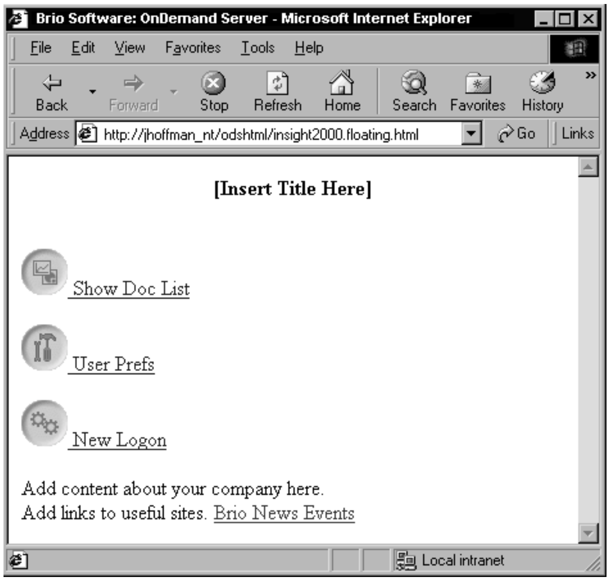
Name	Size	Type	Modified
classic		File Folder	12/6/00 4:25 PM
default		File Folder	12/6/00 4:25 PM
insight2000.floating		File Folder	12/11/00 9:31 AM
insight2000.framed		File Folder	12/12/00 11:31 AM
insight2000.simple.list		File Folder	12/12/00 2:39 PM
newstart		File Folder	12/13/00 4:54 PM
zeroadmin		File Folder	12/6/00 4:26 PM
brio.html	19KB	Microsoft HTML Doc...	12/7/00 10:22 AM
clear.cookies.html	2KB	Microsoft HTML Doc...	9/11/00 11:22 PM
insight2000.floating.html	9KB	Microsoft HTML Doc...	12/14/00 3:17 PM
insight2000.framed.html	1KB	Microsoft HTML Doc...	5/7/00 1:23 PM
insight2000.framed.left.html	8KB	Microsoft HTML Doc...	5/7/00 3:46 PM
insight2000.framed.right.html	1KB	Microsoft HTML Doc...	5/7/00 12:10 PM
insight2000.simple.list.html	8KB	Microsoft HTML Doc...	5/10/00 11:58 AM
insight2000.user.prefs.html	10KB	Microsoft HTML Doc...	5/7/00 10:55 AM
license.html	9KB	Microsoft HTML Doc...	9/11/00 11:22 PM
logon.html	3KB	Microsoft HTML Doc...	9/11/00 11:22 PM
newstart.html	8KB	Microsoft HTML Doc...	12/14/00 10:49 AM
postlogon.html	3KB	Microsoft HTML Doc...	9/11/00 11:22 PM
user.prefs.help.html	4KB	Microsoft HTML Doc...	9/11/00 11:22 PM
user.prefs.html	9KB	Microsoft HTML Doc...	9/11/00 11:22 PM

21 object(s) | 86.4KB (Disk free space: 3.62GB) | My Computer

4 Edit the `insight2000.floating.html` file by adding content about your company between the double `<br>` and the end body tag, `</body>`.

```
<body>
<center><strong>
Welcome to Insight 2000
</center></strong>
<br>
<br>

<a href="javascript:showDocList()"
   onMouseOver="return setStatus('Show Document List')"
   onMouseOut="return setStatus('')">
   <img SRC="/odsimage/extras/icon_reports.gif" BORDER="0" alt="Show
Document List"> Show Doc List</a>
<br><br>
<a href="javascript:showUserPrefs()"
   onMouseOver="return setStatus('Set OnDemand Server User Preferences')"
   onMouseOut="return setStatus('')">
   <img SRC="/odsimage/extras/icon_tools.gif" BORDER="0" alt="User
Preferences"> User Prefs</a>
<br><br>
<a href="javascript:showLogon()"
   onMouseOver="return setStatus('OnDemand Server Logon')"
   onMouseOut="return setStatus('')">
   <img SRC="/odsimage/extras/icon_gears.gif" BORDER="0" alt="New
Logon"> New Logon</a>
<br><br>
Add content about your company here
Add links to useful sites. <a href="http://www.brio.com/news_events/index.html"
_blank>Brio News Events</a>
</body>
```


Brio Software: OnDemand Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History
Address http://jhoffman_nt/odshtml/insight2000.floating.html
Go Links
Local intranet

[Insert Title Here]

Show Doc List

User Prefs

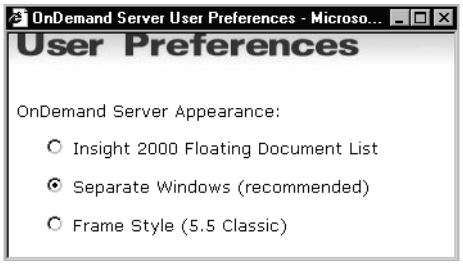
New Logon

Add content about your company here.
Add links to useful sites. Brio News Events

5 Edit the `\\html\insight2000.user.prefs.html` file to delete the Insight 2000 Framed Document List option on the Users Preferences page.

Comment out the following code.

```
<!--/////// Add the insight2000.framed Style choice-->
    <tr valign="top">
      <td width="4%"><img
src="/odsimage/start.page/spacehold.gif" width="1"
height="1"></td>
      <td width="96%" height="10">
        <input type=radio name=visualstyle
value="insight2000.framed">
        Insight 2000 Framed Document List</td>
   </tr>
<!--///////End of Add the insight2000.framed Style choice-->
```



User Preferences

OnDemand Server Appearance:

- ○ Insight 2000 Floating Document List
- [✓ ● Separate Windows (recommended)]
- ○ Frame Style (5.5 Classic)

6 Check that the links of the files in the Insight2000 floating directory listed in Table 4-2 have been changed to use the `Insight2000.floating.html` file and the Insight2000 floating directory.

If the links are not changed, the `\\html\default\` files will be referenced instead of the `\\html\insight2000floating\` files.

Table 4-2 Changed Pointers

Link	Old value	New Value	HTML File (Insight2000floating Directory)
`HTMLfile=`	`“default\...`	`“insight2000. floating\...`	Logon - 2 places Logon.changepwd - 2 places Logon.changepwd2.html - 1 place Logon.normal - 1 place
`window. open`	`(“\odshtml /default\...`	`(“\odshtml \insight2000. floating`	Postlogon - 3 places Reportlist - 4 places Reportlist.footer - 1 change
`src=`	`“\odshtml \default\...`	`“\odshtml \insight2000. floating`	Reportlist - 3 places

7 To run the OnDemand Server using Insight2000floating pages, replace `brio.html in the URL` with your new start page, `insight2000floating.html`.

```
http://<ondemand_server_hostname>:port_number/odshtml/in
sight2000floating.html
```

Specific Modifications

To force all users to use the same look:

On the start page (for example, the `brio.html` or `insight2000floating.html`, comment out all three instances of the line, `var preflook = getCookie(“brioodspreflook=”),` and add the line, `preflook = “insight2000.floating”.`

To disable zero administration:

In `\\Default\reportlist.html\\` comment out the call to `zeroadminMain().`

Developing HTML Pages from Scratch

Most of the Brio HTML pages for the OnDemand Server are standard HTML and Javascript code. Creating your own HTML pages requires ability to work with HTML coding, Javascript coding, the file structure of the Brio HTML pages, and the special ODS tags.

This section provides information on the minimum code requirements to run the OnDemand Server, how the Brio HTML pages work, and the purpose of the ODS tags.

Minimum Code Requirements

The following two files, logon.html and postlogon.html contain the minimum code to perform basic OnDemand Server functions. To bypass the Brio installed HTML files place these files in the HTML folder and access the OnDemand Server using the following URL:

```
http://<ondemand_server_hostname>:port/ods-isapi/ods.ods
```

☆ **Tip** The Brio default pages and sample HTML code pages have JavaScript in them to validate the input fields.

Logon.html

```
<HTML>
<body>
<FORM METHOD=POST ACTION=<ODSLogonMethodTag>><br>
Username: <INPUT NAME=“Username” TYPE=text MAXLENGTH=30 SIZE=15>
<BR>
Password: <INPUT NAME=“Password” TYPE=password MAXLENGTH=30
SIZE=15>
<BR>
<INPUT NAME=“OK” TYPE=submit VALUE=“ LOG IN “>
</FORM>
</body>
</HTML>
```

Postlogon.html

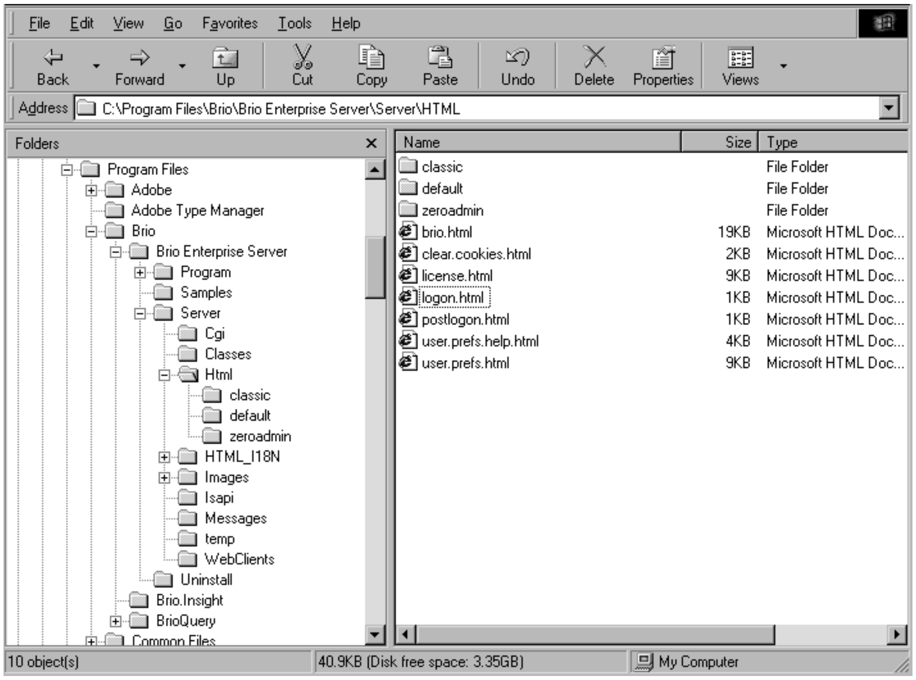
```
<HTML>
<ODSDocListTag>
</HTML>
```

The `insight2000.simple.list.html` file and `insight2000.simple.list` directory is another place to start. These pages contain all of the Brio written functions except the folder tree document list and the option page for user preferences.

HTML Pages Installed with OnDemand Server

The OnDemand Server Web site consists of a series of HTML pages distributed across the OnDemand Server and the Web server. The HTML pages are divided into two categories, pages containing ODS tags (referred to in older Brio documentation as dynamic HTML pages) and pages containing no ODS tags (referred to in older Brio documentation as static HTML pages).

When the OnDemand Server and the Web server are installed on one machine, all the HTML files reside in the HTML directory on the ODS node. The OnDemand Server sets up virtual directories (also known as document directories or directory mappings) on the Web server . The virtual directories, `odshtml` and `odsimage`, point to a set of HTML files without ODS tags installed in `Program Files\Brio\Brio Enterprise Server\server\html`.



Address: C:\Program Files\Brio\Brio Enterprise Server\Server\HTML

```
Program Files
  Adobe
  Adobe Type Manager
  Brio
    Brio Enterprise Server
      Program
      Samples
      Server
        Cgi
        Classes
        Html
          classic
          default
          zeroadmin
        HTML_I18N
        Images
        Isapi
        Messages
        temp
        WebClients
      Uninstall
    Brio.Insight
    BrioQuery
  Common Files
```

Name	Size	Type
classic		File Folder
default		File Folder
zeroadmin		File Folder
brio.html	19KB	Microsoft HTML Doc...
clear.cookies.html	2KB	Microsoft HTML Doc...
license.html	9KB	Microsoft HTML Doc...
logon.html	1KB	Microsoft HTML Doc...
postlogon.html	1KB	Microsoft HTML Doc...
user.prefs.help.html	4KB	Microsoft HTML Doc...
user.prefs.html	9KB	Microsoft HTML Doc...

10 object(s) | 40.9KB (Disk free space: 3.35GB) | My Computer

When the Web server and the OnDemand Server are installed on separate machines the HTML files with ODS tags reside on the ODS node, while the HTML files without ODS tags reside on the Web server. The HTML files without ODS tags are also mapped to the virtual directories, `odshtml` and `odsimage`, on the Web server.

Note If using an Apache Web server see the BrioFAQ, *How do I configure an Apache Web server on Windows NT or UNIX? (Ref. #000410-0000)*, located at `http://www.brio.com/support_services/technical_support/index.html`.

HTML Files with ODS Tags

The OnDemand Server reads the files, interprets the special tags, and replaces them with user-specific information. For example, the server dynamically replaces the special ODS tag, <`ODSDOCListTag`>, with the list of files a particular user is allowed to access. The OnDemand Server performs this replacement as it returns the HTML page to the user's browser. See "Understanding the OnDemand Server Tags" on page 17 for descriptions of the ODS server tags.

Table 4-3 lists the HTML files from the `default` directory that contain ODS tags.

Table 4-3 HTML Files with ODS Tags

File Name (.html)	Location	ODS Tag in the File	Purpose
`logon.changepwd1`	`\\Server\HTML \default`	`<ODSLogonMethodTag>`	Presents dialog to change a password
`logon.changepwd2`	`\\Server\HTML \default`	`<ODSLogonMethodTag>`	Validates the password
`logon.getdoc`	`\\Server\HTML \default`	`<ODSLogonMethodTag>`	Presents the document list upon successful logon
`logon`	`\\Server\HTML \default`	`<ODSSecurityTag>`	Invokes either the logon.normal page (when using OnDemand Server authentication) or the logon.changpwd1 page (when using either database or custom user authentication)
`logon.normal`	`\\Server\HTML \default`	`<ODSLogonMethodTag>`	Presents the logon page without the change password option
`postlogon`	`\\Server\HTML \default`	`<ODSDocListTag format =array>`	Requests the document list array from the OnDemand Server.
`reportlist.ie3`	`\\Server\HTML \default`	`<ODSDocListTag format =array>`	Presents the document list for the Microsoft Internet Explorer version 3. This is a work-around because IE 3 will not accept information from the ODS tag in the postlogon.html file.

HTML Files Without ODS Tags

HTML files that do not contain any ODS tags do not need to be processed or modified in any way. The images and instructional text that users see when they log in reside on the Web server as HTML and GIF files. These files display the opening screen, the logon screen, and the user preferences screen.

`Reportlist.html` includes the JavaScript code for the Zero Administration clients. This code determines 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 page 4-12 for instructions on disabling Zero Administration.) Table 4-4 lists the HTML files that do not contain ODS tags. (This list does not include the files in the classic directory.)

Table 4-4 HTML Files Without ODS Tags

File Name (.html)	Location	Purpose
`folder-tree.js`	`\\server\HTML\default`	Displays the document list folder tree
`brio`	`\\server\HTML`	Diplays the start page
`clear.cook-ies`	`\\server\HTML`	Clears cookies on the server
`license`	`\\server\HTML`	Displays the Brio license agreement
`user.prefs`	`\\server\HTML`	Displays the user preferences
`user.prefs.help`	`\\server\HTML`	Displays help for the user preferences
`reportlist.footer`	`\\Server\HTML\default`	Displays the search on the bottom of the document list
`reportlist`	`\\Server\HTML\default`	Displays the document list
`reportlist.legend`	`\\Server\HTML\default`	Displays the report list legend
`reportlist.tree`	`\\Server\HTML\default`	Displays the word "working" when the ODS is busy
`working`	`\\Server\HTML\default`	Displays the word "working" when the ODS is busy

Understanding the OnDemand Server Tags

OnDemand Server tags in HTML pages are interpreted by the OnDemand Server. The HTML pages use the tags to keep track of the Web server extension (CGI, NSAPI, ISAPI), and the user authentication method (native OnDemand Server, Database, or JavaBean). They can also control how the HTML pages are presented to the user.

The OnDemand Server tags are:

- <ODSSecurityTag>
- <ODSLogonMethodTag>
- <ODSChangePasswordTag> (only in ODS classic)
- <ODSGetDocListMethodTag> (only in ODS classic)
- <ODSDocListTag>

<ODSSecurityTag>

The `<ODSSecurityTag>` returns a `TRUE` if the OnDemand Server is set to use OnDemand Server authentication and `FALSE` if it is set to use either Database or JavaBean authentication.

From the `default/logon.html` file:

```
function showLogon() {
   if (<ODSSecurityTag>) {   // if this comes back as true, means
change password option should be available
       var logonPage =
location.pathname+'?HTMLFile="default/logon.changepwd1.html"'
   }
   else {
       var logonPage =
location.pathname+'?HTMLFile="default/logon.normal.html"'
   }
   location.replace(logonPage)
}
```

<ODSLogonMethodTag>

The URL of the OnDemand Server replaces this tag. This tag works with a generic HTML FORM allowing the browser to set the logon form to the right place (for example, `action=/ods-isapi/ods.ods?/Method=logon &HTMLfile=insight2000floating/postlogon.html`). Without this tag, the browser looks in the top level of the HTML directory.

From the `default/logon.normal.html` file:

```
<FORM METHOD="POST" name="logon" action=<ODSLogonMethodTag
HTMLfile="default/postlogon.html">>
```

Variables Supported	Value	Default
`HTMLfile` `=“`*`reference`*`”`	HTML file that is stored in the same directory as logon.html	`Postlogon.html`

<ODSChangePasswordTag>

The URL of the OnDemand Server that sets the logon form to use the Web extension (CGI, ISAPI, NSAPI) used by the OnDemand Server, replaces this tag. Used only in the classic OnDemand Server (version 5.5), this tag works with a generic HTML Form tag.

From the `classic/logon.html` file:

```
PassTag = "<ODSChangePasswordTag
HTMLfile="classic/postlogon.html">"
```

<ODSGetDocListMethodTag>

The URL of the OnDemand Server that allows the HTML page to build a Brio document list, replaces this tag (for example,“/ods-isapi/1.ods?Method=getAllDocuments &HTMLfile=%22reportlist.html%22”). Used only in the classic OnDemand Server (version 5.5), this tag works with a generic HTML Form tag.

```
<frame src=”<ODSGetDocListMethodTag
HTMLfile=”reportlist.html”>”>
```

<ODSDocListTag format=array>

The `postlogon.html` file requests this array from the OnDemand server. The documents available to the user are returned in a JavaScript array. The `loadArray()` function builds the array, and the `foldertree.js` file uses the array to create the folder display. Table 4-5 describes the docList array elements.

When you use the `<ODSDocListTag>` without the `format=array` parameter, the OnDemand server returns a simple list of documents in HTML format. The Insight2000.simple.list provides this look.

Table 4-5 DocList Array Elements

Element	Description
docList[0][0]	Visual name of the doc/folder
docList[0][1]	Adaptive state (0,6, where 0=no access, 6=data model, query, and analyze)
docList[0][2]	URL to request the document
docList[0][3]	Description
docList[0][4]	Folder or document
docList[0][5]	Folder the document lives in
docList[0][6]	Last modified date

<ODSDocListTag>

The list of Brio documents returned by the OnDemand Server replaces this tag. Table 4-6 describes the variables you can set to customize the appearance of the document list when not using the array variable.

```
<ODSDocListTag Target=”briodoc” fontsize=”2” fontface=”Tahoma,
Arial”>
```

Table 4-6 <ODSDocListTag> Variables

Variables Supported	Description	Values	Default
Global Controls			
Target="*window*" (Target is a variable in the Anchor tag.)	Sets a target where the plug-in will get installed. Loads the link into the targeted window. The window must begin with an alphanumeric character, except for the following four target windows: ■ Blank –Loads the link into a new blank window. This window is not named. ■ Parent – Loads the link into the immediate parent of the document the link is in. ■ Self – Loads the link into the same window the link was in. ■ Top – Loads the link into the full body of the window.	Alphanumeric Name	"self"
Fontsize="*number*"	Sets font size for document list	>=1	Not used
Fontface="*fontname*"	Sets font for document list	Examples include Tahoma, Arial, etc	Not used
Includeicons="*yes/no*"	Includes/excludes icons to identify the adaptive report states. Icons are stored in the *images* folder. OnDemand Server uses the following fixed icon names: ■ View Only (bqv.gif) ■ View and Process (bqvr.gif) ■ Analyze (bqi.gif) ■ Analyze and Process (bqir.gif) ■ Query and Analyze (bqy.gif) ■ Data Model (bqdm.gif) If you want custom icons, create six with the names above and replace accordingly	yes/no	Yes
Table Controls			
Usetable="*yes/no*"	Places document in table or not.	yes/no	yes

Table 4-6 <ODSDocListTag> Variables *(Continued)*

Variables Supported	Description	Values	Default
Bordersize="*number*"	Sets border size of the table	>=0	"0"
Tablewidth="*percentage*" or "*number*"	Sets the width of the table	Percentage > 0%	autosizes
Cellpadding="*number*"	Sets cellpadding of the table	>=0	"1"
Cellspacing="*number*"	Sets cellspacing of the table	>=0	"2"
Linktdwidth="*percent-age*" or "*number*"	Sets the width of the <td> tag of the table	Percentage > 0% or pixel > 0	autosizes
bordercolor="color"or "*RGB value*"	Sets the border color of table	One of the browser accepted colors names like red, green, blue, etc. or a RGB value in the form of "#RRGGBB" where RR, GG, BB are hexadecimal numbers from 0 to 255. For instance, "green" is also RGB value "#008000"	Not used
bgcolor="color"or "*RGB value*"	Sets the background color of table	One of the browser accepted colors names like red, green, blue, etc. or a RGB value in the form of "#RRGGBB" where RR, GG, BB are hexadecimal numbers from 0 to 255. For instance, "green" is also RGB value "#008000"	Not used
align="*left/right/center"*	Aligns the text and icons inside the table cells	Left/right/center	left

Zero Administration

Zero Administration identifies the version numbers of the most up-to-date plug-ins on the server, and downloads and updates the plug-ins when it detects older versions of the plug-ins running on the client. The Zero Administration logic is executed by the Web server (not the OnDemand server operating behind the Web server), therefore, multiple nodes have no effect on how Zero Administration works.

Note For modifying the Zero Administration code to allow different users to download different client modules, see the technical note, *Controlling Zero Admin Downloads by User*, located at `www.brio.com/support_services/technical_support/technical_notes.html`.

Server Processing

After the user logs on to the OnDemand server, the Web server presents the document list, including a reference to the JavaScript file, *zeroadmin.js*. This file includes Zero Administration logic and the most recent version of the Brio Web clients in a parameter line. The Brio Intelligence installer sets this number when the OnDemand Server is first installed.

If the BES Administrator replaces the Brio Web client installers with newer files (for example, from downloads), the version parameter will have to be changed manually in *zeroadmin.js*. When using the Brio Intelligence installer to replace Web clients, the *zeroadmin.js* file updates automatically.

Note In Brio.Enterprise 6.5 you can remove anonymous access on the Web server to make it more secure. The Brio plug-ins will prompt for username and passward. This username and password is stored by the plug-in for the entire duration of the browser session.

Client Processing

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.

The *zeroadmin.js* file is retrieved from the Web server. The version numbers from that file are compared to the version numbers on the client's computer. There are three possible outcomes:

- If no version number is found on the client the user is prompted to install.
- 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. 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 upgrade their product.

Client versioning information is stored and read differently for each platform/browser combination, using a combination of plug-in versioning information, "cookie" files, the browser's internal registry, and/or the Windows system registry. 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 by locating the plug-in *dll* files (for example, in Windows NT find the *npbqs32.dll* file for Insight or the *npbqv32.dll* file for Quickview) and displaying their file Properties.

Note — Zero Administration works differently for Netscape and Internet Explorer (IE). Netscape knows what plug-ins are installed regardless of the server. IE requires a cookie file, and that cookie file is targeted by its name to a specific server.

See Table 4-8 for specific information on platforms and browsers.

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.

The JavaScript automatically provides the correct application (plug-ins for Windows platforms; helpers for Macintosh and UNIX platforms) in a browser- and platform-compatible file format.

☆ Tip For Netscape users, you should set the following preferences in Netscape (Edit, Preferences): enable Java, enable Javascript, accept all cookies, and enable Smart Update; then restart your browser.

Note In a BrioONE deployment, Zero Administration is triggered from the login page for Brio.Portal before the user logs in for the first time.

Managing Upgrades

When a new release of the Brio Web components is available, the BES Administrator must upgrade the installer files and upgrade the */zeroadmin/zeroadmin.js*.file to include the current version number.

To upgrade to a new version:

1 Download Web components from www.brio.com/support_services/technical_support/download_updates.html.

2 Follow the directions on the Web site to upgrade the Brio Web client installers.

3 Manually modify the *zeroadmin.js* with the new version numbers.

The following is an example.

```
VerMajor = 6/6/Major Version Number
VerMinor = 0/2/Minor Version Number (dot release)
VerPatch = 0/2/Patch Version Number
VerBuild = 20/27/Daily Build Number
```

These numbers are compared to the end user's version to determine if an upgrade is required. If the end user's product is outdated, the user receives the latest release automatically. (See page 4-22 for information on client processing.)

This process upgrades the end user's current product. When the user logs in to Brio Intelligence, the newly updated *zeroadmin.js* script is run. The script will read information from the user's system and upgrade their client application (Quickview, Freeview, or Insight).

If users want to migrate from one product to the another, for example from Freeview to Insight, they must uninstall (in Windows NT use the install/uninstall program from the control panel) the plug-in or delete (for UNIX) the helper application and associated cookie files and start again.

Once the user returns and downloads a periodic upgrade, the association will be maintained, as long as each download overwrites the preceding instance of the file. For this reason, it is recommended that users always install helper upgrades of Brio Web clients to the default directory location.

Note If users will be processing from a Macintosh version of Insight, you may need to adjust the advanced TCP/IP options at the user desktop to retrieve data correctly. From the Macintosh TCP/IP Control Panel, Edit the User Mode options and enable Load Only When Needed. OnDemand Server supports Open Transport, not Mac TCP.

Files Downloaded

The following five files (the file sizes are approximate) are downloaded by Zero Administration during an installation or an upgrade of the plug-ins:

- bqinsght.cnt (15KB)
- bqinsght.hlp (2040KB)
- npbqs32.dll (7940KB)
- inetwh32.dll (54KB)
- roboex32.dll (311KB)

The date of the five files will be the date you installed them.

For Netscape users, the cookies.txt file will contain an entry beginning with the URL to the OnDemand Server.

Note For Windows, user profile information previously stored in Brioplg.ini is now stored in the registry under `\\HKEY_CURRENT_USER\Software\Brio\BrioPlugin`. *See* Appendix A, “Brio Intelligence Windows Registry Entries.”

Zero Administration Tables

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.

Table 4-7 and Table 4-8 describe the terms used in Table 4-9.

Table 4-9 indicates which techniques are used for all supported platforms.

Table 4-7 Storage Techniques for Web Clients

Version # storage	Indication
Netscape registry	This is a binary file, stored in the Windows directory as nsreg.dat.
Cookies	These are text files. Cookies can reside together in a single file (as in Netscape) or each cookie can have its own text file (as in Windows). Persistent cookie files 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.

Table 4-8 Platform Specific Archiving Software

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 hexidecimal to ensure portability across platforms. Used for Macintosh files.	No
SE-EXE	Self-extracting executable, used for Windows 3.x platforms.	No

Table 4-9 Zero Administration Specifications

		Netscape 4.X	MS Internet Explorer 4.X	MS Internet Explorer 5.X
AIX	ZAC Installer	JAR	N/A	N/A
	Version # storage	Netscape Registry	N/A	N/A
HP-UX v.10	ZAC Installer	JAR	N/A	N/A
	Version # storage	Netscape Registry	N/A	N/A
SunOS Spar 5.7,5.8	ZAC Installer	JAR	N/A	N/A
	Version # storage	Netscape Registry	N/A	N/A
Mac PPC	ZAC Installer	Bin - Hex	Bin - Hex	Bin - Hex
	Version # storage	Cookies	Cookies	Cookies
Win95	ZAC Installer	JAR	CAB	CAB
	Version # storage	Netscape Registry	Cookies	Cookies
Win NT	ZAC Installer	JAR	CAB	CAB
	Version # storage	Netscape Registry	Cookies	Cookies

Summary

Customizing the look and feel of the Brio OnDemand Server can be as simple as disabling the Zero Administration feature, or as complicated as designing from scratch all of your HTML pages. This chapter describes the elements necessary to customize the OnDemand Server user interface, and shows you how to use the Brio sample HTML pages to easily customize your OnDemand Server.

5 Connection or OCE Files

Brio Intelligence uses Open Catalog Extension (`OCE`) files to connect to data sources. An `OCE` is a file that retains all the information necessary to logon to a specific configuration of database and connection API software. In addition, an `OCE` file retains DBMS-specific connection preferences and, for Brio Intelligence connection files, specifications for automatic access to metadata.

The Server Administrator provides a Database Connection Wizard that steps you through the `OCE` file creation process. You can then save your `OCE` files to reuse or distribute as necessary.

This chapter contains the following sections:

- Types of Connections
- Creating an OCE File

Types of Connections

Brio Intelligence Server uses connection files for the following types of connections:

- Polling connections - Brio's Broadcast Server uses polling connections to poll job repository tables for pending jobs. A polling connection must exist for each database you plan to use as a job queue.
- Repository connections - The OnDemand Server uses repository connections to connect to the database on which the OnDemand Repository resides.
- Processing connections - Both the Broadcast Server and the OnDemand Server use processing connections to process scheduled or requested documents. A Broadcast Server polling connection or an OnDemand Server repository connection can use multiple processing connections, which enables documents to be stored in a centralized repository even when they are processed against different databases.

All connection files are stored in the Open Catalog Extension (.oce) directory. In Windows NT, the default location of the .oce directory is `Program Files\Brio\Brio Enterprise Server\Program\Open Catalog Extensions. In UNIX,` the default location is `\install_dir\opencat.`

Polling and repository connections should be distinct from processing connections for the following reasons:

- Using the same OCE file for polling and processing can result in a complete machine shut-down or other problems when certain combinations of databases and database connection software connection software are configured.
- Using a different OCE file for the processing connection enhances processing time because a single connection does not have the dual purpose of handling the polling and the processing.
- Maintaining separate polling and repository connections makes these files more secure. These OCE files should only be used by the administrators who need access to the job queue (Broadcast Server) or the repository tables (OnDemand Server).

Creating an OCE File

You create a connection file with the Database Connection Wizard through the Server Administrator.

Database Connection Wizard

To create an OCE file through the Server Administrator using the Database Connection Wizard:

1. Click **Connections** on the top bar, and then click **Create**.

   The first page of the Database Connection Wizard appears. See page 5-4 for the field definitions.

2. Select the connection software you want to use to connect to the database server from the drop-down list in the **What connection software do you want to use?** field.

3. Select the database server you want to use from the drop-down list in the **What type of database do you want to connect to?** field.

4. To configure metadata settings, click the **Meta Connection Wizard**.

   For instructions on metadata settings and the Meta Connection Wizard see the Brio Intelligence Designer or Brio Intelligence Navigator online help.

5. To configure advanced connection preferences, click **Show advanced options**.

   Advanced options available to you depend on your connection configuration. If any feature described on page 5-6 does not appear on the advanced options dialog pages, then the database and connection software configuration do not support that feature.

6. Click **Next**.

   The second dialog box of the wizard appears. See page 5-5 for the field definitions.

7. Depending on your database, enter your user name in the **User Name** field, your password in the **Password** field, the IP address, ODBC database source or server alias name in the **Host** field and click **Next**.

8. If you selected to work with metadata settings, the Meta Connection Wizard launches.

For instructions on metadata setting and the Meta Connection Wizard see the Brio Intelligence Designer or Brio Intelligence Navigator online help.

9 If you selected Show Advanced Options, the first advanced options dialog box appears.

Only features supported by your database and connection software will appear. See page 5-6 for the field definitions.

10 The wizard prompts you to save your connection file. To save the connection file so that it can be reused or modified, click **Yes**.

The Save dialog box appears.

11 The Service Administrator saves the connection file in the default OCE directory.

To save the connection file in a different directory, navigate to the desired directory and click Save.

Database Connection Wizard Field Definitions

First page:

What connection software do you want to use?	Select the connection software with which you want to connect to your database from the drop-down list. Depending on the connection software you select, additional fields may appear in this dialog box. These fields allow you to customize your connection file; show metadata settings; and select ODBC logon dialogs.
What type of database do you want to connect to?	Select the type of database to which you want to connect from the drop-down list.

Show Metadata Connection Wizard	Click this field, to view and edit meta data settings. Brio Intelligence’s Metadata Definitions dialog box is configured with specific SQL statements to read metadata on multiple databases. For more information see Brio Intelligence Designer or Brio Intelligence Navigator online help.
Show advanced options	To select advanced preferences for the connection file, click this field. Connection preferences enable you to select instructions and protocols your database connection should observe. The preferences are saved with the connection file and applied each time you use the connection. For example, you can use connection preferences to filter extraneous tables from the Table Catalog or specify how the connection software should manage SQL statements. Connection preferences vary depending on your connection software and database
Prompt for database name	Click this field to select the specific database name on the server.
Use ODBC Logon Dialogs	Click this field, if you want to use the ODBC logon dialog boxes instead of the Brio Intelligence dialog box.

Second page:

User Name	Enter the name you want to use to sign onto the database.

Password	Enter the password you want to use to sign onto the database
Host	Enter the IP address, database alias, or ODBC data source name.

Advanced options page:

Apply Filters to restrict the tables that are displayed in the table catalog	Allows specification of table filter conditions for limiting or customizing the list of tables in the table catalog. See “Apply Filters” on page 5-11.
ALLOW SQL-92 Advanced Set Operations	Allows support for the Intersection and Difference operators in the Append Query option. For more information see Brio Intelligence Designer or Brio Intelligence Navigator online help.
Exclude Brio Repository Tables	Specifies exclusion of all Repository tables from the table catalog. "Apply Filter" and "metadata" definitions override this preference. This option is only available in the Designer and Explorer editions.
Allow Non-Joined Queries	Prohibits processing when topics are not joined in the Query Contents pane. This option is only available in the Designer and Explorer editions.
Use SQL to get Table Catalog	Specifies use of SQL to retrieve tables, instead of using SQL Server sp_tables and sp_columns stored procedures. This option allows table filtering, but may be slower than stored procedures. (Sybase and MS SQL Server)

Choose the Data Retrieval Method	Specifies how the server returns data. In most cases, “Retrieve data as Binary is the most appropriate, and fastest method. Choose “Retrieve data as Strings if your connection software does not support native datatype retrieval, or if queries return incorrect or unreadable data.
Time Limit ____ Minutes	This setting will establish an automatic disconnect from the database after the specified period of inactivity.
Auto Commit After Select	Sends a commit statement to the database server with each Brio Intelligence SQL statement to unlock tables after they have been used. Use this feature if tables are locked after use or users experience long waits for tables.
Save OCE Without User Name	Allows general distribution of an OCE by saving it generically, without a user name. Instead, any authorized database user can log on by typing their own user name.
Use Quoted Identifiers	Specifies that internal keywords or table and column, or owner names with special characters sent to the server be enclosed in quotation marks. For example, `SELECT SUM("AMOUNT"), "STORE_ID" FROM "BRIO"."PCS_SALES"GROUP BY"STORE_ID"` The default value for new connections is off.
Allow Change Database at Logon	Adds Database field to logon dialog box allowing the user to choose a specific database when logging on to the DBMS. (Sybase and MS SQL Server)
Server Dates	See “Server Date Formats” on page 5-9.

Use large buffer query mode	Specifies a binding process to retrieve more records per fetch call. If the ODBC driver supports binding, use this option for faster retrieval. (ODBC only). If this feature is turned on, the ODBC Extended Fetch call requests data at 32k at a time

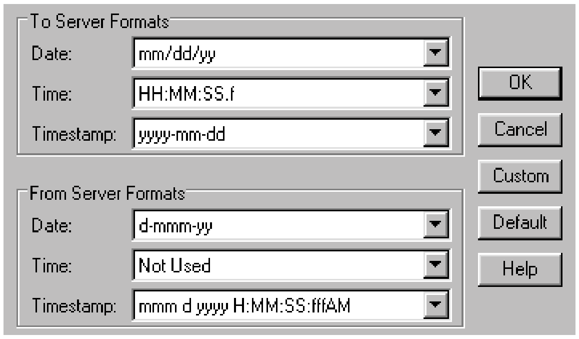
Packet Size Setting.512*	The Packet Size setting allows Sybase's DB-Lib users to set up a large buffer retrieval from the database so that more data can be transferred at one time.
	
	If this feature is selected, you can specify a multiple of 512 bytes for the number of bytes you want to transfer at one time.
	Before you specify a multiple of 512 bytes, your server must have enough memory to allocate for the transmission of the selected packet size.
	
	To check which packet size the Sybase server will support, run the isql command
	
	`sp_configure` and type `go`.
	A list of parameters is returned. Find the parameter showing the "Maximum Network Packet Size". If the packet size you entered exceeds the maximum packet size, you will have to reenter a smaller packet size.
	
	To change the packet size, issue the following command in isql
	`Sp_configure maximum network packet size.`
	`<new value>`
	`(where <new value> is the new size)`.

Retain Date Formats	Brio Intelligence uses the default formats specified by the database server when handling date, time, and timestamp values. If the server's default formats have been changed, you can retain or preserve these adjusted preferences to ensure Brio Intelligence interprets date/time values correctly.

Server Dates	Allows alteration of internal Brio Intelligence date handling to match server default settings in case of a discrepancy. For more information, see "Server Date Formats" on page 5-9.
Disable Transaction Mode	On upload to the Repository, Brio Intelligence brackets SQL Insert statements with transaction statements. Disable Transaction Mode if your RDBMS does not support transactions. This feature is only available in the Designer edition.
Disable Asynchronous Processing	Turns off the ability to make simultaneous requests to the database server. This feature is only available in the Designer edition. For the repository connection, disable the Asynchronous Processing option if your RDBMS does not support simultaneous requests to the database server. For the processing connection, enable the Asynchronous Processing option if your RDBMS does not support simultaneous requests to the database server.

Server Date Formats

Brio Intelligence uses the default formats specified by the database server when handling date, time, and timestamp values. If the server's default formats have been changed, you can adjust preferences to ensure that Brio Intelligence interprets date/time values.


To Server Formats
Date: mm/dd/yy
Time: HH:MM:SS.f
Timestamp: yyyy-mm-dd
From Server Formats
Date: d-mmm-yy
Time: Not Used
Timestamp: mmm d yyyy H:MM:SS:fffAM

To modify server date formats:

1 In the Database Connection Wizard, in the Show advanced options section, click the **Server Dates** button.

The Server Date Formats dialog appears.

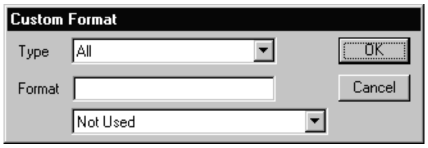
"To Server Formats" refer to date and time formats submitted to the server (such as limit values for a date or time field). "From Server Formats" refer to formats Brio Intelligence expects for date/time values retrieved from the server. The default values displayed in the "To" and "From" areas are usually identical.

2 If the server defaults have changed, select the Date, Time, and Timestamp formats that match the new server defaults from the "To" and "From" format drop-down lists.

If desired, click the Default button to restore all values to the server defaults stored in the connection file.

3 If you cannot find a format that matches the database format, click the **Custom button**.

The Custom Format dialog appears.


Custom Format
Type: All
Format:
Not Used

4 Select a data type from the Type drop-down list.

5 Select a format from the Format drop-down list or type a custom format in the Format field and click **OK**.

The new format will appear as a menu choice in the Server Data Formats dialog box.

Field Definitions

To Date, Time, or Timestamp	Formats Brio Intelligence submits to the database server.
From Date, Time, or Timestamp	Formats Brio Intelligence expects when retrieving data from the database server.

Apply Filters

On databases with many tables, it can help to filter out tables you don't need from the Table Catalog. The Table Filter allows you to specify filter conditions based on table name, owner name, or table type (table or virtual views).

Typically, you filter tables when creating a connection file; although, you can modify an existing connection file later to add filters.

☆ **Tip** The Table Filter works with all database server connections except ODBC. If you are working with a Sybase or Microsoft SQL Server database, create the connection with the option, Use SQL to get Table Catalog, then modify it to filter tables.

To filter tables:

1 Follow the first 8 steps for creating a database connection file on page 5-3.

2 Click **Define** next to a **Filter By** checkbox to filter tables by owner, table or type.

A Filter dialog box appears. The Filter dialog boxes resemble and operate using the same principles as the Limit dialog box. For more information see Brio Intelligence Designer or Brio Intelligence Navigator online help.

3 Select a comparison operator from the drop-down list. Your filter constraints determine which tables will be included in the Table Catalog.

Complete a filter definition by doing one of the following:

- Enter constraining values in the edit field and click the check mark.
- Click **Show Values** to display a list of potential database values and select values in the panel.
- If you are comfortable writing your own SQL statements, click **Custom SQL** to code table filters directly with greater flexibility and detail. For example, you can write a SQL filter, which allows only tables beginning with “Sales” to be displayed in the table catalog. As new "Sales" tables are added to the database, they automatically appear in the Table Catalog.Your custom SQL condition is added to the WHERE clause.

4 Select any other customizing options to apply, and click **OK**.

You are prompted to save the filter settings. Once saved, a checkmark appears in the appropriate filter checkbox, which you can use to toggle the filter on, and off.

5 Continue with the instructions on page 5-4 for creating a database connection file.

Summary

This chapter discusses the different types of connection (OCE) files, and details the fields in the Database Connection Wizard which is used to create all OCE files.

A

Brio Intelligence Windows Registry Entries

Prior to the 6.5 release, the Brio Intelligence product suite stored user profile and application state information in `INI` files located in the `/Windows` or `/winnt` directory on the user's machine. These `INI` files were used by the application to track a variety of things ranging from the most recently used document list, to the serial number that identifies the Brio Intelligence product as Designer, Explorer, or Navigator.

With the release of version 6.5, the information previously stored in a few of the `INI` files is now stored in the Windows Registry. For new installations, the Brio.Enterprise 6.5 installation program makes the appropriate entries to the Windows Registry. For upgrades, the Brio Intelligence application properly migrates the existing `INI` file values to the Registry.

The move to the Windows Registry affects all Windows versions of the Brio Intelligence products—Brio Intelligence (query client tool), Insight, Quickview, Freeview, Broadcast Server, and the OnDemand Server.

Registry Keys and INI File Names

The information from the `INI` files appears in the Windows Registry `HKey_Current_User` keys. These keys are used for the end user applications to support multiple users of the machine retaining independent profile information.

Table 1 lists the specific `INI` file to Registry key mappings.

Table 1 INI File to Windows Registry Key Mappings

INI File Name	Description	Window Registry Key
brioqry6.ini	Contains the general application preferences and personal settings for all Brio Intelligence applications (Designer, Explorer, and Navigator versions).	HKEY_CURRENT_USER/Software/Brio/BrioQuery
brioqplg.ini	Contains the general application preferences and personal settings for all Brio plug-ins (Insight, Quickview, and Freeview).	HKEY_CURRENT_USER/Software/Brio/BrioPlugin
brioqhlp.ini	Contains the general application preferences and personal settings for all Brio.Helpers (stand-alone versions of Insight and Quickview).	HKEY_CURRENT_USER/Software/Brio/BrioHelper

Table 2 lists the `INI` files still used by Brio Intelligence.

Table 2 Brio Intelligence INI Files

bqformat.ini	Contains the list of format codes by locale, as well as the number format and date format codes. Includes a section for each locale that is predefined by Brio Intelligence. These values are used to populate the format dialogs and drop-down lists with the proper format masks for the selected locale.
bqoem.ini	Contains data that control the specification of alternate help files, splash screens, product names, and other OEM customization features for the Brio Intelligence products.
bqmeta0.ini	Defines the pre-packaged metadata definitions the product makes available. This information is used during the OCE wizard process to access pre-defined metadata definitions.
bqtools.ini	Contains the definitions of the Tools menu items used when a custom menu is defined with the application.
bqtbls5.ini	Manages the information from the process-to-table command. This `INI` file is read from and written to when using the Process To Table command.
bqserv1.ini	Contains application preferences and configuration information used by the Brio Intelligence Server administrative tool and the Broadcast Server.

Registry Values and INI File Mappings

The existing `INI` files map in a straightforward manner to the new registry keys. `INI` files are made up of `name=value` pairs, divided into sections in the `INI` file. When mapped to the registry, the `name=value` pairs are Values underneath keys, and the Sections are mapped to subkeys.

B Database Table Reference

This appendix provides information on the tables used by the Brio Intelligence Server. The Broadcast Server uses a job repository of eleven dedicated database tables to store jobs and scheduling details. The OnDemand Server uses a job repository of eight tables to support OnDemand Server. Row Level Security uses three tables.

This appendix contains the following sections:

- Broadcast Server Tables
- OnDemand Server Tables
- Row Level Security Tables

Broadcast Server Tables

Broadcast Server version 6 adds Process to Database Table and FTP actions, and also allows users to register a document for OnDemand Server when they schedule it from BrioQuery. Process to Database Table will use one additional table in the job repository, BRIOSTBL, which logs the tables created by user-scheduled jobs.

Upgrading from Earlier Versions

If you are upgrading from version 5.x, you can simply add new BRIOSTBL tables according to the specifications below. If you do not plan to use the Process to Database Table feature from Broadcast Server, the BRIOSTBL table is not required. Broadcast Server version 6 is completely compatible with existing jobs loaded in the 5.0 job repository.

If you are upgrading from Brio.query.server version 1.0, you can update your existing job repositories and continue running old jobs, as well as new jobs scheduled with version 6. Run the appropriate SQL script, *`ODS6ud.sql`*, which is installed with Broadcast Server.

If you previously upgraded and continue to run version 1.0 jobs, retain the original BRIOVLMT table in the job repository until old jobs are fully migrated. 5.0 versions of Broadcast Server store variable limits in the BRIOACTN table.

Adding a BRIOSTBL Table

The BRIOSTBL table is an optional extension of the job repository which logs table output of Broadcast Server. BRIOSTBL logs tables that are created by a specific database server; an instance of this table must exist on each database supported by a processing connection.

Create BRIOSTBL tables on each processing database according to the specifications described in the job repository schema (see “BRIOSTBL” on page B-9). Make sure to grant access privileges accordingly.

Job Repository Schema

The Broadcast Server job repository tables are detailed in the diagram and tables below.

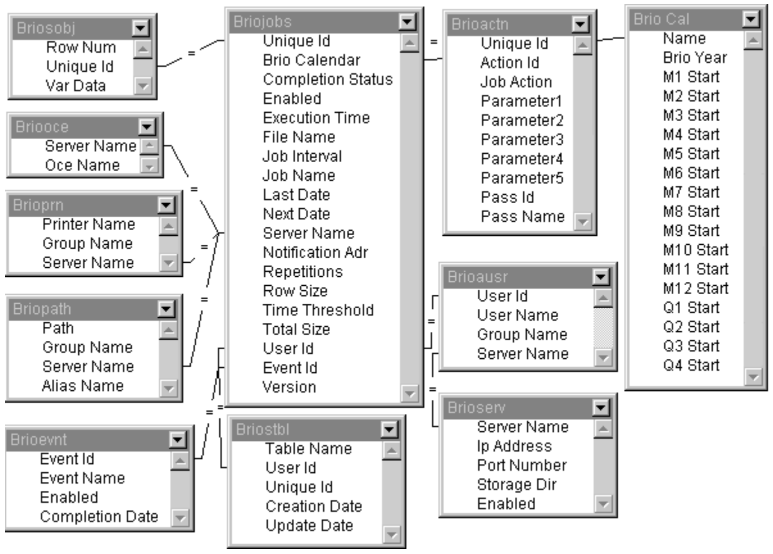


Figure B-1 Broadcast Server Job Repository

BRIOJOBS

The BRIOJOBS table records general information that describes each scheduled job.

Table B-1 BRIOJOBS Table

Field	Datatype	Description
BRIO_CALENDAR	CHAR	Indicates whether a job is scheduled with reference to a custom calendar system. The field is referenced with JOB_INTERVAL by the *Server* to determine the literal date of the next run.
COMPLETION_STATUS	CHAR	Status of most recent job run, populated by the *Server*. Values include: "SUCCESSFUL" -- indicating job ran to completion successfully on LAST_DATE; "FAILURE" -- indicating job failed to complete; "RUNNING" -- indicating job is currently running; "Run ASAP" -- indicating job ran as soon as possible; and "UNKNOWN".
ENABLED	CHAR	Indicates whether job is enabled ('Y') or disabled ('N'). Disabled jobs are not run but remain in the job queue.
EXECUTION_TIME	CHAR	Scheduled run time.
FILE_NAME	CHAR	File name of Broadcast Server document associated with the job.
JOB_INTERVAL	CHAR	Keyword(s) indicating the scheduled interval between job runs. The *Server* uses JOB _INTERVAL and LAST_DATE to compute NEXT_DATE .
JOB_NAME	CHAR	Descriptive job name.
LAST_DATE	DATE	Date and time of previous run.
NEXT_DATE	DATE	Date and time of next scheduled run.
NOTIFICATION_ADR	CHAR	Address for email notification of job status following a run.
REPETITIONS	NUM	Number of times the job will run. The *Server* reads this field after each completed run and subtracts 1 from the total. When 0 is reached, the service disables the job. If null, the job runs indefinitely.
ROW_SIZE	NUM	Reference of rows used in BRIOSOBJ table to store the Broadcast Server job document.
TIME_THRESHOLD	NUM	Overflow time allowed to complete a run. If current server time > EXECUTION_TIME + TIME_THRESHOLD, the job is postponed until NEXT_DATE.
TOTAL_SIZE	NUM	Reference of total bytes used to store the Broadcast Server job document in the BRIOSOBJ table.

Table B-1 BRIOJOBS Table *(Continued)*

Field	Datatype	Description
UNIQUE_ID	NUM	Unique identifier for a job (primary key).
USER_ID	CHAR	User database Logan account associated with the job.
SERVER_NAME	CHAR	The instance of Broadcast Server which will process the job.
EVENT_ID	NUM	Unique identifier for an event used as a trigger to schedule the job (primary key).
VERSION	INT	Identifies jobs by the version of Broadcast Server used to schedule them. If version 1.0, this column is null.
PRIORITY	INT	Priority level for jobs scheduled at the same time. Default value is 0 (range = –100 - +100).

BRIOAUSR

The BRIOAUSR table records the list of operating servers and the users and associated group privilege levels authorized under each server instance.

Table B-2 BRIOAUSR Table

Column	Datatype	Description
USER_ID	CHAR	The unique database login ID of an authorized user (primary key).
USER_NAME	CHAR	Optional descriptive name to help the administrator identify the user .
GROUP_NAME	CHAR	The group the user is assigned to for scheduling and distribution privileges.
SERVER_NAME	CHAR	A Broadcast Server instance under which the user and group are authorized.

BRIOSOBJ

The BRIOSOBJ table stores the document associated with each scheduled job.

Table B-3 BRIOSOBJ Table

Column	Datatype	Description
UNIQUE_ID	NUM	Unique identifier for a scheduled job (primary key).
ROW_NUM	NUM	Sequence ID for segment of the job document.
VAR_DATA	BLOB or LONG RAW	Job document in binary “chunk” format.

BRIOACTN

The BRIOACTN table records descriptions of the output actions specified for each scheduled job.

Table B-4 BRIOACTN Table

Column	Datatype	Description
UNIQUE_ID	NUM	Unique identifier for a scheduled job (primary key).
PASS_ID	NUM	A unique ID for a report cycle of the job.
PASS_NAME	CHAR	The name of the report cycle.
ACTION_ID	NUM	Unique identifier for an assigned output action of the report cycle.
JOB_ACTION	CHAR	"Export", "Print", "Save", or "Email" action description.
PARAMETER1	CHAR	Records a parameter required for the associated action. Values are dependent on the action and can include document section name, export format, retain results, printer, directory or email address.
PARAMETER2	CHAR	See Parameter1.
PARAMETER3	CHAR	See Parameter1.
PARAMETER4	CHAR	See Parameter1.
PARAMETER5	CHAR	See Parameter1.

BRIOOCE

The BRIOOCE table records the polling and processing connections used by the Broadcast Server to poll the job repository and process documents.

Table B-5 BRIOOCE Table

Column	Datatype	Description
SERVER_NAME	CHAR	A Broadcast Server instance.
OCE_NAME	CHAR	The full path of an OCE (relative to the server executable) which is available to Broadcast Server for polling and processing.

BRIOPRN

The BRIOPRN table records the printers available to each user group.

Table B-6 BRIOPRN Table

Column	Datatype	Description
PRINTER_NAME	CHAR	A network printer available to the server for job output.
GROUP_NAME	CHAR	A user group which has access to the printer.
SERVER_NAME	CHAR	A Broadcast Server instance under which the printer and group are authorized.

BRIOPATH

The BRIOPATH table records the default directories available to each user group.

Table B-7 BRIOPATH Table

Columns	Datatype	Description
PATH	CHAR	A network or local (relative to the server) directory path available for job output.
GROUP_NAME	CHAR	A user group which has access to the directory.
SERVER_NAME	CHAR	A Broadcast Server instance under which the directory and group are authorized.
ALIAS	CHAR	An alias for the path name (used as a reference by end-users).

BRIOEVNT

The BRIOEVNT table records events which the administrator defines to trigger job runs.

Table B-8 BRIOEVNT Table

Column	Datatype	Description
EVENT_ID	NUM	Unique identifier for an event (primary key).
EVENT_NAME	CHAR	The name of a particular event used as a trigger for scheduling and running jobs.
ENABLED	CHAR	Indicates whether or not the event is enabled or disabled, affecting its availability for scheduling or running event-driven jobs.
COMPLETION_DATE	TSTMP	The last date the event occurred. If > LAST_DATE in the BRIOJOBS table, the job is run. This field must be updated by the administrator on completion of the external event in order for jobs scheduled against the event to run properly.

BRIOSTBL

The BRIOSTBL table records database tables created through the Process to Database Table feature. The BRIOSTBL table is not required if you do not use Process to Database Table.

Table B-9 BRIOSTBL Table

Column	Datatype	Description
TABLE_NAME	CHAR	The name of a table created by Broadcast Server.
USER_ID	CHAR	The database user name of the user who scheduled the job, and the database account used by Broadcast Server to create the table.
UNIQUE_ID	CHAR	A concatenation of the name of the processing connection and the unique ID of the job.
CREATION_DATE	TSTMP	Date and time the table was created by the server.
UPDATE_DATE	TSTMP	Date and time the table was last updated by the server.

BRIO_CAL

The BRIO_CAL table records custom calendars used to schedule jobs.

Table B-10 BRIO_CAL Table

Column	Datatype	Use
NAME	CHAR	The name of a calendar referenced by a scheduled job (primary key).
BRIO_YEAR	NUM	The year used in the calendar.
Q1_START...	DATE	The starting date of each quarter.
M1_START...	DATE	The starting date of each numbered month.

BRIOSERV

The BRIOSERV table allows documents to be scheduled directly to the Broadcast server machine instead of being stored in a database table. This makes document retrieval faster and more efficient. When the client schedules a job, the user selects which server to schedule the job on by toggling a flag in the ENABLED column. This table is new in version 6.

Table B-11 BRIOSERV Table

Column	Datatype	Use
SERVER_NAME	STRING	The name of the assigned Broadcast Server; related to the SERVER_NAME column in the BRIOJOBS table.
IP_ADDRESS	STRING	The network address of DNS name of the machine that the Broadcast server is running on.
PORT_NUMBER	NUM	The port number the specific Broadcast server machine uses to receive scheduled documents. (This is set in the Server Administrator).
STORAGE_DIR	STRING	The path name of the directory where the documents reside.
ENABLED	STRING	A boolean field with a value of Y or N. Y means the Broadcast Server machine is accepting files for scheduling directly to its file system. N means the documents are stored in the BRIOSOBJ database table instead of on the Broadcast Server machine's file system.

OnDemand Server Tables

OnDemand Server uses an expanded ODS Repository to store information. If you already have a repository, you can use the table creation SQL script installed with OnDemand Server to upgrade, or use Brio Enterprise Server Administrator to create new tables.

Brio Intelligence Repository Schema

The Brio Query Repository tables are detailed in the diagram and tables below.

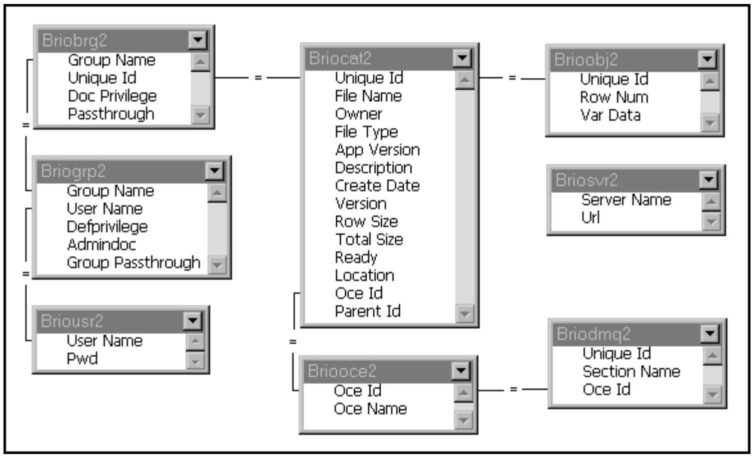


Figure B-2 OnDemand Server Repository

BRIOCAT2

The BRIOCAT2 table records a description of the repository objects and local documents loaded in the ODS Repository. The PARENT_ID column is new in version 6.

Table B-12 BRIOCAT2 Table

Column	Datatype	Description
UNIQUE_ID	NUM	Unique identifier for a stored Repository object or registered OnDemand Server document.
OWNER	CHAR	The creator of the object.
APP_VERSION	CHAR	Version of BrioQuery used to upload the object.
CREATE_DATE	DATE	Most recent date of upload for the object.
ROW_SIZE	NUM	Number of rows occupied by the stored object in the BRIOOBJ2 table.
READY	CHAR	Indicates whether previous upload of the stored object was completed successfully.
FILE_NAME	CHAR	Descriptive name of the stored object.
FILE_TYPE	CHAR	File type of the stored object, such as data model, locked query, locked report, LAN-based, folder.
DESCRIPTION	CHAR	Description of the object.
VERSION	CHAR	The latest version number of the object, used for ADR.
TOTAL_SIZE	NUM	Total size of the stored object in bytes.
LOCATION	CHAR	The complete path to the directory used by OnDemand Server as the document registry.
OCE_ID	NUM	The database connection used to refresh the object or registered document (no longer used for new documents).
PARENT_ID	NUM	The BRIOCAT2 unique ID of the object that contains this object (added to support ODS document hierarchies [folders]).

BRIOOCE2

Lists the connections used to access registered documents in the repository and process them.

Table B-13 BRIOCE2 Table

Column	Datatype	Description
OCE_ID	NUM	Unique identifier for a connection.
OCE_NAME	CHAR	File name of the connection file (`.oce`).

BRIOOBJ2

Stores the actual objects loaded in the Brio Query Repository.

Table B-14 BRIOOBJ2 Table

Column	Datatype	Description
UNIQUE_ID	NUM	Unique identifier for a stored Repository object.
ROW_NUM	NUM	Sequence ID for segment of the object.
VAR_DATA	BLOB or LONG RAW	Data Model object in binary “chunk” format.

BRIOUSR2

Stores user authentication privileges to access the OnDemand Server.

Table B-15 BRIOUSR2 Table

Column	Datatype	Description
USERNAME	CHAR	The name of the user.
PWD	CHAR	Encrypted password that authenticates a Web-enabled user connecting to OnDemand Server; this field is blank if database authentication is used.

BRIOBRG2

Stores the associations between registered documents and Repository groups.

Table B-16 BRIOBRG2 Table

Column	Datatype	Description
UNIQUE_ID	NUM	Unique identifier for a repository document.
GROUPNAME	CHAR	The name of a Repository group.
DOC_PRIVILEGE	NUM	An override privilege level for a member of the group when accessing this document.
PASSTHROUGH	CHAR	Enables members of the group to 'pass through' authentication by the database when processing this document.

BRIOGRP2

Maintains the list of Repository groups, and their associated users and privileges.

Table B-17 BRIOGRP2 Table

Column	Datatype	Description
GROUPNAME	CHAR	The name of a Repository group.
USERNAME	CHAR	The name of a Brio Query Repository user assigned to the group.
DEFPRIVILEGE	NUM	The default privilege level assigned to the group (values = 1-6) .
ADMINDOC	CHAR	Enables members of the group to register documents.
GROUP_PASSTHROUGH	CHAR	Enables members of the group to 'pass through' authentication by the database when processing documents.

BRIOSVR2

Stores the identity and location of the OnDemand Server.

Table B-18 BRIOSVR2 Table

Column	Datatype	Description
SERVER_NAME	CHAR	The name of an instance of OnDemand Server.
URL	CHAR	The URL and port number used to communicate with OnDemand Server through the intranet server.
RM_URL	CHAR	The server name and port number of the Brio Portal you connect to in a BrioOne install. This field is ignored in a stand alone installation.

BRIODMQ2

Maps query sections of a document to processing connections. This table was not included in earlier releases than version 6.

Table B-19 BRIODMQ2 Table

Column	Datatype	Description
UNIQUE_ID	NUM	The object's ID in the BRIOCAT2 table.
SECTION_NAME	CHAR	The name of the query or data model section in the document.
OCE_ID	NUM	The OCE ID used to process for this section.

Row Level Security Tables

These tables are required if you are using the Row Level Security feature. The tables must co-reside (same owner) with the server's repository tables. For more information see "Introduction to Row Level Security", which is available from Brio Systems Consultants.

BRIOSECO

Tells the server that the row level security feature is enabled.

Table B-20 BRIOSECO Table

Column	Datatype	Description
BENABLE	CHAR	A coded variable that tells the server to enable row level security. Y -Enable row level security N - No row level security

BRIOSECG Table

The table defines the users and the groups that are subject to row level security restrictions.

BGROU
Table B-21 BRIOSECG Table

Column	Datatype	Description
BUSER	CHAR	The name of the user.
BGROUP	CHAR	The name of the group the user belongs to.

BRIOSECR Table

Table B-22 BRIOSECR Table

Column	Datatype	Description
UNIQUE_ID	INT	Unique identifier not accessed by the server. Use to maintain the table by whatever means you choose. (optional)
USER_GRP	CHAR	The name of the user or the name of the group the user belongs to.
SRCDB	CHAR	The database where the topic resides (for connections that support multiple data-bases).
SRCOWNER	CHAR	The owner/schema of the topic in the Data Model.
SRCTBL	CHAR	The table/relation identified by the topic in the Data Model.
SRCCOL	CHAR	A column in SRCTBL.
JOINDB	CHAR	The database name qualifier of a table/relation that must be joined to SRCTBL. (optional)
JOINOWNR	CHAR	The schema/owner name qualifier of a table/relation that must be joined to SRCTBL.(optional)
JOINTBL	CHAR	The table/relation that must be joined to SRCTBL. (optional)
JOINCOLS	CHAR	The column names from SRCTBL to be joined to a column from JOINTBL. (optional)
JOINCOLJ	CHSR	The column name in JOINTBL that will be joined to a column from JOINCOLS. (optional)
CONSTRTT	CHAR	A coded value that identifies a table/relation used for applying a constraint (limit). S - Column to be limited is in SRCTBL J - Column to be limited is in JOINTBL O - Restriction on the column named in SRCCOL is not applied to the current USER_GRP even if they are a member of the group named in OVRRIDEG Null - No additional restriction is defined
CONSTRTC	CHAR	The column that a limit is applied to in the table/relation identified by CONSTRTT.

Table B-22 BRIOSECR Table

Column	Datatype	Description
CONSTRTO	CHAR	The constraint operator.
CONSTRTV	CHAR	The value(s) to be used as a limit.
OVRRIDEG	CHAR	The name of a group or user.

C OnDemand Server Error Codes

This appendix lists error codes you may encounter when using OnDemand Server. The following information is included for each error message:

- Error # - the number of the error code.
- Message - the message you will see when this error code is generated.
- Troubleshooting - actions you can try to fix the error.
- Comments - additional information to help you understand the error.
- Severity level - See Table C-1 for information on the severity levels.

Table C-1 Severity Level Definitions *

Severity Level	Description	Definition
1	High	The system has stopped functioning. Requires pager notification. Immediate response required.
2	Medium	A problem that if not taken care of can cause the server to crash. Anything that would interrupt the workflow of the users by 50% or higher.
3	Low	Everything else.

* *Definitions by Morgan Stanley Dean Witter*

Note Some of the severity levels are subjective to whether it's only happening on the client or server side. If these errors occur only on a few client machines, this usually points to a problem with the client configuration. If this is the case, compare the problematic client machines with the ones that are functioning properly.

Table C-2 OnDemand Server Error Codes

Error #	Message	Troubleshooting	Comments	Level
1000	Invalid user (SERVER_INVALID_USER)	■ Check the user list in the Server Administrator. ■ Verify the user ID and password. ■ If using database authentication, verify the user ID and password in the underlying RDBMS.	Most of the error codes in Brio point back to permissions, (OS or DBMS), environment, or file locations. This is the case here.	3
1001	Can not locate document	■ Verify that the path pointing to the documents directory is correct in the ODS.ini file. You can access this setting through the ODS.ini file or through the Server Administrator.	This error occurs when the requested document can not be found in the specified Documents directory.	3
1002	Invalid document name (SERVER_INVALID_DOCUMENT_NAME)	■ Look at the generated URL. Is it pointing to the correct document? ■ If the document's URL is not generated by ODS, ensure that the document referenced by the URL is correct.	This error usually occurs when you do not use an ODS-generated report list to open a document.	3
1003	Invalid database username or password (SERVER_INVALID_DATABASE_LOGON)	■ Look at the account used to process the request (processing connection or individual user) and see if anything has changed.	Accounts can have very complex parameters, (time lockouts, expirations, individual table rights).	3

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1004	Unknown processing error (SERVER_UNKNOWN_PROCESS_ERROR)	■ Check the log file for detailed information (for example, a stack trace). ■ Check that your temporary disk space is not full.	This error typically occurs when your user password changes or gets corrupted. This error also occurs when your query returns no rows because of an error (like you filled up a file system). If there are issues with connectivity, or db errors, it could show up in the *dbgprint* and display 1004 to the user. This error usually occurs on the client side.	2
1005	Invalid URL Path (SERVER_INVALID_URL_PATH)	■ If your Web browser is configured to use a proxy, check to be sure the Brio plug-in is also using a proxy. ■ Is the ODS machine up? ■ Is the daemon running on the correct port? ■ Is the DNS server running? ■ Can you ping the server? ■ Can you see the virtual directories from your browser?	If this error still occurs after you verify the troubleshooting information, the URL to the ODS may not be defined correctly. This error usually appears when the ODS document list is bypassed.	3
1006	Unknown Web communication error (SERVER_UNKNOWN_WEB_ COMMUNICATION_ERROR)	■ Backup the configuration file and reinstall the Web server. (For Netscape Web servers, back up the *obj.conf* file.) If this error occurs, it is more than likely that something is wrong with your Web server. The database may also be down. This error usually occurs when processing, and will be displayed to end users.		1 or 2

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1007	Invalid access privilege (SERVER_INVALID_ACCESS_PRIVILEGE)	■ Ensure that the URL to the ODS is defined correctly.	This error only occurs if the URL to ODS is defined incorrectly and the permissions to see document were revoked after the user received the document list.	3
1008	Invalid administrator username or password (SERVER_INVALID_ADMIN_USERNAME_PASSWORD)	■ Check your typing. ■ Check the spelling of the user name in the *ODS.ini* file. ■ Ensure that the encrypted password stored in the *ODS.ini* file is not corrupt. ■ Does your library path include the path to your middleware libraries?	The password sometimes gets corrupt when users manually edit the *ODS.ini* file. It is a good idea to back up the *ODS.ini* file before doing a manual edit.	3
1009	Can not logon to the OnDemand Server database repository (SERVER_CANT_CONNECT_TO_ADMIN_REPOSITORY)	■ Are you pathed to the middleware library files? ■ Are things where you think they are? ■ Does the Server Repository tab contain the correct information? ■ Is the repository database running?		2

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1010	HTTP Error (SERVER_HTTP_ERROR)	■ Check the Web server mime types. ■ Verify that the host name resolves to the correct IP address. ■ If you browser is set up to use a proxy server, be sure the plug-in is also using a proxy server. ■ If you are using Microsoft's Internet Information Server (IIS), verify that the IUSR_[servername] exists as a user account and that it has permissions to access the Brio Intelligence directories.	This error occurs when the client is denied access to the Web server. This error only occurs on the client side and does not occur if the Web server is set up properly. (See ODSError 1006.)	2
1011	Error while processing an ODS document or Exception error (SERVER_EXCEPTION_THROWN_IN_REC_RESULTS)	■ Restart the OnDemand Server. ■ Ensure that the Brio plug-in and ODS version numbers are in sync. ■ If you are using Netscape 4.0 or above, make sure that you use NSAPI instead of CGI middleware.	It is extremely unlikely you will ever see this error. The error indicates that the results returned are corrupted. If you do see this error, be sure to let Brio know. The error will probably generate *EXECPTION : null* in the *ODSManager.log* or *ODSNode.log* files.	1
1012	Unknown error message (SERVER_UNKNOWN_ERROR_MESSAGE)	■ If this error occurs on the *server* side, restart the OnDemand Server.	If this error occurs on the *client side*, something is wrong with the plug-in, or there is a possible corruption in the BQY file. If this error occurs on the *server side*, something is wrong with the communication between ODS and the database repository. (See ODSError 1011.)	1

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1013	Server cannot serialize request to file (SERVER_CANT_SERIALIZE_REQUEST_TO_FILE)	■ Check permissions on the *tmp* file directory. ■ Check to see whether the file system is full.		1
1014	No database connection information (SERVER_NO_DB_CONNECTION_INFORMATION)	■ Check to see if DNS is confused? ■ Did your OCEs get moved? ■ Are the repository tables linking the docs to the right OCEs? ■ Do a query against the repository tables. Are the OCEs where the tables think they are? ■ Did you do a re-install? If so, *ODS.pwd* may not be valid.	It is possible to see this error if you have processing permissions and no processing connection is defined. The error usually indicates an administration problem.	2
1015	The Brio ODS server could not change the password (SERVER_FAIL_TO_CHANGE_PASSWORD_ERROR)	■ Check rights at the DBMS level. (You need at least *insert* rights.) Are you doing these the same as the user who created the repository tables? ■ Check your typing.		3
1016	The Brio ODS server had an error while processing show values or show remarks (SERVER_WHILE_PROCESSING_SHOWREMARKS_ERROR)	■ Try to log into the database as a different user using Brio Intelligence.	This error indicates a missing OCE or that you do not have select privileges to a specific table or remarks database.	3
1017	BQ command script failed (SERVER_BQ_COMMAND_FAILED_ERROR)	■ Clean up and bounce and then try to install a second time.	This error typically indicates a version issue in a re-install.	3

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1018	Brio OnDemand Server is currently operating at peak level. Please try to connect again later.	■ For Oracle queries, turn off Asynchronous processing in the OCE. ■ Make sure the paths to all Brio Intelligence components are eight-dot-three in the command-line for the procfac and ODS manager. ■ Ensure Brio Intelligence processes can start and run from the command-line. ■ On UNIX, make sure to set the DISPLAY variable correctly. ■ Ensure that the temporary directories are not filled up.	You will usually receive this error when you are attempting to logon tot he OnDemand server.	
1019	ODS failed to execute the query (SERVER_FAIL_TO_EXECUTE_A_QUERY)	■ Ensure that the Brio plug-in and ODS version numbers are in sync. ■ Ensure that the database did not sever the connection.		2

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1020	ODS failed to load the document (SERVER_FAIL_TO_DOWNLOAD_ DOCUMENT)	■ Ensure that the load is not too high on ODS. If the load is too high, try to load the document again. ■ Ensure that the URL defined to retrieve the document is correct. ■ Change the WebClient default configuration settings to *not* pass data using streams instead of files (see the comment to the right for more information). ■ If using MS SQL Server 7.0, change the owner name to the database user (usually *dbo*) in the Server Repository tab of the Brio Intelligence Server Administrator. This may be different from the login name you used when performing the installation.	If there is a firewall, and therefore a random port number is not likely to be usable, the transfer by file option should be used (`PassByDC-Stream` unchecked). The repository will serialize the file and transfer it over the already open socket the service broker is using. Without a firewall, the service broker can use a new socket to transfer files. When this is done (`PassByDC-Stream` checked), the repository will open a socket for the service broker to use, and pass the socket address to the service broker. This address, or port number, is generated by the OS when an opensocket made. There is no way to control what the port number will be. The file is transferred through this socket, as if from a read file command. Transferring by file has the potential to do disk writes, especially with larger files. The file transfer has been optimized over several releases, including memory buffering as much as possible. While there may be some impact using file transfer over using streams, it should not be too great.	2

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
1021	SERVER_FAIL_TO_EXECUTE_SCRIPT			
1022	ODS failed to schedule a job (SERVER_SCHEDULER_ERROR)	■ Check permission to Broadcast Server repository tables. ■ Ensure that the ODS UID has permissions to schedule on BCS.		3
1023	ODS can't save result to a temporary file (SERVER_CANT_SAVE_RESULT_TO_FILE)	■ Check the disk space on the client machine.	This error only occurs on the client side. The most likely cause is that users ran out of disk space.	2
1024	Query is aborted (SERVER_QUERY_ABORTED)		A notification that the user cancelled a query.	3
1025	ODS failed to query a document list (SERVER_FAIL_TO _QUERY_DOCUMENT_LIST)		See ODSError 1016	2
2000	Cannot communicate to process (SERVER_CANT_COMMUNICATE _TO_PROCESS)		The ODS process cannot communicate to the BQ process. Possible version mismatch on a re-install. Watch for this when using Manager/Node configuration. If the Node is not the same version as the Manager you may get this error. Will likely occur during upgrades.	2
2001	Request is not found (SERVER_REQUEST_NOT_FOUND)		Message used during the development of ODS. You will never see this message.	3
2002	Unknown message is received (SERVER_UNKNOWN_MESSAGES)		See ODSError 1012.	1
2003	Interface is not recognized (SERVER_INTERFACE_NOT_FOUND)	■ Clean up between installations.	Most likely a versioning issue between bq and ODS. (See ODSError 1019.)	2

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
2004	Invalid access to document (SERVER_ILLEGAL_ACCESS_TO_ DOCUMENT)		See ODSError 1007.	3
2005	Cannot register document (SERVER_CANT_REGISTER_DOCUMENT)	■ Check permissions at the ODS level. ■ Ensure that your UID is in a group that is allowed to register documents to the OnDemand Server (unless PUBLIC has this right.)	Probably an administration problem.	3
2006	Request is still processing (SERVER_REQUEST_IS_PROCESSING)		This error appears when you hit the process button multiple times.	3
2007	OnDemand Server is currently busy (SERVER_IS_BUSY)	■ Consider adding another node.	This error appears when there are too many concurrent users.	3
2008	Server process is currently in use by another thread (SERVER_IS_INUSE_BY_OTHER_THREADS)		Message used during the development of ODS. You will never see this message.	3
2009	Exception error (SERVER_EXCEPTION_IS_THROWN)		See ODSError 1011.	1
2010	Server is not in sync with clients (unmatched data type) (SERVER_UNMATCHED_DATA_TYPE)		Message used during the development of ODS. You will never see this message.	3
2011	Server is not in sync with clients (invalid memory usage) (SERVER_ILLEGAL_MEMORY_ACCESS)		Message used during the development of ODS. You will never see this message.	3
2012	Server cannot connect to user authentication Java bean host (SERVER_CANT_CONNECT_TO_ USER_AUTH_BEAN_HOST)	■ Check port configuration. ■ Is the java bean still there? ■ Has the java bean been touched recently? ■ Is the java bean listening (netstat -a |grep 9999)	Substitute your Java bean's port number for 9999.	3

Table C-2 OnDemand Server Error Codes *(Continued)*

Error #	Message	Troubleshooting	Comments	Level
2013	New password is not confirmed (SERVER_NEW_PASSWORD _ISNT_CONFIRMED)	■ Check your typing. ■ Ensure that you have insert permission to the repository tables?	Probably an administration problem.	3
2014	Server is being shut down (SERVER_MESSAGE_QUEUE_IS_CLOSED)	■ Try again later.		3
2015	Server cannot find Script file (SERVER_CANT_FIND_SCRIPT_FILE)		Message used during the development of ODS. You will never see this message.	3
2016	Server can not accept local authentication (SERVER_CANT_ACCEPT_LOCAL_AUTH)	■ As a temporary solution set ODS_LOCAL_AUTH_ ACCEPTED=false. This will allow acceptance of local authentication.	Accepting local authentication is not as secure as not accepting local authentication.	3
2017	ODS server version mismatch with client: y.y.y.y ODS expects client with version x.x.x.x and above. Please check your installation. (SERVER_VERSION_MISMATCH)	■ Check your version server and client version numbers. ■ Update the client.		3

D Custom JavaBean Authentication

The Custom JavaBean Authentication mode provides the opportunity to integrate the OnDemand Server with your existing security systems. While Brio does not provide supported JavaBean implementations, this appendix presents a simple JavaBean as a sample of how to write the Java code, and guides you through configuring your OnDemand Server to run in Custom Authentication mode.

See "User Authentication" on page 1-21 for information on all of the OnDemand Server authentication modes.

Note To implement a custom JavaBean authentication, an understanding of the Java Programming language and the workings and administration of the Brio OnDemand Server is required.

Note In BrioOne, NT and LDAP custom authentication is handled through ONE/SERVICES. See the *Brio Portal Administration Guide* for more information.

Use the custom JavaBean authentication mode in the following sample situations:

- For accessing an LDAP directory to lookup user name/password combinations
- For accessing the NT Domain Services security information
- For performing a logon to a mainframe to authenticate with RACF
- For reading user names and passwords from a local text file on the server

The sample JavaBean presented in this paper describes the last situation.

Sample JavaBean

This fully functioning sample shows the basics of how to write a JavaBean to perform OnDemand Server authentication. The sample file reads user name/password pairs from a text file. The user name/password entered by the user on the OnDemand Server login form is passed to this code, where the user name is looked up to retrieve the password associated with that user. The password from the lookup is compared to the password entered by the user. This sample code supports case sensitive or case-insensitive comparisons of the user-entered values.

Input Parameters

The password file contains the user name/password pairs that are passed to the JavaBean by the value of the `BQ_USER_AUTH_HOST_ NAME` entry in the ODS.ini file. The entry in the INI file should be the full path and file names. This file could be an automatic extract from your mainframe security system, or it could simply be a list of names/passwords you manually edit. Either way, be sure to enforce security privileges on this file. The user name the OnDemand Server is running will require read access to the file – while you will probably want to deny access to all other users.

The contents of the input file should be clear text only, no header lines or footer lines. Use a simple list of user names and passwords, one pair per line, separated by either a blank space or a tab character. User names and passwords can not be all numeric or contain blank spaces:

Jim	ceventura
Teri	loislane
Kim	catwomen
Tom	topgun

The `BQ_USER_AUTH_PORT_NUM` entry in the ODS.ini file is used to determine if the user name/password comparisons should be case sensitive or not. A value of 1 means case insensitive, and any other value means case sensitive. If running in case sensitive mode, a password in the input file of “loislane” will not match the user entered password of “LoisLane”. Case sensitivity is for both the user names and passwords.

Configuring the Java Development Environment

While this sample was built with Microsoft J++ 6.0, it is 100% Java. The JavaBean can be built using any Java development environment. Prior to building the JavaBean, you must define the development project to include the main OnDemand Server classes. This is done by including the ODSClasses.jar file as part of your project classes. In Microsoft J++ 6.0, this is done by selecting the Properties item on the Project menu, and moving to the ClassPath tab. Add a new project-specific class, including the …/server/classes/ODSClasses.jar file.

Consult your development environment documentation for details on how to include an external class into your project.

Getting a Copy of the Sample

The Java source code and a compiled Java class file for this sample is available on the Brio ftp server: `ftp://ftp.brio.com/public/support/Scripts_and_samples`. If you choose to extend this sample, start with the Java source code, `ODSLogin.java` (see below), and build your own JavaBean. The class file, `ODSLogin.class`, may be used as is from the ftp site; just follow the directions in this appendix on how to configure the OnDemand Server to use it.

ODSLogin.java

```
// Brio Enterprise Server - OnDemand Server version 5.5.4
// User Authentication JavaBean Sample
// copyright 1998, Brio Technology, Inc.
// all rights reserved
//
// Sample code provided ASIS.
//


package ODSLogin;

import brio.server.*;
import java.beans.*;
import java.io.*;
import java.util.*;

public class ODSLogin extends SimpleBeanInfo
   implements UserAuthentication, Visibility
{
   private String    m_hostName = null;
private int       m_portNum = 0;
private String    m_optionString1 = null;
private String    m_optionString2 = null;
private Hashtable usr_pwd = new Hashtable();


public void setHostName(String hostName)
{
   m_hostName = hostName;
}

public void setPortNum(int portNum)
{
   m_portNum = portNum;
}

public void setOptionStr1(String optionStr)
{
   m_optionString1 = optionStr;
}

public void setOptionStr2(String optionStr)
{
   m_optionString2 = optionStr;
}
public void connect()
   throws CantConnectToHostException, HostNotFoundException
{
   readPasswordFile();
}

public boolean authenticate(String usr, String pwd)
   throws CantConnectToHostException
{
   try {
      String h_pwd = null;
      if (m_portNum == 1) { // case in-sensitive mode
         usr = usr.toLowerCase();
         h_pwd = (String) usr_pwd.get(usr);
         if (h_pwd.equalsIgnoreCase(pwd))
            return true;
         else
            return false;
      }
      else { // case-sensitive mode
         h_pwd = (String) usr_pwd.get(usr);
         if (h_pwd.equals(pwd))
            return true;
         else
            return false;
      }
   }
   catch (NullPointerException e) {
      return false;
   }
   }

   public void readPasswordFile()
   {
       try {
           File pwdfile = new File(m_hostName);
           FileInputStream inFile = new FileInputStream(pwdfile);
           BufferedReader bufferIn = new BufferedReader(new
InputStreamReader(inFile));
           StreamTokenizer tokenIn = new
StreamTokenizer(bufferIn);
           if (m_portNum == 1)  // case-insensitive mode
              tokenIn.lowerCaseMode(true);
           else // case-sensitive mode
              tokenIn.lowerCaseMode(false);
           String usr = null;
           while (tokenIn.nextToken() != tokenIn.TT_EOF) {
              usr = tokenIn.sval;
              if (tokenIn.nextToken() != tokenIn.TT_EOF)
                  usr_pwd.put(usr, new String(tokenIn.sval));
           }
           bufferIn.close();
           inFile.close();
           Log.writeLine("*****Usernames read from external file:
".concat(Integer.toString(usr_pwd.size())));
       }
       catch(IOException e){
           Log.writeLine("*****IOException Reading the password
file");
       }
   }

   public boolean avoidingGui()
   {
       return true;
   }

   public void dontUseGui()
   {
   }

   public boolean needsGui()
   {
       return false;
   }

   public void okToUseGui()
   {
   }

}
```

The interesting parts of the Java code are explained below

```
package ODSLogin;
```

Defines the package name the class will be part of. This will also be the name of the directory you create inside the …/server/classes directory.

```
import brio.server.*;
```

Imports other Java Classes into your code. There are four imported class files specifically included in this project:
brio.server.* -- this is the main OnDemand Server classes which are part of the ODSClasses.jar file.
java.beans.* -- the main JavaBeans classes.
java.io.* -- the Java file i/o routines
java.util.* -- Java utility classes, this sample file makes use of a hash table, which is part of the utility package.

```
public class ODSLogin extends SimpleBeanInfo
```

Defines the ODSLogin Class, which extends the simplebean class. This package name will be the name of the sub-directory you create in the …/Server/Classes directory of you OnDemand Server directory structure.

```
private String    m_hostName = null;
```

The first of five lines that declare variables are used throughout the JavaBean. The first four, m_hostName, m_portNum, m_optionString1, and m_optionString2 are used to hold the values passed into the JavaBean from the calling program. (The two optionStrings are defined here even though their values are not actually passed in version 6.2.x.) These values end up mapping to the same values entered in the configuration dialog (or the ODS.ini file). The fifth variable, usr_pwd, is defined as a Hashtable. This is a special data structure for which Java has built-in support for. A hashtable is basically an array of keys and associated values for the keys. In this sample, we will load user names into the hashtable as the keys, with the passwords being set to the value for the individual keys. The Java language includes support for easily adding and looking up keys and values in a hashtable.

```
public void setHostName(String hostName)
```

Defines the setHostName method, which is called by the OnDemand Server at startup time to pass the entry from the BQ_USER_AUTH_HOST_NAME entry in the ODS.ini file to the JavaBean. In this sample, the value passed in will be the full path and file name of the text file containing the user names and passwords.

```
public void setPortNum(int portNum)
```

Defines the setPortNumber method, which is called by the OnDemand Server at startup time to pass the entry from the BQ_USER_AUTH_PORT_NUM entry in the ODS.ini file to the JavaBean. In this sample, the value passed in will be a 0 or 1, indicating whether the user name and passwords should be treated as case sensitive or not (0=case sensitive, 1=case insensitive).

```
public void setOptionStr1(String optionStr)
```

Defines the setOptionStr1 method, which is called by the OnDemand Server at startup time to pass the entry from the BQ_USER_AUTH_OPT_STR1 entry in the ODS.ini file to the JavaBean. This entry is not used in this sample file, but the routine still needs to exist, as it will be called in the startup sequence.

```
public void setOptionStr2(String optionStr)
```

Defines the setOptionStr2 method, which is called by the OnDemand Server at startup time to pass the entry from the BQ_USER_AUTH_OPT_STR2 entry in the ODS.ini file to the JavaBean. This entry is not used in this sample file, but the routine still needs to exist, as it will be called in the startup sequence.

```
public void connect()
```

Defines the connect method, which is called by the OnDemand Server at startup time. This routine performs any initialization work the JavaBean requires. While the method is named connect, it can be thought of as a generic initialization routine. In this sample, we will be reading a list of user names and passwords from a text file. The connect method, on line 53, simply calls the method to read in the contents of the password file. This method is invoked once, at the time the OnDemand Server startups.

```
public boolean authenticate (String user, String pwd)
```

Defines the authenticate method, which is called by the OnDemand Server each time a user requests to be authenticated (by logging on or by selecting a document). This is where the logic to perform the actual authentication is performed. The OnDemand Server calls this method, passing it two string variables – usr and pwd.

```
String h_pwd = null;
```

Defines a temporary variable, h_pwd, which will hold the value of the password retrieved from the hashtable. This value will be compared to the user-entered password to perform the authentication.

```
if (m_portNum == 1) { // case in-sensitive mode
             usr = usr.toLowerCase();
             h_pwd = (String) usr_pwd.get(usr);
             if (h_pwd.equalsIgnoreCase(pwd))
                 return true;
             else
                 return false;
          }
```

Defines the code to be executed when running in case insensitive mode. The comparisons of user-entered values to the values saved in the hashtable will be made without regard to upper/lowercase characters. The variable m_portNum is used to identify whether case is to be an issue or not – a value of 1 means case is not important.

```
usr = usr.toLowerCase();
```

Converts the user-entered username to lowercase characters.

```
h_pwd = (String) usr_pwd.get(usr);
```

Sets the variable h_pwd to the value retrieved from the hashtable. The command usr_pwd.get(usr) is the Java command to retrieve a value from a hashtable – it automatically searches the hashtable for the named key (passed to it via the usr variable), and returns the value associated with that key.

```
if (h_pwd.equalsIgnoreCase(pwd))
```

Performs a case-insensitive comparison of the user-entered password to the password just retrieved from the hash table.

```
return true;
```

Returns true, meaning the user was authenticated to access the OnDemand Server.

```
return false;
```

Returns false, meaning the user failed the authentication.

```
else { // case-sensitive mode
            h_pwd = (String) usr_pwd.get(usr);
            if (h_pwd.equals(pwd))
                return true;
            else
                return false;
         }
```

Defines the code to be executed when running in case-sensitive mode. The comparisons of user-entered values to the value saved in the hashtable will be made as is – with upper/lower case being significant.

```
h_pwd = (String) usr_pwd.get(usr);
```

Retrieves the value from the hashtable.

```
if (h_pwd.equals(pwd))
```

Performs a case-sensitive comparison of the user-entered password to the password just retrieved from the hash table.

```
return true;
```

Returns true, meaning the user name and password entered by the user exactly matched the values stored in the hashtable.

```
catch (NullPointerException e) {
```

Defines the error handling for a NullPointerException. The lookup in the hashtable for user name (lines 63 and 70) will throw this exception if the user-entered user name does not exist in the hashtable. Catching this exception allows the JavaBean to recover gracefully. In this sample, it returns a false, meaning the user is not authenticated.

```
public void readPasswordFile()
```

Defines the readPasswordFile method. This is the code which physically reads in the list of user names and passwords from a text file, loading them into a hashtable for lookup by the authenticate routine. readPasswordFile is called by the connect method.

```
File pwdfile = new File(m_hostName);
```

Defines the variable pwdfile as a handle to an operating system file – the filename is the value passed to the JavaBean via the hostName variable.

```
FileInputStream inFile = new FileInputStream(pwdfile);
```

Defines a FileInputStream to read the contents of the file handle created in the line above.

```
BufferedReader bufferIn = new BufferedReader(new
InputStreamReader(inFile));
```

Defines a BufferedReader to read the contents of the FileInputStream. Reading from a BufferedReader is more efficient than reading lines one at a time from the file. Java automatically handles the buffering of the file.

```
FileInputStream inFile = new FileInputStream(pwdfile);
```

Defines a StreamTokenizer to read from the buffer. A tokenizer reads from the input file a word at a time. This allows the program to simply read complete words rather than reading individual characters into an array. This tokenizer is what the user names and passwords will be read from.

```
if (m_portNum == 1)  // case-insensitive mode
   tokenIn.lowerCaseMode(true);
```

If running in case-insensitive mode, as determined by the m_portNum value, the tokenizer has the lowerCaseMode set to true. This setting instructs the tokenizer to convert the words it reads to lowercase as they are read.

```
tokenIn.lowerCaseMode(false);
```

If running in case-sensitive mode, the lowerCaseMode for the tokenizer is set to false, which leaves the words read from the file in the original case as they appear in the file.

```
while (tokenIn.nextToken() != tokenIn.TT_EOF) {
   usr = tokenIn.sval;
   if (tokenIn.nextToken() != tokenIn.TT_EOF)
       usr_pwd.put(usr, new String(tokenIn.sval));
   }
```

Defines a while loop, which goes through the input file reading the user name/password pairs. The tokenIn.nextToken() call reads the next word from the input file. If the end-of-file is reached, the attribute tokenIn.TT_EOF is enabled. The loop continues until the TT_EOF is reached.

```
usr = tokenIn.sval;
```

Sets the variable usr equal to the value read in from the token, which is available in tokenIn.sval.

```
if (tokenIn.nextToken() != tokenIn.TT_EOF)
```

Reads the next word from the input file. The format of the file must be user name <space> password. Performs a check to TT_EOF to be sure a value was correctly read.

```
usr_pwd.put(usr, new String(tokenIn.sval));
```

This line adds the user name and password to the hashtable. The variable usr, read from line 95, is set as the key for the hashtable, and the just read password, available in tokenIn.sval, is set as the value for that key. The user_pwd.put(…) command makes the actual entry to the hashtable.

```
bufferin.close()
```

Closes the BufferedReader.

```
inFile.close()
```

Closes the FileInputStream.

```
Log.writeLine("*****Usernames read from external file: 
".concat(Integer.toString(usr_pwd.size())));
```

Writes a message to the OnDemand Server log file indicating how many user names were read from the file. This is just for informational purposes.

```
catch(IOException e){
```

Defines a catch for the error IOException. This is a generic error thrown when reading from the input file.

```
Log.writeLine("*****IOException Reading the password file");
```

Writes a message to the OnDemand Server log file indicating an error occurred reading the password file. More detailed error handling to identify the precise io problem could be added if problems are encountered.

```
public boolean avoidingGui()
```

The methods avoidingGui(), dontUseGui(), needsGui(), and okToUseGui() are required by the Java.Beans class to be defined in the code, but don't need to be filled out – they just need to exist.

Configuring the OnDemand Server

There are two steps to configuring the OnDemand Server to use the Custom method of authentication:

- Write the JavaBean as directed earlier in this appendix.
- Change the ODS.ini file by using either the Server Administrator or manually editing the file, and by making changes to the Windows NT Registry or the Unix OnDemand Server startup scripts.

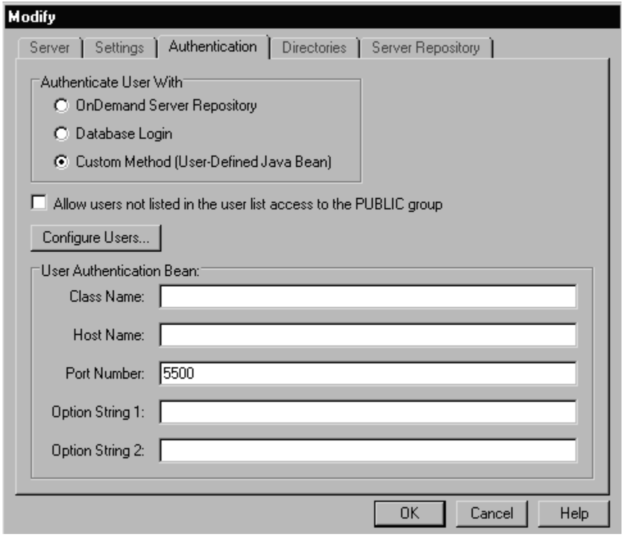
Server Administrator

Brio recommends that you use the Server Administrator to make changes to the ODS.ini file. The Server Administrator dialogs limits the Class Name and Host Name fields to 30 characters. When the Class or the Host Name entry is longer than 30 characters, you will need to make the entries manually to the ODS.ini file.

To use the Server Administrator to change the INI file for Custom Authentication:

1 Go to the Authentication tab of the Modify Server dialog (accessed by double-clicking the OnDemand Server name in the tree list) and select the **Custom Method** radio button in the Authenticate Users With group box.

The User Authentication Bean group box appears.



Authenticate User With
( ) OnDemand Server Repository
( ) Database Login
(●) Custom Method (User-Defined Java Bean)

[ ] Allow users not listed in the user list access to the PUBLIC group

User Authentication Bean:

Class Name:	
Host Name:	
Port Number:	5500
Option String 1:	
Option String 2:	

2 Fill in the required field, **Class Name**.

This is the name of the class file that contains your JavaBean, without the .class extension. Prepend the package name (the subdirectory your class file is placed in) to the class name.

In the example above, a file named `ODSLogin.class` contains the JavaBean, and is located in a directory named `ODSLogin` inside the …`/Server/Classes` directory, so the entry is the `package_name.class_name`, or `ODSLogin.ODSLogin`. See “JavaBean Class File Location” on page D-17 for more details.

3 Fill in the required field, **Host Name**. This is used to pass a character value to the JavaBean.

While the field is named Host Name, you can use this field to enter any value you wish.

One common function may be to connect to a service on another machine in the network to perform the authentication task. In this scenario, the host name of the machine to connect to would be needed by the JavaBean. For this reason, the field is named Host Name – although the receiving JavaBean can use the value for any purpose you choose. The value is simply passed to the JavaBean in an argument named Host Name.

An entry is required for this field – if you are not using it to pass an argument to the JavaBean, simply enter the name of the machine the OnDemand Server is running on.

4 Fill in the required field, **Port Number**. This is used to pass an integer value to the JavaBean.

The field is named Port Number because this field is commonly used to pass the JavaBean the TCP/IP port number on the remote machine.

This entry can be used to pass any integer value you wish to the JavaBean (the other entries all pass character strings).

An entry is required for this field – if you are not using it to pass an argument to the JavaBean, simply enter any integer number you choose (i.e. 0 or 1 work fine).

Option String 1 and Option String 2 are not used in OnDemand Server 6.2.x. They will be enabled in a future version.

ODS.ini File

The ODS.ini file holds the JavaBean configuration information entered on the Authentication tab of the Server Administrator. These configuration entries should be entered using the Server Administrator except when the Class Name or the Host Name entry is more than 30 characters. If the combination of package_name.class_name is longer than 30 characters, you will need to make the entry manually to the ODS.ini file, as the user interface will not allow you to enter longer strings. If you are using the Host Name field to pass a string longer than 30 characters, you must also manually edit the ODS.ini file.

In the sample JavaBean described earlier in this document, we are passing a path and file name to a file containing the list of user names and passwords – since this path is longer than 30 characters, this entry would have to be manually entered into the .ini file. If you do enter more than 30 characters in the .ini file, when you use the Server Administrator tool it will reset the value to a null entry and prompt you to enter a new value, which must be less than 30 characters again.

Due to this 30-character limit it is recommended you keep the name of your JavaBean package relatively short, so the combination of packagename.classname is 30 characters or less.

Sample ODS.ini

```
BQ_LANGUAGE=en
BQ_MAXIMUM_PROCESS=50
BQ_ADMIN_TABLE_OWNER=brio
BQ_USER_AUTH_MODE=3
BQ_USER_AUTH_CLASS_NAME=ODSLogin.ODSLogin
BQ_USER_AUTH_HOST_NAME=c:\Program Files\Brio\Brio Enterprise
Server\Server\pwd.txt
BQ_USER_AUTH_PORT_NUM=1
BQ_USER_AUTH_OPT_STR1=
BQ_USER_AUTH_OPT_STR2=
BQ_OCE_DIRECTORY=C:\Program Files\Brio\Brio Enterprise
Server\Program\Open Catalog Extensions
BQ_EXEC_PATH=C:\Program Files\Brio\Brio Enterprise
Server\Program\Brioqry.exe
BQ_ADMIN_USERNAME=system
BQ_ADMIN_OCE=odsrepository.oce
BQ_LOG_DIRECTORY=C:\Program Files\Brio\Brio Enterprise
Server\Server
BQ_KEEP_TEMP_FILES=false
BQ_PORT_NUM=5500
BQ_ADMIN_PASSWORD=_…e©Ür ;AV¾>VŸ"w#
BQ_PUBLIC_FOLDER_READ_PERMISSION=true
BQ_MAXIMUM_PROCESSING_MEMORY=1000
BQ_START_LOG=false
BQ_DOCUMENT_DIRECTORY=C:\Program Files\Brio\Brio Enterprise
Server\Program\Documents
BQ_TEMP_DIRECTORY=C:\Program Files\Brio\Brio Enterprise
Server\Server\Working
BQ_TOTAL_PRESPAWNED_PROCESS=0
```

Explanation of the code:

```
BQ_USER_AUTH_MODE=3
BQ_USER_AUTH_CLASS_NAME=ODSLogin.ODSLogin
BQ_USER_AUTH_HOST_NAME=c:\Program Files\Brio\Brio Enterprise
Server\Server\pwd.txt
BQ_USER_AUTH_PORT_NUM=1
BQ_USER_AUTH_OPT_STR1=
BQ_USER_AUTH_OPT_STR2=
```

The above lines comprise the JavaBean configuration information.

```
BQ_USER_AUTH_MODE=3
```

Instructs the OnDemand Server to use the custom JavaBean for authentication (0=OnDemand Server repository authentication; 1=Database login authentication; 3=JavaBean authentication).

```
BQ_USER_AUTH_CLASS_NAME=ODSLogin.ODSLogin
BQ_USER_AUTH_HOST_NAME=c:\Program Files\Brio\Brio Enterprise
Server\Server\pwd.txt
BQ_USER_AUTH_PORT_NUM=1
BQ_USER_AUTH_OPT_STR1=
BQ_USER_AUTH_OPT_STR2=
```

Maps directly to the six edit boxes on the Authentication tab of the Server Administrator.

JavaBean Class File Location

The JavaBean you create is compiled to a Java class file. This class file is made available to the OnDemand Server by placing it in a subdirectory inside the OnDemand Server's .../`Server/Classes` directory. It is important to name the directory the same name as the package name you have specified in your JavaBean.

In a nutshell, the JavaBean will have a statement defining the package name, and will be saved into a `.class` file named `your_package_name.class`. The directory you create inside the .../`server/classes` directory must be the same as this package name. In the sample file, we create a package named ODSLogin, which compiles to a file named `ODSLogin.class`. This class file should be placed in the subdirectory .../`server/classes/ODSLogin`.

Registry Changes (Windows NT)

The Windows NT registry must be manually modified to identify the directory containing the OnDemand Server Class files. Modify the registry to add the path to the `/server/classes` directory (without specifying the sub-directory your actual JavaBean is located in). This addition is made in the `classpath (-cp)` argument that passes, to the JRE, the name of the file and directory locations of the Java classes the application uses. The additions to make are to the `cmdline` and `ProcFactCmdLine` values under the registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Brio Technology\Brio Enterprise
Server\OnDemand Server
```

Add the bolded text below, which is a semi-colon after the `ODSClasses.jar` followed by the full directory path to the OnDemand Server classes directory.

```
Cmdline
C:\PROGRA~1\JavaSoft\JRE\1.1\bin\jre -cp
C:\PROGRA~1\Brio\BRIOEN~1\Server\Classes\ODSClasses.jar;C:\PROG
RA~1\Brio\BRIOEN~1\Server\Classes brio.server.ServerMain
"C:\Program Files\Brio\Brio Enterprise Server\Server" 1

ProcFactCmdLine
"C:\Program Files\Brio\Brio Enterprise
Server\Program\odsprocfac.exe" courier 5500
C:\PROGRA~1\JavaSoft\JRE\1.1\bin\jre -cp
C:\PROGRA~1\Brio\BRIOEN~1\Server\Classes\ODSClasses.jar;C:\PROG
RA~1\Brio\BRIOEN~1\Server\Classes "C:\Program Files\Brio\Brio
Enterprise Server\Server" 0
```

ODS.START Shell Script changes (UNIX)

The Unix script to start the OnDemand Server must be manually modified to identify the directory containing the OnDemand Server Class files. Modify the startup script to add the path to the /server/classes directory (without specifying the sub-directory your actual JavaBean is located in). This addition is made in the classpath (-cp) argument that passes, to the JRE, the name of the file and directory locations of the Java classes the application uses.

The additions to make are in the script file `ods.start`. Add the bolded text below, which is a colon (not a semi-colon like Windows NT uses) after the `ODSClasses.jar` followed by the full directory path to the OnDemand Server classes directory.

```
ods.start
#!/bin/sh
# This script is automatically generated during product install.
# This script is configured for Solaris.
if [ "$LD_LIBRARY_PATH" != "" ]
then
   LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/courier/brio/ods/lib
else
   LD_LIBRARY_PATH=/courier/brio/ods/lib
fi
if [ ${DISPLAY}. = . ] ;then
   echo DISPLAY variable needs to be set
   exit 1
fi
export LD_LIBRARY_PATH
BRIO_LDPATH=$LD_LIBRARY_PATH
export BRIO_LDPATH
UIDPATH=/courier/brio/ods/bin/%U; export UIDPATH
/courier/brio/ods/server/jre1.1.6/bin/jre -cp
/courier/brio/ods/server/classes/ODSClasses.jar:/courier/brio/o
ds/server/classes brio.server.ServerMain
/courier/brio/ods/server 1 &

sleep 5
exec /courier/brio/ods/server/odsprocfac courier 5500
/courier/brio/ods/server/jre1.1.6/bin/jre -cp
/courier/brio/ods/server/classes/ODSClasses.jar:/courier/brio/o
ds/server/classes /courier/brio/ods/server 0 &
```

Summary

This appendix presents a working sample JavaBean for the OnDemand server custom JavaBean authentication mode, and how to configure the OnDemand Server to use the JavaBean for authentication. By coding your own JavaBean, you will be able to integrate programmatically with your existing security solution.

It is hoped this sample program will be just a starting point for custom solutions developed by Brio customers and partners. As you implement solutions of your own, please share the ideas and, where possible, the actual Java code you have written with Brio and our other customers.

Glossary

Adaptive reporting 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.

Brio Intelligence Server A Server that works on behalf of the user to deliver reports either when scheduled or when requested. Its two main components are the Broadcast Server and the OnDemand Server.

Brio Intelligence Server Administrator A configuration and administration tool used to configure and administer both the Broadcast Server and the OnDemand Server.

Broadcast Server A server that distributes scheduled reports through email, the Web, FTP, internet file servers, or printers.

cluster A Manager and one or more nodes.

failover Restarting a request that failed.

heap size A common pool of free memory a program can use.

load balancing Routing requests across nodes.

OCE files Open Catalog Extension files contain all the information necessary to logon to a specific configuration of database and connection API software.

OnDemand Server A Web-based application that allows users to select documents from a list of available documents.

replication Copying BQY documents and OCE’s across nodes.

Index