CGI Advantage® 4

System Administration Guide

Table of Contents

Overview

The System Administration Guide is intended to help system administrators initiate, configure, monitor, and control various aspects of CGI Advantage. This guide contains information about the CGI Advantage system architecture, and configuration (including the embedded third party components), post-installation setup, and administering various aspects of the application. Additional guides (such as Workflow and Collaboration User Guide and System Processing User Guide) are available through the Advantage Administration online help and PDF versions of guides are available for download from the User Guide Downloads topic in each of the application's online help system. Refer to the **User Guides in PDF** link in the footer of the online help for each application to view a list of all System Administration Guides and Run Sheet Guides that are available for that application.

Refer to the following topics for additional information:

- Overview of System Architecture
- Overview of Unified UI Concept
- Overview of Third Party Tools

Overview of System Architecture

CGI Advantage has a standards-based open architecture, which employs Java Platform, Enterprise Edition (JEE) as its foundation. It has an n-tier architecture, which is flexible and scalable for various loads and needs.

There are many possible configurations of these logical tiers. These logical tiers do not necessarily imply separate physical servers. For example, it is technically possible to host the application server and database servers (and even the Web server) on the same physical server, although it may not be preferred in the production environment due to various operational, procedural, performance, and security reasons. The application is scalable vertically (more than one application server on the same physical server, to leverage the available CPU and memory) as well as horizontally (application servers spread across multiple physical servers), which provides tremendous flexibility in handling any load. Furthermore, for each tier, there are various deployment options in terms of hardware and Operating Systems platforms. The following topics describe some of these deployment options.

Client Tier

CGI Advantage uses a thin HTML client that requires only a Web browser. A component based rich user interface (UI) makes the application more intuitive to the end users, as it is consistent with basic Web application navigation and usage paradigms. It also eliminates the software distribution issues associated with the traditional client server applications and reduces the hardware requirements on the client tier. These factors combine to make the application more accessible to all users.

Web Server Tier

The HTTP requests originating from end users' browsers are handled first by the Web server(s). This traffic between the client and Web server may be encrypted and secured through the use of Secure Sockets Layer (SSL). While the Web server "understands" the HTTP and HTTPS protocols, it does not contain any application knowledge. Rather, it forwards the request to the

servlet pointed to by the URL. The servlet, in turn, communicates to the application server, which contains the application business rules. The servlet can be configured in many ways: it can reside on the same machine as the Web server, or alternatively it can reside on the application server machine.

Application Logic Tier

The application server tier contains the core functionality of the application. IBM's WebSphere / Red Hat's JBoss provides the "container" services for a J2EE based architecture of CGI Advantage. WebSphere /JBoss is responsible for application services such as failover, connection pooling, thread pooling, session management and others, which provide the foundation for a transactional application like CGI Advantage. The application-level services can be broadly classified into the following logical sub-tiers:

- The Business Logic Server (BLS) contains all the business logic associated with the CGI Advantage objects and processes. It consists of Java classes and Enterprise Java Beans (EJBs) generated from the business rules specified in the design studio, custom Java code, and all the supporting class libraries.
- The Presentation Logic Server (PLS) is responsible for providing the presentation services required for client framework. Just like the business logic server, this is composed of the generated Java classes, EJBs, custom code, and supporting class libraries.
- The client framework is responsible for interacting with the PLS. It constructs and renders the user interface and controls its behavior.
- The Connectors provide the interface to the data and help to keep the business logic independent of the data access mechanism. In other words, connectors ensure that how the data is accessed is independent of the business logic. Although the connectors are most commonly used to interface with Relational Database Management Systems (RDBMS), they can also be used to interface CGI Advantage with external systems or other data stores.

Depending on the load on the system, multiple logical application servers can be configured to run on one or more physical servers. For example, it may be possible to run four "copies" of the application server on two physical servers. The "copies" are capable of load balancing and failover.

Database Tier

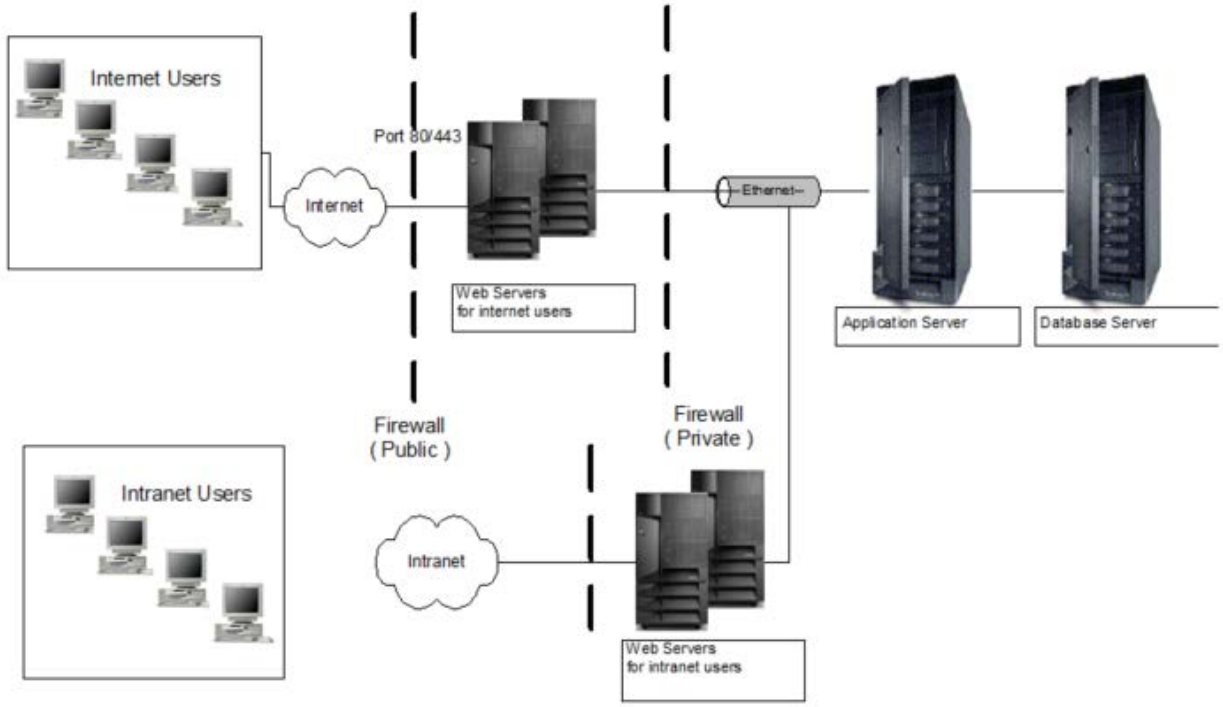
The database server hosts an RDBMS. The application server retrieves data from and updates data in the database through the connectors, which utilize the JDBC standard.

Please refer to the *CGI Advantage Container Deployment Guide* for the complete list of certified platforms (hardware, operating system, and other system software) for the various tiers.

N-tier Deployment Environment

While planning for a production environment, additional configuration issues become very important. For example, in a Web-based application, firewalls may need to be planned for in order to have a more secure environment. Although every implementation of CGI Advantage is different and it is not possible to recommend a "universal" production environment, the following figure

depicts a sample production configuration to illustrate some of the issues related to Web-based application deployment. It shows a possible need for different configurations for internet and intranet users.



Overview of Unified UI Concept

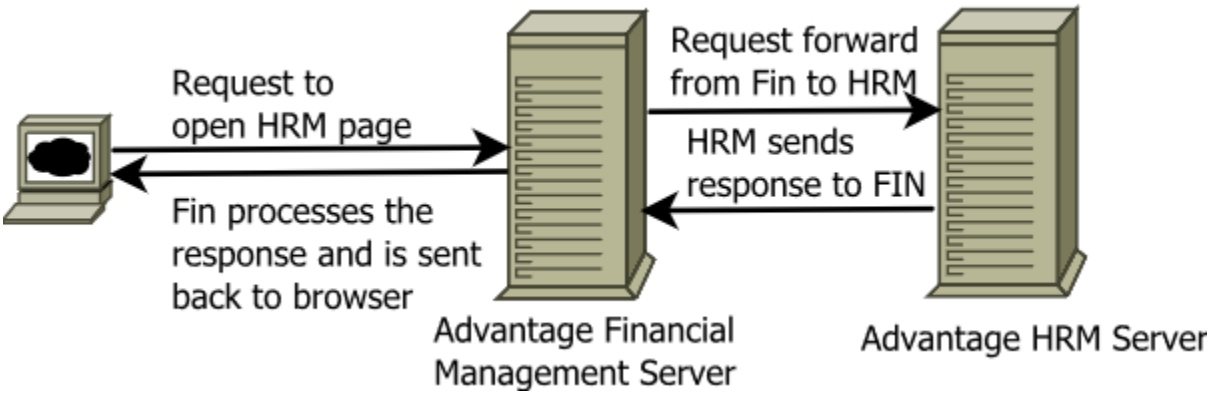
The Advantage ERP is a suite of different products such as, Financial Management, Human Resource Management, and Performance Budgeting. These are typically independent products although since they belong to the same family internally they are integrated. Users can typically access each application through its own end point URL. There can be a scenario where a site wants to allow a user to open a page from a different application from within the current application. Unified User Interface helps users to navigate across application pages without logging in to the other Advantage application through a separate browser window. In other words, a User can launch Advantage Financial and do his/her daily routine task. For one task (say submitting a leave application) he/she can navigate to the Advantage HRM page directly in the same window without launching the HRM application in a separate window.

Refer to the following topics for more information:

- Environment Setup
- User Setup
- Limitations

Environment Setup

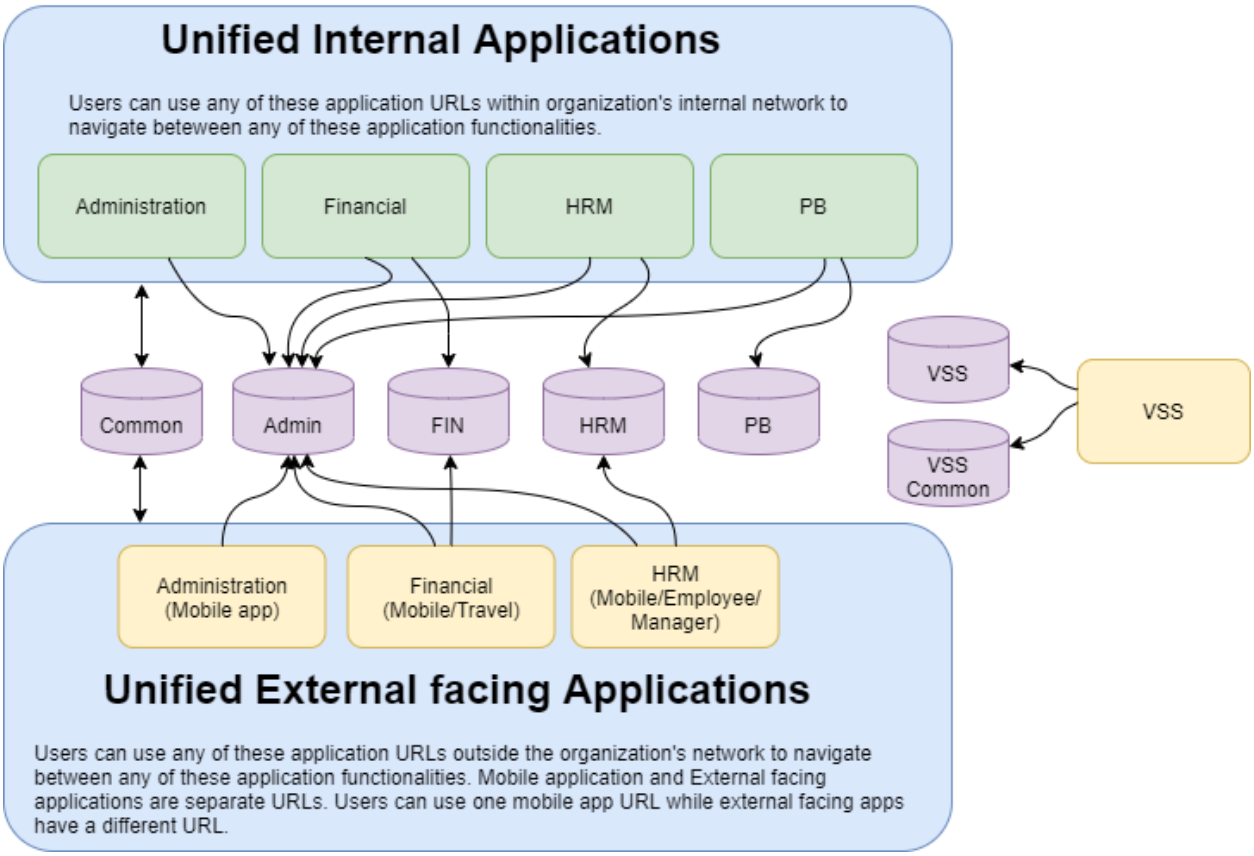
Unified UI internally relies on inter server communication. Therefore, if we consider the scenario of opening a HRM page from the Financial application, the request to open the HRM page flows through the Financial Management server and it internally processes the request and submits it to the HRM server. The HRM server sends the response back to the Financial server where the response is processed and sent back to the browser.



For unified UI to work seamlessly, all applications hosted on different servers or ports should be accessible to each other. This is the basic requirement needed. Additionally, the application URLs need to be configured in the configuration file placed in <VLS_HOME>/VLS/bin/server_bootstrap.properties.

Overall, the goal of unified UI is to allow users to seamlessly navigate between applications, although which application URL the user should use, should be based on the number of users registered on a particular application. In addition, there is no true single URL as the external facing applications such as Employee/ Manager/ Travel and Mobile have restricted functionalities. These applications serve only a subset of applications where pages are whitelisted. Please refer to the “Whitelisting of Application Metadata” section in this guide.

The following diagram explains the high-level deployment structure with respect to the usage of URLs for application access.



User Setup

The Advantage Administration application is responsible for Application security. Users are defined and maintained in the Administration application. With Unified UI, any typical application page can be opened and operated from another Advantage application. The following setup is required to allow the user to access pages in each authorized Advantage application:

1. Users need to be defined/created in the Administration application of each Application URL from where the cross application page navigation is expected. The Administrator can use different ways to create a user as mentioned in the “User Setup and Maintenance” topic in the *CGI Advantage Security Administration Guide*. All applications that are involved in cross application page navigation should have the same User Id defined across all Administration applications.

2. The defined user needs to have application authorization for any of the primary applications (that is, Administration, Financial, HRM or Performance Budgeting) to be able to login to the application URLs provided. The VSS application is independent and users need specific application access and can only login to the associated VSS application URL. The VSS application is not unified with other applications.

3. The defined user needs to have relevant access control authorization to the page and functionality that he/she is trying to access. If the user has access to any of the primary

applications, (that is, Administration, Financial, HRM or Performance Budgeting), then the user can use any URL to login to the application. However, the user can only open pages where necessary security authorization is granted. Refer to the “User Information” topic in the *CGI Advantage Security Administration Guide* for more information.

4. The user needs to have the proper Security Roles assigned along with the action authorization on the Access Control page. Refer to the “Security Role” and "Access Control” topics in the C*GI Advantage Security Administration Guide* for information.

For sites that have a shared Administration Application across Advantage Financial and HRM, since the Administration Application/DB is shared across other applications, the user needs to be defined/created only once. The security authorization setup is also defined at a common place.

For sites that have a separate Administration application for Advantage Financial and HRM, Steps 1 and 2 above needs to be in sync with all Administration applications where the User want to achieve cross application page navigation. Step 3 can be done in the specific Administration application and does not need to be in sync. Such authorization will always be performed by the application that is serving the page or action.

Limitations

As of the Advantage 4 release, if the separate or split Administration approach (that is, the Financial Management and HRM applications have their own Administration setup) is implemented for a site, then the Unified UI will not be able to provide the following features:

- Automatic synching of User setup as defined in the previous section is not permitted. The user or Administrator will have to manually keep the setup in sync.
- A user won’t be able to navigate to a page in Financial Administration from the HRM application; however, a user will be able to open a Financial Administration page from the Financial application. The same applies to a page in the HRM Administration application. That page can only be accessed from the Administration application or from HRM and cannot be accessed from Financial.

Overview of Third-Party Tools

This topic provides a brief description of each of the third-party tools used in conjunction with CGI Advantage. Please refer to later sections of this guide as well as the *CGI Advantage Developer’s Guide* for more details.

Versata

Versata is an enterprise application development tool, which provides a powerful business rules paradigm of application development, on top of a J2EE architecture. The business logic can be defined by using the point and click business rules capabilities of the Design Studio, which generates Java code to implement the rules. Additionally, the user interface and the database definitions can be generated from within the studio, thus providing a single development environment to cover all aspects of n-tier application development.

WebSphere

IBM WebSphere® Application Server is the premier Java™ technology-based application server, integrating enterprise data and transactions with the e-business world. It provides a rich, e-business application deployment environment with a complete set of application services including capabilities for transaction management, security, clustering, performance, availability, connectivity, and scalability. It provides the J2EE “container services” (as mentioned above) for the CGI Advantage application.

JBoss

JBoss® Application Server is the premier Java™ technology-based application server, integrating enterprise data and transactions with the e-business world. It provides a rich, e-business application deployment environment with a complete set of application services including capabilities for transaction management, security, clustering, performance, availability, connectivity, and scalability. It provides the J2EE “container services” (as mentioned above) for the CGI Advantage application.

BIRT Server

BIRT is a tool that allows for the creation and printing of forms. It has two main components: a visual designer for creating BIRT forms and a runtime engine for generating and printing forms. Eclipse IDE with BIRT design plugin provides a user friendly GUI for forms designing. The printing server is responsible for generating the pdf files, printing, emailing etc., based on the data file provide. It allows the creation of custom forms for the application.

Business Objects

Business Objects is a tool for query, reporting, and analysis of the information stored within the CGI Advantage implementation. It uses everyday business terms to describe functions, which simplifies report creation. It has easy-to-use yet powerful analysis tools, which enable you to combine information from several sources and build reports in which you can drill down for more detail. You can create on-demand reports easily and vary the format of finished reports based on content.

RightFax

RightFax is a tool that has been integrated with the online printing architecture within the CGI Advantage system. This tool provides the user a way to send out the generated PDF from the BIRT Server to a Fax number.

The user can "print" a transaction using a "virtual printer" which, instead of producing a paper printout, sends the transaction to the fax server, which then transmits it. In addition to this, an email notification is sent to the user intimating whether the Fax has been sent successfully or not. This tool provides many advantages to include:

- Users can send faxes without leaving their desks.
- Any printable computer file can be faxed, without having to first print the transaction on paper.

- The number of fax lines in an organization can be reduced, as the server can queue large numbers of faxes and send each of them when any of a number of available lines is free.

Common Terms and Glossary Used

The pages in CGI Advantage reflect the layers of the application. This guide uses some terms interchangeably to refer to certain concepts. These terms are as follows:

- The terms Versata Logic Server (VLS) and Business Logic Server (BLS) have been used interchangeably in this transaction to represent the logic services layer of the application.
- The terms Versata Presentation Server (VPS) And Presentation Logic Server (PLS) have been used interchangeably to refer to the presentation layer of the application.
- The terms "Job" and "Batch" have been used interchangeably throughout the transaction. Please note that the CGI Advantage technical architecture is flexible enough to support the execution of jobs/batch processes while the application is available for online usage. In other words, the jobs/batch processes are technically not required to be "offline" processes.

The following naming conventions have been used throughout this guide:

- <ADV_HOME>: CGI Advantage root installation directory (for example, C:\AMSADV3)
- <VERSATA_HOME>: Versata root installation directory (for example, C:\AMSAPPS\Versata564)
- <HTTP_HOME>: Web Server root installation directory (for example, IBM WebSphere C:\AMSAPPS\WebSphere61\IBMHTTPServer and JBoss: C:\CGIAPPS\EAP-6.4.0\JBoss-ews-2.1)
- <JBOSS_HOME>: JBoss root installation directory (for example, C\CGIAPPS\EAP-6.4.0\JBoss-eap-6.4
- <WAS_HOME>: IBM WebSphere root installation directory (for example, C:\AMSAPPS\WebSphere85\AppServer)
- <BUS_OBJ_HOME>: Business Objects root installation directory (for example, C:\AMSAPPS\Business Objects)
- <BIRT_HOME>: BIRT Server root installation directory (for example, C:\CGIAPPS\AdvFormsModule)
- <APPSERVER_NAME>: IBM WebSphere/JBoss Application Server Name (e.g. aa310ow1)
- <Advantage_APP_NAME>: CGI Advantage Application name specified in WebSphere/JBoss Application Server Admin console (for example, Advantage_aa310ow1)
- Configuration folder: A configuration folder is maintained for each product separately.

  /apps/4100CGIAdvantageGA/AdvantageAdmin/Configuration

  Where “4100CGIAdvantageGA” is the project folder and “AdvantageAdmin” is the product folder.
- <app>: This is the short name for application, for example, admin, fin, vss, hrm, or pb.
- <env>: This is the environment name for deployment.

Application Server Administration

This chapter includes information on the following topics:

VLS Console

The Versata Logic Server (VLS) Console provides enterprise-wide management of CGI Advantage components such as application business objects, user roles, and system-wide parameters.

The VLS Console allows you to perform administrative functions such as:

- **System Administration**: Configuring the VLS and managing deployed business objects.
- **System Tuning**: Setting server properties for data access, and database connection pooling parameters.
- **Performance Monitoring**: Monitoring performance, managing sessions, managing server operations, tracing, and logging.

Although the VLS Console provides many parameters to control the behavior of the application, the following topics discuss only a subset of those parameters, which are of particular significance. Refer to Versata Help or the *Versata Administrator Guide* for more details about the other parameters.

VLS Properties

Node level properties are defined inside EJBAppServerConfig.txt file under /RTFiles/<app>/Custom/Configuration/AppConfig/VLSEnv/<app>/VLSComponents/ directory.

You may need to create a copy of EJBAppServerConfig.txt from the RTFiles/<app>/Custom/Configuration/DefaultFiles/VLSEnv/<app>/VLSComponents location if it is not present at the AppConfig path.Add EJBAppServerConfig.txt to the external config folder if other parameters are expected to be changed.

For session_timeout and max_session, we use SESSION_TIMEOUT and MAX_SESSION inside values.yml, you can change in there and then just re-deploy application.

Refer to the *CGI Advantage Container Deployment Guide* for additional details on these VLS properties.

Property	Description	Value	Comments
DEBUG_INTERVAL	Number of seconds to wait before VLS prints out connection debug information.	Recommended setting is a very high value, for example, 9999999. Printing connection debug information is not expected	A low value can possibly be detrimental to performance as debug information has to be written to the VLS log at frequent intervals.

		in a production environment.	
SessionTimeout	Number of seconds to wait after the last ping from a client application before the session is considered terminated and is closed by the server.	Recommended setting is 300 to 900 (5 to 15 minutes). The aim is to balance freeing resources and not force users to login frequently after short periods of inactivity. This number should be at least three times greater than the session ping rate.	Default is 300 seconds.
SessionMonitorRate	Number of seconds to wait to check if any expired sessions can be cleaned up.	Should be set equal to or slightly higher than the SessionTimeout value.	
CoLocated	Flag that Indicates whether PLS is running in the same VM as VLS.	Set to true (PLS and VLS on same VM).	
SessionPingRate	Number of seconds to delay between pings by the session to check if the application is still running.	Recommended setting is a value less than 1/3 of SessionTimeOut.	
STATEMENT_TIMEOUT	Number of seconds after which the prepared statement expires.	Recommended setting is 1800 seconds.	Default is 1800 seconds.
VLS_SecurityManager	The value set to specify the custom security manager instead of Versata's default security manager.	Must be set to: com.amsinc.gems.adv.server.basecode.security.AMSSecurityManager.	Default is versata.vls.VSDefaultSecurityManager.

Data Server Connection Properties

Data servers provide named connections (or pointers) that bind a connector to its data source. Data servers map the client sessions to the database connection that handles and binds the data control to the source/object. The system uses data servers to group business objects. Each data server has its own login to the database. Therefore, the business objects in one data server can access different data than those deployed to another data server while using the same JDBC connectivity between the VLS and the data source.

CGI Advantage requires at least two data servers to be set up. One data server is used for connection pooling and prepared SQL statements; no data objects are assigned to it. The second (or any other) data server has the actual data objects assigned to it. The CGI Advantage installation process automatically creates all the required data servers. The following topics provide a brief description of data server concepts.

- Why Connection Routing?
- ConnectionRouter Data Server Connection Properties
- Application Data Server Connection Properties

Why Connection Routing?

CGI Advantage routes the most complex transactions in the application, such as validate and submit actions on transactions, to a separate sub pool of connections for which a separate data server is defined. This concept, known as connection routing, enables CGI Advantage to use and cache prepared SQL statements on a smaller pool of connections and thus helps balance the performance benefits of prepared statements against the memory footprint of caching these statements. Without the connection routing, the prepared statements would be cached in all connections of the connection pool and therefore need more memory; whereas this feature ensures that they are cached for the connections in the special sub pool only.

ConnectionRouter Data Server Connection Properties

No data objects need to be assigned to the CGI Advantage ConnectionRouter Data Server. Select, Insert, Update and Delete privileges to all the CGI Advantage tables must be granted to the database schema assigned to the CGI Advantage ConnectionRouter Data Server.

For example, the following SQL commands can be used to create a set of SQL statements that grant the database privileges on the Advantage database to the CGI Advantage ConnectionRouter schema in an Oracle environment. First, open SQLPlus and log in to the CGI Advantage Schema and then execute the following SQL commands (with appropriate modifications):

```
Set heading off;

Set feedback off;

spool d:\temp\grants.sql;

Select 'GRANT SELECT, INSERT, UPDATE, DELETE  ON '
||   table_name  ||  ' to <Connection Routing User>  ; ' from user_tables;

Spool off
```



```
@d:\temp\grants.sql;

commit;
```

Execute the SQL created after executing the above script (should be in d:\temp\grants.sql unless the spooled output file was modified) in the CGI Advantage Schema. If you are using SQL Server, execute the equivalent steps to grant the rights.

Application Data Server Connection Properties

The System Administrator can modify the business object properties for all data objects or for a specific data object. It is not recommended to modify the business object properties at the global level as CGI Advantage defaults have been set for these properties. Modify the value of these properties only at a specific data object level, if required. Any modifications to these properties require the application server to be recycled before taking effect. The CGI Advantage installation package automatically configures the property settings for all the existing business objects; however, if you need to modify this, you need to update RTFiles/<app>/Custom/Configuration/AppConfig/dev/advantage/Source/ExtendedObjects.properties. The application must be redeployed to generate the new set of vls console XML files.

You may need to create a copy of ExtendedObjects.properties from RTFiles/<app>/Custom/Configuration/DefaultFiles/dev/advantage/Source location if it is not present at the AppConfig path.

Property	Description	Value	Comments
Database	Database Instance Identifier for SQL Server.	The CGI Advantage database instance identifier for SQL Server.	
Data Server Name	Name of the data server.	Recommended setting is Advantage.	This value can also be configured to match the schema name.
Data Server Type	How the data server connects to the VLS	Select Oracle-Jdbc Thin Client for Oracle. Select MSFT-MSSQL for SQL Server.	Each default data server has its own set of properties that define how the data server connects with its Versata data connector. New data server types can also be defined. Refer to the *Versata Administrator Guide* for details.

Data Server Login	Login used to connect to the data source.	The CGI Advantage database user/schema name.	
Data Server Password	Password for the data server login data source.	The CGI Advantage database user's password.	
Connection Idle Timeout	Maximum time an idle resource can remain in pool before being removed.	Recommended setting is *1800 seconds*.	Default is *1800 seconds*.
Connection Timeout	Maximum request wait time when all connections are in use.	Recommended setting is *100 seconds*.	Default is *1 secon*d.
Host	Database server host name.	The CGI Advantage database host name.	
In_Tran_Isolation_Level	Transaction isolation level used for a connection while it is in a VLS transaction	Set to *2*.	Defaults are set based on the type of database.
Lock Deferred	Stage of the transaction at which locks are applied.	Set to *true* (Locks applied only when data object is saved to ensure that the data is more available)	Default is false.
Locking Hint	Literal used for database locking	FOR UPDATE WITH RS for Oracle WITH (UPDLOCK) for SQL Server	Defaults are set based on the type of database.
MaxCachedStatements	Maximum number of statements cached in memory	Recommended setting is *1000*.	Should be configured based on: • Number of unique SQLs in processing • Memory allocated to Versata

			Application Server/Copy
Optimistic Lock	Flag to specify whether VLS should do optimistic locking.	Set to *true* (Overrides Lock Deferred and Locking Hint properties; the VLS does not apply any locks, nor does it do any synchronization at save. All locks and synchronization is done by CGI Advantage custom processing).	
Out_Tran_Isolation_Level	Transaction isolation level used for a connection while it is not in a VLS transaction.	Set to *2*.	Defaults are set based on the type of database.
Port	Database Port Number	The CGI Advantage database port number.	
Schema	Schema Name	The CGI Advantage database schema name.	
SID	Database Instance Identifier for Oracle	The CGI Advantage database instance identifier for Oracle.	
SQL Allowed	Flag to indicate if manually coded SQL statements (not Versata generated SQLs) can be executed by data server.	Must be set to *true*.	Default is *false*.
Use Batch Update	Flag to enable batch update of SQLs, which can reduce network traffic.	Must be set to *false*. This property should not be set to *true* as the order of SQL execution is important for CGI Advantage.	Default is *false*.
Use Prepared Update	Flag to indicate if prepared statements	Set to *false*.	

	should be used for updates when executed by the data server.		
Use Prepared Delete	Flag to indicate if prepared statements should be used for deletes when executed by the data server	Set to *false*.	
Use Prepared Insert	Flag to indicate if prepared statements should be used for inserts when executed by the data server.	Set to *false*.	
Use Prepared Select	Flag to indicate if prepared statements should be used for selects when executed by the data server.	Set to *false*.	

Business Object Properties

The System Administrator can modify business object properties for all data objects or for a specific data object. It is not recommended that business object properties be modified at the global level as CGI Advantage defaults have been set for these properties. Modify the value of these properties only at a specific data object level if required. Any modifications to these properties require the application server to be recycled before taking effect. The CGI Advantage installation package automatically configures property settings for all existing business objects; however, if you need to modify this, then you can update **Custom/Configuration/VLSConsole/Input/ExtendedObjects.properties**. The application needs to be redeployed to generate a new set of vls console XML files.

User Roles Properties

The CGI Advantage installation package automatically configures user roles for all the existing business objects and client applications. However, if any new business objects/client applications are added to CGI Advantage, those new components must be added through data source mapping files available at RTFiles/<app>/Custom/Configuration/AppConfig/dev/advantage/Source/*_DataSourceMapping.xml and advantage.xml files.

You may need to create copy of *_DataSourceMapping.xml and advantage.xml from the RTFiles/<app>/Custom/Configuration/DefaultFiles/dev/advantage/Source location if it is not present at the AppConfig path.

Locking Application Pages for Users

During a nightly cycle or any specific job execution, there is a need to lock certain application functionality. Some jobs read tables modified by the processing of transactions and some read tables for processing instructions or other input. If these types of updates are made during job execution, it may result in job failure or unwanted outcomes from processing.

Typically, administrators stop users from accessing applications by bringing down the application to prevent online access. With a unified UI, if access to a specific URL is prohibited, then it could prevent all users from accessing all of Advantage. For instance, if users are using the HRM URL to sign in to the application and they want to run a process that requires the online HRM application to be locked, then all Administration, Financial, and Performance Budgeting users are locked out of the application. This chapter explains a way to lock a specific application without locking users from accessing other applications through a unified URL.

To lock a specific application, an administrator with database schema access, updates the IN_SYS_ST table. This table always has a single row that maintains the system state. Each application schema has its own version of this table. The SYS_ST column value decides the state of the system. The CVL_SYS_ST table defines all the possible values of SYS_ST.

The following table provides the description for all possible values:

SYS_ST	ST_DESC	Description and Usage
1	Down	With this value the application will be truly locked and no user will be able to login to the application.
2	Up	Default value. With this value the application will be up and running. Users can login and perform necessary tasks.
3	Offline Only	Users are not able to login to the application; however, the jobs can be scheduled and executed.
4	Offline Only	Jobs cannot be scheduled; however, users can login to the application.
5	Offline Only For Current Application	Apart from offline access, in unified UI setup, users are allowed to login to the application using an application URL. However, users are not allowed to open any page that belongs to the application URL. For example: Users logging in to the HRM internal or external URL can still login to the application and can open any page from FIN/Admin/PB; however, the user will not be able to open HRM application pages.

6	Offline And External Facing Only For Current Application	Apart from offline access, in unified UI and external facing application setup, users are allowed to login to the application using an application URL. However, users are not able to open any page that belongs to the application URL. For example: User logging in to the HRM internal URL, can still login to the application and can open any page from FIN/Admin/PB; however, the user cannot open any HRM application pages. If the user is using an external facing HRM application URL, he/she can continue to use the application normally.
7	Offline And Internal Facing Only For Current Application	Apart from offline access, in unified UI and internal facing application setup, users are allowed to login and use the internal application normally. However, while using the external application URL, users are allowed to login, but are not allowed to open any page. For example: User logging in to an external HRM URL can still login to the application and can open any page from FIN/Admin/PB; however, the user cannot open any HRM application pages. If the user is using an internal facing HRM application URL, he/she can continue to use the application normally.

Whitelisting of Application Metadata

To solve the problem of restricting access to Employee and Manager users, the full HRM application is deployed with truncated application metadata. The application.xml stores the list of all pages whose metadata should be present in the metadata database (pal db). The page entry for each external facing application has an additional property with the roles to which these pages are tied to. The key value generator has a new property "externalApplication" to generate metadata for that external application.

For Example:

If a site needs Employee and Manager related pages to be external facing, then the externalApplication property should be set with the following value:

externalApplication=employeeManager

If a site needs Travel and Procurement related pages to be external facing, then the externalApplication property should be set with the following value:

externalApplication=travelExpense,procurementPortal

If a site needs just Procurement related pages to be external facing, then the externalApplication property should be set with the following value:

externalApplication=procurementPortal

Currently, when the user logs in to the system, the user is presented with a list of business roles that the user belongs to. These roles can be seen in the Role drop-down list in the Account Profile page.

In the Business Role definition page, every role has a "Allowed in DMZ" flag. This flag is used to filter out the roles that should be seen in the Account Profile page. If the server config property "deployedTopology" ="DMZ" then all roles are filtered based on this flag.

The users can view the roles available for them in the Account Profile page. The users can view the roles only for which the Allowed in DMZ flag is set to true (Allowed in DMZ= "true").

The Enterprise search bar is always hidden for external facing applications.

The corresponding URL configuration of other applications should be removed from the server_bootstrap.properties file. This restricts the users from opening any cross application pages, if any in the unified UI scenario.

Advantage Business Integration

Refer to the following topics for information on connecting CGI Advantage to Advantage Business Integration and for information on ABI setup within Advantage.

- Connect CGI Advantage to Advantage Business Integration
- Advantage Permitting Event Detail
- BrassRing Event Detail
- Common Event Log
- Dashboard
- Event Mapping
- Financial Event Detail
- Flow Level Logs
- HRM Event Detail
- infoAdvantage Event Help
- Integration Global Parameters
- Integration Resource
- Integration Server List
- Integration Service Parameters
- Integration Service Status Log
- Maximo Event Detail
- Meridian Global Event Detail
- NEOGOV Event Detail
- Notification Destination
- Notification Profile
- Notification Provider
- Related Events
- Schedule Flow
- Service Catalog
- Service Definition
- Service Flows

- Service Groups
- Service Pool
- Service Type
- Synchronous Event Detail
- Transaction Integration Log
- Transaction Integration Messages

Connect CGI Advantage to Advantage Business Integration

This section describes the configuration setup required to connect CGI Advantage to Advantage Business Integration (ABI). Changes are required in the CGI Advantage initialization parameter file (ADV30Params.ini).

1. IntegrationManagerURL is a parameter in the ADV30Params.ini file. This parameter's value determines the Web Service URL to connect CGI Advantage to the Advantage Business Integration (ABI) servers associated with the Integration Manager UI application. These are the ABI servers that host the ABI services controlled by Integration Manager.

   ##

   ##

   ## Integration Manager WebService connection URL.

   ## This URL will be used to schedule/run service flows on ABI.

   ##

   ##

   IntegrationManagerURL=http://<hostname>/<AdvIMEngineWeb>/IntegrationManager/

   - hostname is the host on which ABI server is running.
   - AdvIMEngineWeb is the Web Context Root of the Web module - ProcessFlowManagerRS of Integration Manager application.

2. UseIntegrationEnabled is a parameter in the ADV30Params.ini file. This parameter's value determines if ABI integration is enabled or not.

   ##

   ## Configure Advantage to Integration Manager

   ##

   ## Variable Possible Values Comment

   ## -------- --------- ----------

1.UseIntegrationEnabled true/false This property indicates if IntegrationManager is Enabled.

##

###

UseIntegrationEnabled=true

ABI CSOD Event Detail

The ABI CSOD Event Detail (ABICDEVE) page displays ABI events and related errors, if any, that are originating from/destined for the CSOD application. This page provides basic search functionality. The **View Source** link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the status field and selecting **Save**.

Advantage Permitting Event Detail

The Advantage Permitting Event Detail (ABIPEEVE) page stores the Advantage Business Integration (ABI) events and related errors, if any. By looking at the **Direction** field you can see whether the event originated in or is destined for the Advantage Permitting application.

If the event is originating in the Advantage Permitting application, the **Direction** is *Outbound.* Outbound events cannot be changed but they can be duplicated for reprocessing when they are in *Error* status. This is done by changing the **Status** to *Ready* and clicking on the **Save** link. This is the only update possible on this page. Status is the only unprotected data field.

If the event originated in ABI and is destined for the Advantage Permitting application the **Direction** is *Inbound*. Unlike outbound events, inbound events cannot be directly reprocessed. Instead, a **View Source** link is active for inbound events. You can use this link to navigate to the outbound event that caused ABI to generate the inbound event. This outbound event will be presented on the Event Detail for the ABI partner application that generated that event (for example, Advantage Financial Event Details). If the outbound event is in *Error* Status, reprocessing can then be initiated by changing the **Status** of the outbound event to *Ready.*

The Related Event link navigates you to the Related Events page, which shows the related events for the current event. For outbound events, the related events are:

- Any reprocessing events generated by changing the Status of the current event.
- Any inbound events generated by ABI while processing or reprocessing this event.

For inbound events, the related events are:

- The outbound event that caused ABI to generate the current event.
- Any other inbound events generated by ABI while processing that outbound event.

The **Service Flow** link navigates you to the Service Flow page, which will show the service flows that ABI used to process the current event. From that page the **Logs** link can be used to review the details of all processing for the current even.

BrassRing Event Detail

The ABI BrassRing Event Detail (AIBREVE) page displays ABI events and related errors, if any, that are originating from/destined for the BrassRing application. This page provides basic search functionality. The **View Source** link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the status field and selecting **Save**.

Common Event Log

The ABI Common Event Log (ABICMEVE) page displays ABI logs if logging is configured with the database. This page provides basic search and sort functionality.

Dashboard

This page provides the ABI System Administrator user a graphical user interface representation having an at-a-glance view of Key Performance Indicators related to ABI events processed/services and system health. The Dashboard identifies whether anything is going wrong in the system and enables the System Administrator to make instantaneous and informed decisions.

It contains the following sections showing real-time graphical data captured from the ABI application.

1. Application statistics - This section contains the following real-time statistics:
    - Number of Integrations Configured – This provides a count of the number of ABI services configured. The hyperlink takes the user to the Service Catalog page.
    - Service flows running – This provides a count of the number of services currently being processed by ABI. The hyperlink takes the user to the Service Flows page to show services filtered by state as Running.
    - Services failed – This provides a count of the number of services that Failed. The hyperlink takes the user to the Service Flows page to show services filtered by state as Failed.
    - Events processed – This provides a count of the number of events processed by Direction. The hyperlink takes the user to the Related Events page.
    - Documents integrated – This provides a count of the number of documents integrated to Advantage. The hyperlink takes the user to the Document Integration Log page.
- Application alerts - This grid shows the list of alerts with an option to delete those raised by the ABI application under the following situations:
    - Failed service flow
    - Rejected event
    - Unprocessed events exceeding a threshold lying in Ready state
2. Services:
    - Graph 1: The Bar graph shows the total number of services processed since the last captured server time.

    - Graph 2: The line graph shows the total number of services processed per unit time per ABI server.

- Transactions:
    - Graph 1: The Pie Chart shows a breakdown of transactions integrated by Transaction Code with selection by Department.
    - Graph 2: The Pie Chart shows a breakdown of transactions integrated by Transaction Integration Status with selection by Department.
- Events:
    - Graph 1: The Pie Chart shows a breakdown of events by Event Type with a selection by Partner.
    - Graph 2: The Bar chart shows the number of events (Inbound/Outbound) for an Event status with selection by Partner. Events that are captured belong to Ready for processing, Completed with Error, and Success.

Event Mapping

The Event Mapping page allows you to assign the poller and outbound event service to be used to process each ABI Event Type.

Financial Event Detail

The ABI Financial Event Detail (ABIFINEVE) page displays ABI events and related errors, if any, that are originating from/ destined for the Financial application. This page provides basic search functionality. The **View Source** link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the status field and selecting **Save.**

Flow Level Logs

The Flow Level Logs page allows you to view the detailed component by component log produced by the service flow.

HRM Event Detail

The ABI HRM Event Detail (ABIHREVE) page displays ABI events and related errors, if any, that are originating from/ destined for the HRM application. This page provides basic search functionality. The **View Source** link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the status field and selecting **Save.**

infoAdvantage Event Help

The ABI infoAdvantage Event Detail (ABIINEVE) page displays ABI events and related errors, if any, that are from or destined for the infoAdvantage application. This page provides basic search functionality. The **View Source** link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the Status field and selecting **Save.**

Integration Global Parameters

The Integration Global Parameters page allows you to view and maintain Global Parameters. These Global Parameters are available for use in service definition parameters. The *System Administration Guide* explains how to reference Global Parameter values in a parameter.

Integration Resource

This page is used to view details of the resource integrated during execution of the service flow.

Integration Server List

The Integration Server List page allows you to view and maintain a list of ABI Integration Servers.

Integration Service Parameters

The Integration Service Parameters page presents the default parameter values from Service Definition and allows you to specify alternate values to be used in this specific service flow.

Related Topic(s):

- Schedule Flow

Integration Service Status Log

The Integration Service Status log page allows you to view the detailed component by component log produced by the service flow.

Maximo Event Detail

The ABI Maximo Event Detail (ABIMXEVE) page displays ABI events and related errors, if any, that are originating from/ destined for the Maximo application. This page provides basic search functionality. The View Source link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the status field and selecting Save.

Meridian Global Event Detail

The ABI Meridian Global Event Detail (ABIMREVE) page displays ABI events and related errors, if any, that are originating from/ destined for the Meridian Global application. This page provides basic search functionality. The **View Source** link is active for incoming events and it links to the source application event. You can resubmit a failed event by changing the status field and selecting **Save**.

NEOGOV Event Detail

The ABI Neogov Event Detail (ABINGEVE) page stores the Advantage Business Integration (ABI) events and related errors, if any. By looking at the **Direction** field you can see whether the event originated in or is destined for the NEOGOV application.

If the event is originating in the NEOGOV application the **Direction** is *Outbound.* Outbound events cannot be changed but they can be duplicated for reprocessing when they are in *Error* status. This is done by changing the **Status** to *Ready* and clicking on the **Save** link. This is the only update possible on this page. Status is the only unprotected data field.

If the event originated in ABI and is destined for the NEOGOV application the **Direction** is *Inbound*. Unlike outbound events, inbound events cannot be directly reprocessed. Instead, a **View Source** link is active for inbound events. You can use this link to navigate to the outbound event that caused ABI to generate the inbound event. This outbound event will be presented on the Event Detail for the ABI partner application that generated that event (for example, Advantage Financial Event Details). If the outbound event is in *Error* Status, reprocessing can be then be initiated by changing the **Status** of the outbound event to *Ready.*

The **Related Event** link navigates you to the Related Events page, which shows the related events for the current event. For outbound events, the related events are:

- Any reprocessing events generated by changing the Status of the current event.
- Any inbound events generated by ABI while processing or reprocessing this event.

For inbound events, the related events are:

- The outbound event that caused ABI to generate the current event.
- Any other inbound events generated by ABI while processing that outbound event.

The **Service Flow** link will navigate you to the Service Flow page, which will show the service flows that ABI used to process the current event. From that page the **Logs** link can be used to review the details of all processing for the current event.

Notification Destination

The Notification Destination (NDES) page allows you to set up multiple destinations for a Notification Profile and Provider. The destination address must match the Destination Type. For example: If Destination Type is *Email* then Destination should be a valid email address.

Notification Profile

The Notification Profile (NPRF) page allows you to set up notification profiles. The Notification Profiles are then assigned to one or more Notification Destinations on the Notification Destination (NDES) page. The system sends notification to a profile that is eventually routed to all destinations belonging to the profile. The Profile ID should be unique and Profile Name can be any desired name.

Notification Provider

The Notification Provider (NPRO) page allows you to set up Notification Provider details such as Provider Realm and Provider Implementation Class for the notification channel. The Notification Providers are then assigned to one or more Notification Destinations on the Notification Destination (NDES) page.

Related Events

The Related Events (ABIREEV) page lists the records (Events) for all of the ABI Partners. Use this inquiry page to view the related events, that is, if the current selection is an Outbound Event, we can view the clones of that Outbound Event and Inbound Events originating from that Outbound Event. If the current selection is an Inbound Event, we can view the clones of that Inbound Event and the Outbound Event that caused this Inbound Event.

Each Partner will have the following common search fields out of which Direction and Event Partner fields are mandatory:

- Direction: Inbound or Outbound.
- Event Partner: The name of the ABI partner.
- Event Status: Status of the Event.

The following Partner specific fields are mandatory.

For ADVFIN, ADVHRM, INFOADVANTAGE:

- Event Id: The Event Id of the Event selected.

For MAXIMO:

- Message Id: The Message Id of the Event selected.

For BRASSRING, MERIDIAN, and ADVPERMITTING:

- Client Id: The Client Id of the Event selected.
- Transact Id: The Transact Id of the Event selected.
- Packet Id: The Packet Id of the Event selected.

Schedule Flow

The Schedule Flow page is used to schedule a flow for the selected service. The **Submit** link is used to schedule the service flow. Prior to clicking on the Submit link you can change the Scheduled Date, the Expiration Date and/or override the default parameter values by clicking on the **Edit Parameters** link.

Related Topic(s):

- Integration Service Parameters

Service Catalog

The Service Catalog page lists the services that are defined to Advantage Business Integration (ABI). Any service having a status of *Active* is available for scheduling a new service flow via the page's **Schedule Flow** link.

Service Definition

The Service Definition page allows you to add, change, delete, or view the details and parameters for ABI Services.

Service Flows

The Service Flows page lists the Service Flows for ABI Services. By default, all service flows are listed in reverse order of their creation. This allows the most recently created service flows to be easily viewed. It is possible to filter the list by entering Service Name, Service Group and/or Status as selection criteria.

Service Groups

The Service Groups page allows you to add and maintain the Service Groups to which Integration Services may belong. The Service Groups are used to select related services as a group.

Service Pool

The Service Pool page allows you to set up new thread pools having characteristics such as Service Pool ID, Service Pool Name and Thread Count.

Service Type

The Service Type page allows you to map a Service Pool to a pre-defined Service Type.

Synchronous Event Detail

The ABI Synchronous Event Detail (ABISYNC) page displays ABI events and related errors, if any, that are originating from/ destined for the Synchronous Event Handler. This page provides basic search functionality. You can resubmit a failed event by changing the status field and selecting **Save**.

Transaction Integration Log

The Transaction Integration Log page is used to display a list of transactions that have integrated to, or have attempted to integrate to, the Advantage application.

Transaction Integration Messages

The Transaction Integration Messages page is used to display messages associated with transactions that have integrated to, or have attempted to integrate to, the Advantage application.

Advantage Connect

Advantage Connect (AC) is a multi-tenant integration framework for the Advantage ERP applications. Advantage Connect exposes Advantage ERP business functions as a set of Rest APIs and messages. Rest API specifications are published using OpenAPI specification in YAML format.

Please refer to the *CGI Advantage Connect - Technical Architecture Guide* for more information on the technical architecture overview of Advantage Connect.

The Advantage Connect configuration and management pages allow administrators to perform configuration and operational management. This chapter provides information on configuration and operational related pages.

The following configuration pages are used only for setup and configuration during initial setup and then required only on an as needed basis:

- Source Applications
- Tenant Properties
- Web Service Operations
- Event/Notification to Request Message Mapping
- Endpoint Names
- External Listeners
- General Properties
- Orchestration Services
- Messages
- Notification Recipient Addresses
- CEH Notification Subscriptions
- ER Delivery Subscriptions

The following operational pages are used on a day-to-day basis to view services requests, responses, log entries, and so on:

- Service Requests
- Log Entry
- ER Delivery Status
- External Listeners Status

Source Applications

A Source Application in Connect represents an external or internal client application that can submit a service request to Advantage Connect. Each Source Application belongs to a single Tenant Id. A Tenant Id may have one or more Source Applications. For every service request received by Connect, it must be able to determine the Source Application.

If an API Manager is being used in front of Connect to secure APIs, then a Client Application defined within the API Manager has a one-to-one correspondence with the Source Application defined within Connect.

By default, Connect comes pre-configured with the Internal Source Applications for Advantage Financial, Advantage HRM, and Advantage Admin, which are internal Advantage applications. It also comes pre-configured with a default Source Application (DEFAULT_SOURCE_APP) representing an external client application. All pre-configured Source Applications are assigned to a default tenant.

The Source Applications (CSRCAPP) page displays a list of Source Applications for the current Tenant Id. This page also allows users to create, edit, and view the Source Application for the current Tenant Id.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

Tenant Properties

The Tenant Properties (CTNTPRP) page displays a list of properties and their values for the current Tenant Id. This page also allows a user to edit property values.

Below is an example of a Tenant specific Property Code and Property Value:

Tenant Id	Source Application	Property Code	Property Value
DEFAULT_TENANT	$COMMON	fin-job-add-app-username	sa
DEFAULT_TENANT	$COMMON	fin-job-add-run-mode	2
DEFAULT_TENANT	$COMMON	hrm-advapi-rest-service-url	http://server:port/VLS1/advapi/jaxrs

In almost all scenarios, a tenant Property Value is going to be the same for all Source Applications belonging to the same Tenant Id. In the above example, the tenant property codes/values for the selected Tenant Id are applicable to all Source Applications, since the Source Application is *$COMMON*.

However, there might be some rare scenarios where a property value for same Tenant Id needs to be different based on the Source Application of a service request. In those rare scenarios, the user can create a row using the same Tenant Id and same Property Code but use a specific Source Application, instead of using $COMMON. This allows the property value to be different for that specific Source Application.

Most of the tenant properties are set post-deployment. Connect caches the tenant properties and the changes take affect only after the re-start of Connect modules.

Web Service Operations

Connect comes pre-configured with all baseline Web Service Names and their operations. However, some configuration changes are allowed to change the behavior of a Web Service Operation if the request comes from a specific Source Application.

The Web Service Operations (CWSOBSP) page allows administrators to view existing web service operations and allows them to configure the Web Service Operations by Source Application.

The following table provides the description of various fields for the Web Service Operations by Source Application entity:

Field	Description
Web Service Name	Indicates the Web Service Name.
Operation Name	Indicates the Web Service Operation Name.
Source Application	It is mandatory and must be a valid Source Application for current tenant. • If set to *Active*, a client corresponding to the Source Application can invoke this web service operation. • If set to *Inactive*, a client corresponding to the Source Application cannot invoke this web service operation.
Orchestration Service Name	Orchestration Service Name that processes service requests for this Web Service operation. When configuring Web Service Operation by Source Application, set this field to the same value configured for the WS Operation in baseline. Note: A different orchestration service name can be entered but that usually involves development effort to develop a new Orchestration Service.
Processing Pattern	It is set to *Request Response* for all Web Service operations in baseline. You can change this behavior by setting a different processing pattern. For example, you may want to use *ONEWAY* processing pattern for a specific WS Operation if a request comes from a specific Source Application. Usually, *ONEWAY* processing pattern is applicable only for POST/PATCH operation types.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

Event/Notification to Request Message Mapping

Advantage applications can generate events based on their respective Audit Control configuration. These events are put into a table by Advantage applications within their respective database schema.

External Listeners of type “adv-event-table” are pre-configured in Connect to listen for these internally generated events for default tenant. For processing an event (which is a row in IN_AUD_LOG table), Connect converts an event into a Request Message (JSON format) for further processing within Connect.

The Event/Notification to Request Message Mapping (CENMSGMP) page allows a user to modify the existing mapping for current tenant. This mapping from an Event/Notification Name to the corresponding Request Message Name is tenant dependent.

The following table provides the description of various fields for the Event/Notification to Request Message mapping entity:

Field	Description
Host System	Advantage Host System. Valid values are: • fin – Advantage Financial • hrm – Advantage HRM • admin – Advantage Admin
Source Application	Indicates the Source Application Name. Only the Internal Source Application Names for the current tenant are valid. Internal Source Applications are those where the Host System Name is not null.
Event Process/Table Name	Indicates the Advantage Process Name or Advantage Table Name. This information needs to be determined from the respective Advantage Application. Note: When an event is generated by an Advantage application, it populates PROC_NM or TBL_NM columns in the IN_AUD_LOG table row. Connect tries both of these two column values to match against the Event Process/Table Name field value configured on this page to determine a mapping row.
Message Name	Indicates the Message Name. When Connect converts an event to a JSON message, this message name is used. When creating a new row on this page, the Message Name to be used must be configured first on the Message Name page.
Active?	Valid values are *Yes* or *No*. • When set to *No*, event is ignored by Connect.

	• When set to *Yes*, event is converted to JSON using the message name specified in the Message Name field. For pre-configured mappings, you may set the Active flag to *No* to ignore the events.
Endpoint Name	Once an event is converted to a message, it is put into a JMS queue of internal Artemis Queue broker. This Endpoint name points to that JMS Queue Name. Endpoint's type must be "queue-jms". Endpoint names for these JMS queues are already pre-configured in Connect. Endpoint names are: • "en-adv-fin-queue-jms" – Use this endpoint for FIN related events. • "end-adv-hrm-queue-jms" – Use this endpoint for HRM related events. • "en-adv-admin-queue-jms" - Use this endpoint for Admin related events.

Endpoint Names

An Endpoint in general represents one end of a communication channel. In Connect, an Endpoint type indicates the type of communication such as queue, webhook, directory, and so forth. An Endpoint name always has only one Endpoint type. An Endpoint name belongs to only one tenant.

The Endpoint Names (CENDPNT) page displays a list of Endpoint Names for the current Tenant Id. This page also allows the user to create, edit, and view an Endpoint Name for the current Tenant Id.

Connect supports the following Endpoint types:

- queue-jms: This Endpoint type represents JMS queues hosted in Connect internal Artemis queue server.
- queue-broker: This Endpoint type represents JMS queues hosted in an external queue broker that supports AMQP based connections.
- directory: This Endpoint type represents directories that are directly accessible to Connect.
- webhook: This Endpoint type represents Webhook, which supports HTTP POST.
- adv-event-table: This Endpoint represents Advantage Event table.

Every Endpoint type has one or more properties, which serve as a template of properties. When an Endpoint name is created, endpoint properties are also created based on its endpoint type properties. Once an endpoint name has been created, its Endpoint type cannot be changed. An Endpoint Name's property values provides details such as queue name, webhook URL, queue broker host name, and credentials depending on the type of endpoint.

Connect uses Endpoint names for two purposes:

- External Listeners – External Listeners are technical connectors that are used by Gateway IN to receive messages, events, or files. Each External Listener has an associated endpoint name.

- ER Delivery Subscription – Event/Response Delivery subscription is used by Gateway OUT to send service responses and event notifications to endpoints. Each ER Delivery Subscription may have one or more endpoint names.

Note that Endpoint configuration is not required for using Rest APIs hosted in Connect.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

External Listeners

External Listeners are technical connectors that are used to receive messages, events, or files. Administrators can configure multiple External Listeners. Each External Listener runs in a Gateway IN module as a separate camel route.

The External Listeners (CXTLSNR) page allows a user to view, edit, and create external listeners. Only external listeners assigned to the current tenant's Source Applications are displayed.

Note that External Listener configuration is not required for using Rest APIs hosted in Connect.

Before configuring a new External Listener, create an Endpoint name using an appropriate Endpoint type and set the property values.

The following table provides detail on each field for an External Listener:

Field	Description
External Listener Name	This is a unique identifier for the External Listener Name. Do not use spaces or special characters. Keep it simple using alphanumeric characters and dash or underscore. No need to repeat word "Listener" as suffix.
Endpoint Name	Only those Endpoint names that belong to the current tenant can be selected that has an Endpoint type of either queue-broker or directory or adv-event-table or queue-jms.
Source Application	Only those Source Applications that belong to the current tenant are available. When configuring External Listener for queue-broker or directory, use a Source Application that represents external client applications.
Gateway IN Module Name	Select the Gateway IN module name. By default, there is only one Gateway IN module name.
Active?	• If set to *Yes*, the Gateway IN module will start the External Listener during startup.

	• If set to *No*, the Gateway IN module will not start the External Listener during startup.
Sequential?	• For adv-event-table and directory based listeners, it must be set to *Yes*. • If set to *Yes* for queue-based listeners, it represents that Gateway IN will wait for a message to be processed by the Orchestration module before reading the next message from queue. For directory listeners, it represents that Gateway IN will wait for a job to be created in the corresponding Advantage host application before reading the next file. • If set to *No* for queue-based listeners, it represents that Gateway IN will not wait for a message to be processed by the Orchestration module before reading the next message from queue. For directory listeners, it represents that Gateway IN will not wait for a job to be created in the corresponding Advantage host application before reading the next file.
Concurrent?	• This flag should be set to *Yes* for queue-based listeners. If set to *Yes* – when multiple instances of a Gateway IN module name are running, listener will run in all of the instances. If set to *No* – when multiple instances of a Gateway IN module name are running, listener will actively run in only one of the instances. • For adv-event-table and directory based listeners, this flag must be set to *No*.
Message Format	Set it to *XML* or *JSON*. • For queue and adv-event-table based listeners, set it to *JSON*. • For directory-based listeners, set it to *XML*.
Orchestration Service Name	This field is used only for directory-based listeners. • Set it to *GeneralFINFileXMLOrchService* if files are for the Financial application. • Set it to *GeneralHRMFileXMLOrchService* if files are for the HRM application. Gateway IN does not peek into the contents of files that it receives.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

General Properties

The General Properties (CGENPROP) page allows a user to view, edit, copy, and delete general properties. Creating a new general property is not allowed. The following table provides detail on each field:

Field	Description
Module Type	The Connect Module Type for which the general property is applicable. It cannot be updated.
Module Name	The Module Name $COMMON indicates that the property value is applicable to all module names of that module type. The Module Name other than $COMMON indicates that the property value is applicable to the specific module name only. It cannot be updated.
Property Code	Indicates the General Property Code. It cannot be updated.
Property Value	Indicates the General Property Value
Record Timestamp	The Date and Time when the property code/value was last updated. It cannot be updated.

When copying a general property, the Module Type and Property Code fields cannot be changed for the new record. Copying a general property validates the following for a new record:

- General Property Code already exists for the same Module Type.
- Module Name selected is valid for the Module Type.
- Module Name is not $COMMON.

Delete operation is allowed for General Properties; however, it is intended for administrators to use it with extreme caution. It should be used only for deleting those properties that were created (using copy) by administrators (that is, the properties that were not part of the initial deployment).

All modules cache their respective properties during startup. Any changes will not take effect until module is re-started.

Below are examples of some General Properties:

Module Type	Module Name	Property Code	Property Value
ORCH	Orch-Tran-FIN-Def	concurrentConsumer	5

ORCH	Orch-Tran-FIN-Def	req-queue-name	artemis:queue:jms/AC-INT-ORCH-FIN-DEF-REQ
AC	$COMMON	trustStoreType	JKS
CNM	Config-Mgmt-Def	purge-commit-rows	100

The $COMMON module name is used to indicate that this property value is applicable to all module names of that module type. If there is a need, a property code can be duplicated (within module type) to use a specific module name to set a different property value. This avoids having to keep same property code/value for multiple module names unnecessarily.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

Orchestration Services

The Orchestration Services (CORCHSRV) page displays all existing orchestration services and their properties. This page allows adding new orchestration services and updating existing orchestration service properties. It has two tabs: Orchestration Service Name and Orchestration Service Properties. The following table provides detail on each field on these two tabs:

Field	Description
Orchestration Service Name	
Orch Service Name	Indicates the Orchestration Service Name. It cannot be updated.
Service Template	Orchestration Service Templates are provided in a drop-down list box.
Module Name	Indicates the Module Name, where Orchestration Service will be hosted, that is, the module name where orchestration service will process the service requests. At run time, once Gateway IN has determined Orchestration Service Name that should process a file or message or web service call, it then looks at this configuration to determine which module should receive the service request.
Orchestration Service Properties	
Source Application	Source Application other than $COMMON indicates that the property value should be used only when processing a service request from that specific source application.

	$COMMON indicates if the property code/value record is not found for the specific source application, then the record with $COMMON should be used to get the property value. By default, all the orchestration property code/value records in Day-Zero DML are created for $COMMON. It cannot be updated.
Property Code	Indicates the Orchestration Property Code. It cannot be updated.
Property Value	Indicates the Orchestration Property Value.
Record Date	The Date and Time when the property code/value was last updated. It cannot be updated.

Below are examples of some Orchestration Service Properties:

Orchestration Service Name	Source Application	Property Code	Property Value
AdvApiFinUpdateOrchService	$COMMON	allow-update-of-final	true
AdvApiFinUpdateOrchService	$COMMON	allow-patch-to-add-null-values	true

The $COMMON source application is used to indicate that this property value is applicable irrespective of the source of service request to set a value specific to the source application. If there is a need, a property code can be duplicated (within Orchestration Service Name) to use a specific source application name to set a different property value. However, this is not allowed from this page and can only be done via a database.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

Messages

Messages are delivered to Connect via external queues or from Advantage internal events. Payload is in JSON format.

Connect comes pre-configured with many Message Names, which are used for importing a transaction. In general, any Transaction Type for which the Import Web Service operation is supported, a message name is also pre-configured to allow external clients to send a message via a queue, instead of invoking Rest API. For example, the "PRrequestRootEntity" Message Name is pre-configured for the import of a Payment Request Transaction Id.

The Messages (CMESGS) page allows administrators to create/view/edit existing Message Names. It also allows them to configure the Message Names by Source Application. It has two tabs: Message and Message by Source App. The following table provides detail on each field on these two tabs:

Field	Description
Message	
Message Name	Indicates the Message Name. It cannot be updated.
Description	Description for the message name.
Status	• If set to *Active*, any client can send this message unless it is overridden at the Source Application level. • If set to *Inactive*, no client can send this message unless it is overridden at the Source Application level.
Format	Format is either JSON or XML. Note: All existing messages are supported in JSON format only. XML is being allowed for future scenario that requires XML formatted message.
Orch Service Name	Indicates the Orchestration Service Name that process the service requests for this message. It can be overridden at Source Application level.
Preserve Message Order?	• A value of *Yes* indicates that the messages received for same key values are processed in the order they are received, one at a time. Order is preserved even if messages were received on different external listeners. • A value of *No* indicates messages received for same key values may be processed in parallel. This may cause problems if sequential processing of messages with same key values is important because Connect is a multi-threaded application.
Key Tags	It is mandatory, if the Preserve Message Order flag is set to *Yes*. Provide comma separated field names. Note that field names are case-sensitive and do not use any spacing before or after comma.
Tran Type	Some message names are specific to an Advantage Transaction Type. For example, message name “CRrequestRootEntity” has a Tran Type of *CR*. Similarly, some messages are not specific to any Advantage Transaction Type. For example, message name “advEventJobCompleted” is not specific to any Transaction Type.

Message Name by Source App	
Source Application	It is mandatory and must be a valid Source Application for current tenant. It cannot be updated.
Status	• If set to *Active*, a client corresponding to the Source Application can send this message. • If set to *Inactive*, a client corresponding to the Source Application cannot send this message.
Orch Service Name	Indicates the Orchestration Service Name that process the service requests for this message. When configuring message name by Source Application, set this field to the same value configured for the message in baseline. Note: A different Orchestration Service Name can be entered but that usually involves development effort to develop a new Orchestration Service.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

Notification Recipient Addresses

The Notification Recipient Addresses (CRECADR) page displays a list of all configured recipient addresses. It also allows user to add, update, and delete an existing recipient address code.

The following table provides detail on each field on the Notification Recipient Addresses page:

Field	Description
Recipient Address Code	This is a unique identifier for the address code.
Recipient Addresses	This is semi-colon separated email addresses. Please do not put any spaces before or after semi-colon.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

CEH Notification Subscriptions

The CEH Notification Subscriptions (CNOTSUBS) page displays a list of all configured CEH Subscriptions that are valid for the current tenant. It also allows the user to add, update, and delete an existing CEH Notification Subscription.

A CEH Notification Subscription can have one or more criteria. When a CEH Entry is matched against CEH Subscription criteria, all criteria of a subscription must be true for the subscription to be considered as matched.

The CNOTSUBS page has two tabs: CEH Subscriptions and Criteria. The following table provides detail on each field for a CEH Notification Subscription:

Field	Description
CEH Subscriptions	
Subscription Name	This is a unique identifier for the CEH Notification Subscription Name. Do not use spaces or special characters.
Description	Description for the subscription.
Active?	• If set to *Yes*, this subscription will be used for matching purpose. • If set to *No*, this subscription will not be used for matching purpose.
Notification Template Name	The Notification Templates allow a notification message body to be flexible. A pre-configured list of templates is available in the drop-down list box.
Notification Channel Name	The Notification “Email” Channel provides Email server connection configuration. A pre-configured list of email channels is available in the drop-down list box.
Notification Recipient Address Code	The Notification Recipient Address Code provides one or more email addresses to which the notification should be sent. A pre-configured list of Address Codes is available in the drop-down list box.
Criteria	
Criteria Number	Enter a unique number within the Subscription Name (for example, 1,2,3, and so on). Ordering of criteria does not matter.
Object Type	This field name must be available in the CEH Entry. A picklist is provided to help user select a valid field name. Valid values are: WS_NAME, WS_OPER_NAME, MESSAGE_NAME, SOURCE_APP_NAME, ERROR_ID, MODULE_TYPE, MODULE_NAME, OS_NAME, OS_TEMPLATE_CODE and SEVERITY, ERROR_TEXT

	At run time, Connect uses this field name to get the corresponding field value from the CEH Entry.
Operator	This is an operator code. Valid values are *Equals*, *CONTAINS,* and *IN*.
Criteria Value	Enter a value that will be used for matching purposes. At runtime, Connect uses the Object Type value from CEH Entry, Operator, and this Criteria Value to determine if this subscription criteria is true or not. Assume a Subscription Criteria is setup as: Object Type is “WS_NAME”, Operator is “Equals” and Criteria Value is “PaymentRequest”. Note that double-quotes are being used here just to make it clear; do not enclose criteria value in double or single quotes. When a CEH Entry is received by the Connect Error Handler module, it will determine its Web Service Name and then check if it equals PaymentRequest. Similarly, if Operator is “IN” then criteria value must be comma-separated values, for example, PaymentRequest, CashReceipt and so on. If a match is true, Connect will then check the next criteria for the same subscription name. If checks for all criteria of a subscription name are true, then the subscription is considered as matched.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

ER Delivery Subscriptions

The Event/Response (ER) Subscription entity allows configuration based on the original web service name, operation name, message name, processing outcome, and so forth so that events and responses are routed only if configured to do so. A subscription may have one or more endpoints of Endpoint type queue-broker or webhook.

The ER Delivery Subscriptions (CERDSUB) page allows a user to search for all configured subscriptions for the current tenant based on web service name, source application, and message name. It also allows the user to add, update, and delete subscriptions, configure optional content-based criteria and one or more endpoints.

The following table provides detail on each field for an ER Delivery Subscription:

Field	Description
Event Response Delivery Subscription	

Subscription Number	This is a unique auto-generated number for the Subscription.
ERD Subscription Name	A user-defined Subscription Name. This is not unique.
Source Application	The Source Application of the original service request or event.
Original Web Service Name	The Web Service Name of the original service request.
Original WS Operation	The Web Service Operation of the original service request.
Original Message Name	The Message Name corresponding to the original service request or event.
External Listener Name	External Listener on which the original service request was received. It is not applicable to the web service based requests.
Processing Outcome	Processing Outcome (SUCCESS or FAILURE) of the original service request.
Transformation Active?	Gateway Out allows transformation of event/response payloads from the baseline JSON format to subscription specific JSON format using Camel Jolt. • Set this flag to *true*, if transformation needs to be done before sending the payload. • Set this flag to *false*, if transformation should not be used.
Transformation (Jolt) Script	If Transformation Active flag is *true*, then the Camel Jolt script must be provided. This script is set by implementation teams or customers on an as-needed basis. It is specific to a payload.
ER Delivery Subscription Criteria	
Criteria Number	This is a unique auto-generated number for the Criteria.
Field Name	The Field Name that should be present in either Gateway Out Metadata or in event / response payload.
Field Source	Indicates the source of the field name: METADATA or PAYLOAD.

Field Operator	This is an operator code. Valid values are *Equals*, *CONTAINS,* and *IN*.
Criteria Value	Enter a value that will be used for matching purpose. At runtime, Connect uses value based on the Field Name / Field Source, Operator, and this Criteria Value to determine if this subscription criteria is true or not. If checks for all criteria of a subscription are true, then the subscription is considered as matched.
ER Delivery Subscription Endpoints	
Endpoint Name	The Endpoint Name to which the Event/Response needs to be sent.
Active?	• If set to *Yes*, ER Delivery will be attempted for this Endpoint. • If set to *No*, ER Delivery will not be attempted for this Endpoint.
Retry wait seconds	In case of a failure in sending to a specific endpoint, Connect will try to re-send the event/response to the same endpoint after this many number of seconds. Default value is *300* seconds.
Max Tries	Indicates the number of times that Connect will attempt to send an event/response to this endpoint.

Please refer to the *CGI Advantage Connect - Configuration & Management Guide* for more information on this page.

Service Requests

The Service Requests (CSRVREQ) page allows you to search for service requests using various criteria such as request received date range, web service name, message name, listener name, processing outcome, and so forth. The Service Request Detail page shows details such as date/time when a request was received, web service name, operation name, the orchestration service name that processed the request, processing outcome, request payload, service response, and service response payload.

This page does not allow any modifications to any service request or service response. Using the row-level menu, you can navigate to the corresponding Log Entry page to view any log entries for a service request. In addition, you can also navigate to the ER Delivery Status page to view any ER Delivery Status for a service request.

The following table provides the description of various Service Request fields:

Field	Description
Service Request	
Service Request Number	A unique number for a Service Request. It is internally generated by Connect.
Correlation Id	An identifier that can be included by client applications when submitting a request to Connect. It is used for easier tracing of service requests across multiple systems.
Source Application	It identifies the source application that sent a service request.
Processing Pattern	For web services, it is either *Request Response* or *One Way*. For events/messages, it is *Sequential* or *Asynchronous*.
Processing Status	Valid values are: • *Received*: Gateway IN has received request. • *Routed*: Request has been passed to Orchestration module. • *Processed*: Request has been processed. • *Timeout*: Gateway IN timed out while waiting for response (applicable in case of Request/Response and Sequential processing patterns).
Processing Outcome	Valid values are: • *Success*: service request processing passed. • *Failure*: service request processing failed. • *Partial-Failure*: service request was processed but with some failures.
Web Service	Indicates the Web Service Name. Applicable if Service Request represents a Web Service call.
Web Service Operation	Indicates the Web Service Operation Name. Applicable if Service Request represents a Web Service call.
Message Name	Indicates the Message Name. Applicable if service request represents a message or an internal event.

Key Values	Every WS Operation and Message Name in Connect has associated Key Tags (or Key Field Names). This field represents concatenated Key values for the service request.
Format	Indicates the format of service request payload. • For messages, it is JSON. • For web services, it is usually JSON or application. • For files, it is usually XML.
Listener	External Listener over which the service request was received. Applicable to Non Rest API related service requests.
Request Received Timestamp	Exact date/time when a service request was received by Connect in the local time zone.
Gateway IN Module Name	Gateway IN module name that received the service request.
Gateway IN Module Instance	Gateway IN module instance Id that received the service request.
Orchestration Module	Orchestration module name that processed the service request.
Orchestration Service name	Orchestration Service name that processed the service request.
Orchestration Service template	Orchestration Service template associated with Orchestration Service name.
Message Order Group Hash	Gateway IN uses configured Key Tag/Field Names for message grouping purposes. This field represents a hash/mod of the key values. Used only for internal purposes.
Request Payload	Original payload (typically JSON) for the service request.
Service Response	
Service Response Number	A unique number generated by Connect for every service response.

Receive Instance ID	The Gateway IN Module Instance that received the service response. For Request/Response and Sequential processing patterns, it will be the same as the Gateway IN Module Instance field value. However, in case of one way / asynchronous processing patterns or timeout scenario, a different module instance may receive the service response.
Processing Status	Processing Status associated with the Service Response. It could be either *Failure* or *Success*.
Received Timestamp	Date/Time when the Gateway IN module received the service response.
Generated Timestamp	Date/Time when the service response was generated.
Response Payload	This indicates the Service Response payload.

Log Entry

The Log Entry (CLOGENT) page allows a user to search for log entries using various criteria. The Log Entry page shows details of the error including error text, stack trace, and original request payload if log entry is associated with a service request.

The Log Entry page also allows the user to modify certain fields:

- Reviewed: This field indicates that a user has reviewed the log entry; it allows system administrators to ignore log entries later on.
- Request Payload: This field displays original service request payload if log entry is associated with a service request. User may modify the request payload if needed, in which case the modified payload is saved along with log entry.
- Mark for Resubmit: If an error indicates that the service request failed completely and it failed in an Orchestration module, the user may re-submit the service request back to the same Orchestration module. Setting this flag to *Yes* will make the Resubmit sub-module to select the log entry, use modified request payload or original payload and re-submit the service request back to the orchestration module where it was originally failed.

The following table provides the description of various Log Entry fields:

Field	**Description**
Log Entry Number	A unique number for Log Entry. It is internally generated by Connect.

Service Request Number	The Service Request Number for which the log entry was generated.
Correlation Id	The Correlation Id of the service request.
Error Date	The Date/Time when the log entry was created.
Saved Date	The Date/Time when the log entry was saved in the database.
Error Id	A Connect Specific Error identifier. All Error Identifiers are table driven.
Error Severity	Every Error Identifier has an associated severity – Fatal (F), Error (E), and Warning (W), which is displayed in this field.
Error Text	Error Text is table driven with placeholders for important values.
Error Additional Text	Error Additional Text is not table driven. It is optionally included by Connect when generating an error.
Source Application	The Source Application of the service request.
Reviewed?	A flag indicating whether an administrator has already reviewed this log entry. This field is also a search criterion allowing administrators to search for only non-reviewed log entries
Tenant Id	The Tenant Id associated with the Source Application of the service request.
Web Service	The Web Service Name associated with the service request.
Web Service Operation	The Web Service Operation associated with the service request.
Message Name	The Message Name associated with the service request.
Module Name	The Module Name where the error occurred.
Module Instance Id	The Module Instance Id where the error occurred.
Orchestration Service	The Orchestration Service Name that encountered the error.

Tran Type	A service request is typically for a specific Advantage transaction. Tran Type indicates the type of transaction, for example, PR, JV, ESMT, and so forth.
Tran Code	The Transaction Code for the service request if it is for a specific Advantage transaction.
Department	The Department code for the service request if it is for a specific Advantage transaction.
Tran Id	The Tran Id for the service request if it is for a specific Advantage transaction.
Resubmit	
Reprocessable?	Yes indicates that the log entry may be re-submitted.
Marked for Resubmit	Yes indicates that an administrator has marked this log entry for re-submission.
Resubmitted?	Yes indicates that the associated service request has been re-submitted.
Resubmitted Date	The Date/Time when the associated service request was re-submitted.
Modified Payload	If request payload was modified by a user for the purpose of resubmitting the associated service request, then this field contains the modified payload; otherwise, it will contain the original service request payload.

ER Delivery Status

The ER Delivery Status (CERDLSTS) page allows a user to search for ER Delivery Status records based on criteria such as Service Request Number, delivery date, web service name, and so forth.

The following table provides the description of various ER Delivery Status fields:

Field	Description
ER Delivery Number	A unique number for the ER Delivery Status.
ER Delivery Subscription Number	Each ER Delivery Subscription has a unique internally generated number.

ER Delivery Subscription Name	Indicates the ER Delivery Subscription Name. Note that subscription name is not unique.
Endpoint Name	Indicates the ER Delivery Subscription Endpoint name.
Source Application	The Source Application of the Service Response or Event Notification.
Service Request Number	Indicates the Service Request Number.
Service Response Number	Indicates the Service Response Number.
Delivery Status	Valid values are *Success* or *Failure*.
Last Tried Date/Time	The Date/Time when the ER Delivery was submitted to endpoint.

External Listeners Status

The External Listeners Status (CEXTLST) page displays all the existing External Listeners and their status for the current tenant. It also allows a user to start/stop a specific External listener.

The following table provides detail on each field in the External Listeners Status page. All the fields displayed on this page is read-only.

Field	Description
External Listener Name	This is a unique identifier for the External Listener Name.
Endpoint Name	The Endpoint Name for the listener.
Source Application	The Source Application Name that represents the external client applications.
Gateway IN Module Name	Indicates the Gateway IN module name. By default, there is only one Gateway IN module name.
Active?	• If set to *Yes*, the Gateway IN module will start the External Listener during startup. • If set to *No*, the Gateway IN module will not start the External Listener during startup.

Sequential?	• If set to *Yes*, it represents that Gateway IN will wait for a service request to be processed by the Orchestration module before receiving next request. • If set to *No*, it represents that Gateway IN will not wait for a service request to be processed by the Orchestration module before receiving next request.
Concurrent?	• If set to *Yes* – when multiple instances of a Gateway IN module name are running, listener will run in all of the instances. • If set to *No* – when multiple instances of a Gateway IN module name are running, listener will actively run in only one of the instances.
Last Status	Valid values are *RUNNING* and *STOPPED.* • *RUNNING* – External Listener is running in at least one of the Gateway IN module instance. • *STOPPED* - External Listener is not running in any of the Gateway IN module instance.
Last Status Timestamp	The timestamp when the status was determined.
Instance Id	The Gateway IN module instance Id, where this External Listener is running.

› Record Level and Grid Actions

- **Start Listener** - Sends a request to Gateway IN for starting the specific listener. If it is stopped, then Gateway IN tries to start it. This action is considered as valid only if External Listener is marked as *Active*.
- **Stop Listener** - Sends a request to Gateway IN for stopping the specific listener. If it is running, then Gateway IN tries to stop it.

Enterprise Content Management Integration

An Enterprise Content Management (ECM) system is a web-based electronic library that allows you to safely archive and catalog any electronic file. ECM allows you to store and index analytical raw data, reports, and any other transactions. The stored files are available to be shared with other users. You can store any electronic file type, such as Microsoft Office transactions, Adobe PDF transactions, pictures, and so forth, as well as raw data and reports generated by the CGI Advantage application. ECM allows you to easily collect, organize, search and review all of your data.

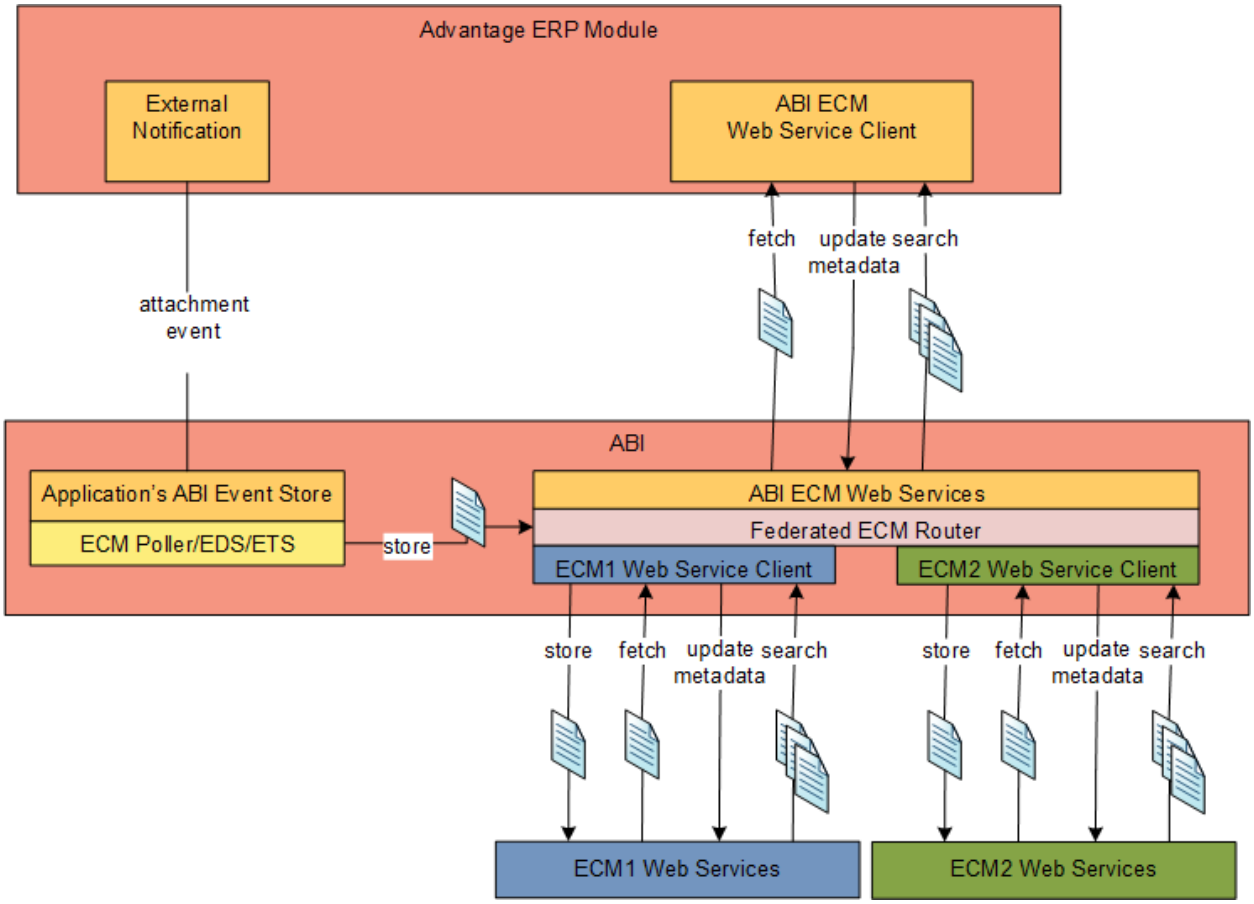
The Advantage application can be simultaneously integrated with multiple ECM products and also with multiple storage repositories for an ECM product. This is to accommodate the need for Departments to adopt different ECM products to meet their specialized requirements.

The following topics describes how ECM can be integrated with CGI Advantage along with the required setup tables and usage considerations.

How is ECM Integrated with CGI Advantage?

CGI Advantage allows you to efficiently store attachments associated with the Advantage pages and transactions in one of the integrated ECM systems instead of storing them as binary objects in the CGI Advantage database.

The Object Attachment Framework (OAF) used in Advantage stores the attachment in the Advantage database at the point the user saves the attachment without regard to whether the customer wants those attachments to eventually reside in its ECM repositories rather than the Advantage database. Another part of OAF, running asynchronously, then completes all ECM related tasks via asynchronous event services running in ABI. These ABI services provide all metadata handling, ECM routing, and ECM storage.



Key objectives are listed below:

- To store attachments from CGI Advantage pages and transactions in an ECM system asynchronously.
- To retrieve and view attachments from an ECM system using CGI Advantage pages and transactions asynchronously.
- To provide authentication between CGI Advantage and ECM in a Web service message. The authentication credentials are configured in ABI.

Metadata Mappings

Within the ECM the content is organized into repositories. Within each repository the content is further broken down into various types of objects (for example, invoices, pay checks, and wiring diagrams) with each type of object having a set of properties (for example, invoice number, employee number, and pay date) that can be used to search for relevant content. Before Advantage can correctly store content in the ECM in the correct type of object with an appropriate set of property values a relationship needs to be established between the ECM's classification system and Advantage's tables and data attributes. The following pages are used to establish that mapping:

- ECM Type (ECMTYP) defines the types of objects stored in the repository to which Advantage will be connected.

- ECM Metadata (ECMDATA) defines the properties of each object type in the repository to which Advantage will be connected.
- CGI Advantage Metadata (ADVDATA) is automatically populated by a batch utility to make the attributes of Advantage tables and transaction components available for mapping to ECM object types and properties.
- ECM Metadata Mapping (ECMDM) defines the individual mapping of Advantage metadata exposed in the ADVDATA table to ECM metadata exposed in the ECMDATA table.

ECM Type

The ECM Type (ECMTYP) page identifies the ECM object types that are used to store CGI Advantage attachments. This table may be provided as initial “Day Zero” data by the CGI Advantage application. For example, the CGI Advantage Financial application’s “Day Zero” data expects the ECM to be configured with ECM Type adv_fin_doc_md for use in storing Manual Disbursement related attachments. This value would need to be changed only if your ECM is configured to store Manual Disbursement attachments as an ECM object type other than adv_fin_doc_md.

Enter the definition for your ECM Type:

- The **ECM Type** is the category name of the type to be stored in ECM (for example, adv_fin_doc_md is the ECM Type for the Manual Disbursement related attachments). The value entered in this field must exactly match the type name created in ECM.
- The **Description** is a free form, user-defined description for the type (for example, Manual Disbursement Attachments). You can enter any description in this field; it does not have to match the type label.

ECM Metadata

The ECM Metadata (ECMDATA) page allows you to expand on the ECM Type definition by identifying the ECM metadata attributes available for mapping to CGI Advantage attributes. These metadata attributes are initially defined in ECM and subsequently entered in this CGI Advantage Administration page so that the ECM Metadata can be linked to CGI Advantage Metadata.

This table is primarily used as a onetime configuration during initial setup, but you can add records to it or delete records from it.

Advantage Metadata

The CGI Advantage Metadata (ADVDATA) page records all transaction/table fields available for mapping onto ECM Types and ECM metadata attributes. This table is generally completely populated by the Populate CGI Advantage Metadata batch job, but you can add records to it or delete records from it. You can run the Populate CGI Advantage Metadata Job to populate the CGI Advantage Metadata page. This job should be run under three circumstances:

- A new Transaction Type is added to Transaction Control (DCTRL), or
- A new table with the attachment feature has been added, or
- A column on a transaction or table is modified that supports the attachment feature.

To add an entry to the Advantage Metadata page in CGI Advantage:

- **Advantage Type** – Enter the Transaction Type for transactions and the Page Code for tables.
- **Component** – Enter the name of the transaction component or table data object (for example, MD_DOC_HDR).
- **Field** – Enter the attribute name for the desired field, for example, ACAT_CD.
- **Field Name** – Enter the name (caption) of the field, for example, Activity Category.

ECM Metadata Mapping

The ECM Metadata Mapping (ECMDM) page allows you to map the metadata between CGI Advantage and the ECM repository. For example, you could map CGI Advantage Type *MD* to various attributes/metadata (for example, *CHK_NO*).

Repository Mapping

ECM Repository Mapping (ECMMAP) defines the mapping of Advantage Types and Departments to ECM Repositories. ECM Repository can be configured for a specific Advantage Type and Department combination or for any combination of Advantage Type and Department. Entering ALL is considered a wildcard in the selection criteria. ECM repositories are registered in the ABI application inside the “ecm.properties” property file. The Repository Name on the ECM Repository Mapping page maps the “repositoryName” property configured in the “ecm.properties” file.

Connecting CGI Advantage to the ECM System

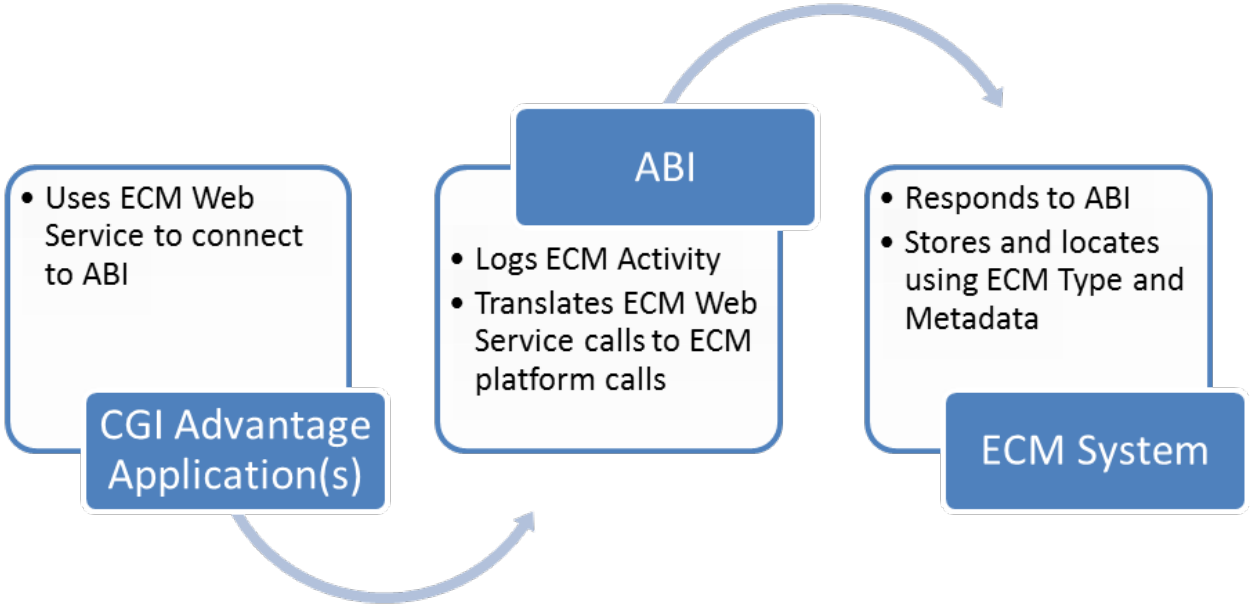
This following topics are intended to provide a basic understanding of the tasks required to make the web service and security connections and configurations required to integrate CGI Advantage to an ECM system. More detailed instructions are provided in install guides for the particular ECM system product.

Enable Audit Logging of IN_OBJ_ATT_STOR table

The Setup Audit Log and External Notification (AUDCTRL) page allows you to enable audit and external notification of the IN_OBJ_ATT_STOR table. This configuration triggers the moving of Advantage attachments to ECM. Sites that do not wish to use ECM should remove this configuration or set the External Notification flag as false (not selected).

Advantage Business Integration’s Role in ECM Integration

A more detailed view of the connection between a CGI Advantage application and the ECM system reveals that the connection is mediated by the Advantage Business Integration (ABI) engine.



CGI Advantage utilizes the Advantage Business Integration (ABI) engine to send service requests (secured using SSL) to ECM containing the attachment and the corresponding metadata information. Advantage determines the appropriate metadata to send to the ECM based on configuration settings.

The attachment's ECM Metadata values consist of the set of information associated with the Advantage Transaction that contains the attachment. For instance, when an Advantage user attaches a local transaction, the attachment inherits metadata from the Advantage Transaction and is sent to the ECM along with the attachment.

When an attachment is stored in Advantage using an 'upload' link, the following steps take place:

- A record for the "IN_OBJ_ATT_STOR" table with action type "INSERT" is added to the Audit Log table.
- ABI poller picks up this Audit Log entry to process an attachment Event in ABI. This event has a unique identifier for the attachment record.
- ABI sends a web service request containing a unique identifier to the source Advantage application and gathers the attachment and its metadata.
- Advantage Web service provides the following data to ABI:
    - ECM Repository Name
    - Metadata
    - The attachment file associated with the record
- ABI invokes ECM web services or ECM system API to store that attachment in the indicated ECM repository.
- In return, ABI receives the Object Id and the Direct View URL (if supported) for the respective attachment record in ECM. ABI invokes REST service to store this ECM attachment the Object

Id and the Direct View URL in Advantage. The Direct View URL can launch the attachment via the ECM system's native viewer.

Users can search for Advantage attachments stored in ECM through the Content Search page. In the background, Advantage gathers the search metadata and sends a Search web service request to the ECM system. The ECM System returns a unique identifier (for example,, Documentum's Chronicle ID) and associated metadata for matched records. If supported, the ECM system (for example, Filenet) also returns the Direct View URL.

While attaching the searched ECM Content, using the **Attach** link on the Content Search page, Advantage gathers the metadata associated with the Advantage transaction/page and sends a Meta Data Update web service request to the ECM system. ECM system inserts a new metadata record referencing the attachment record. New content for the attachment is not created.

Similarly, when the user views an attachment using the attachment page, Advantage gathers the ECM's unique identifier (for example, Chronicle ID) associated with the attachment and sends a Retrieve web service request to the ECM with the ECM's unique identifier. It uses ECM credentials configured in ABI. If the ECM finds a matching record, the attachment is returned from ECM and the user is able to open and view its contents in Advantage.

If an error is encountered, an error message is sent back to Advantage and the attachment will not open. This happens if any of the expected parameters' values do not match with the ECM, for example: Invalid Chronicle ID<<OBJSTART>>.

How to Attach ECM Content to Advantage Pages

Users can search for CGI Advantage attachments stored in an ECM system through the Content Search page. This search page can be accessed via the **Add from Repository** link on the Attachments pages for transactions and reference tables. The **Add from Repository** link is active whenever the CGI Advantage application has been configured to use an ECM system. When a user selects a particular CGI Advantage Type, the system infers all of the associated metadata information created for that CGI Advantage Type. The **Attach** action on the Content Search page links the existing stored content in ECM to the existing transaction/table from which you accessed the search page.

The ECM Synch batch job is used to synchronize the metadata information between the CGI Advantage and ECM systems. Once the metadata is initially sent to ECM, any changes made to the CGI Advantage transaction/table will not be re-integrated to ECM until this batch process is scheduled and executed again. For example, when a user first uploads an attachment to a CGI Advantage transaction, which is in draft phase, the values that were in the CGI Advantage metadata fields at that time will be integrated to ECM. If the user updates any of the CGI Advantage transaction's metadata fields after adding the attachment and subsequently submits the CGI Advantage transaction to Final, the updated metadata will only be sent to ECM when the Sync Batch Process is run.

Initialization File Settings for ECM

The following parameters on the ADV30Params.ini file are associated with ECM and should be setup accordingly.

- UseECMSearch is a parameter in the ADV30Params.ini file. This parameter's value determines whether the CGI Advantage attachments can be searched in an ECM system. If enabled (that is, UseECMSearch=true), then the Search link on attachments is enabled.

  ```
  ##########################################################################

  ## This property will enable/disable the Search link.

  ## It can be used only when ECM is used for attachments
  ##########################################################################

  UseECMSearch=true
  ```

- UseECMDirectView is a parameter in the ADV30Params.ini file. This parameter's value determines whether the attachment can be viewed directly in ECM system's native viewer without using ABI in the backend. If enabled (that is, UseECMDirectView=true), then the Direct View link on attachments is enabled. Since this direct link relies on the ECM product's security rather than Advantage's security, display of the ECM Attachment URL is optional.

- Documentum does not support this feature; therefore, this parameter should be disabled (that is, UseECMDirectView=false) for Documentum.

  ```
  ##########################################################################

  ## This property indicates if Advantage can view attachments in ECM directly, without using ##
  ABI. If enabled, "Direct View" link will be enabled on attachments page.
  ##########################################################################

  UseECMDirectView=true
  ```

- SeparateContentRecord is a parameter in the ADV30Params.ini file. When SeparateContentRecord=false, store content directly in metadata record and do not create a separate content record. When true, create a separate metadata record and content record. This property is only used when ECM is used to store attachments

  ```
  ##########################################################################

  SeparateContentRecord=false

   ##########################################################################
  ```

- ConvertMetadataToString is a parameter in the ADV30Params.ini file. When ConvertMetadataToString = true, add support for non-string ECM metadata attributes. This parameter is only used when Documentum ECM is used to store attachments.

  ```
  ##########################################################################

  ConvertMetadataToString=false

  ##########################################################################
  ```

- The ECM Retrieve Parameters section provides the configuration needed by the CGI Advantage application to use ABI's ECM Retrieve web service to retrieve existing attachments from ECM. Only the ABIECMRetrieveServiceURL parameter should be changed. It must provide the URL configured in ABI for this web service.

  ##

  ## ECM Retrieve Parameters

  ##

  ABIECMRetrieveServiceNm=RetrieveService

  ABIECMRetrieveServiceURL=(provide URL of ABI's ECM Retrieve web service)

  ABIECMRetrieveServicePortNm=RetrieveServicePort

  ABIECMRetrieveServiceWSDLLoc=RetrieveService?wsdl

  ABIECMRetrieveServiceWSDLNMSp=http://retrieveservice.ecm.abi

  ABIECMRetrieveServicePckgNM=abi.ecm.retrieveservice

- The ECM Update Parameters section provides the configuration needed by the CGI Advantage application to use ABI's ECM MetadataUpdate web service to update the metadata associated with existing attachments in the ECM. Only the ABIECMUpdateServiceURL parameter should be changed. It must provide the URL configured in ABI for this web service.

  ##

  ## ECM Update Parameters

  ##

  ABIECMUpdateServiceNm=MetadataUpdateService

  ABIECMUpdateServiceURL=(provide URL of ABI's ECM Metadata Update web service)

  ABIECMUpdateServicePortNm=MetadataUpdateServicePort

  ABIECMUpdateServiceWSDLLoc=MetadataUpdateService?wsdl

  ABIECMUpdateServiceWSDLNMSp=http://metadataupdateservice.ecm.abi

  ABIECMUpdateServicePckgNM=abi.ecm.metadataupdateservice

- The ECM Search Parameters section provides the configuration needed by the CGI Advantage application to use ABI's ECM Search web service to locate existing content in ECM. Only the ABIECMSearchServiceURL parameter should be changed. It must provide the URL configured in ABI for this web service.

  ##

  ## ECM Search Parameters

  ##

  ABIECMSearchServiceNm=SearchService

ABIECMSearchServiceURL=(provide URL of ABI's ECM Search web service)

ABIECMSearchServicePortNm=SearchServicePort

ABIECMSearchServiceWSDLLoc=SearchService?wsdl

ABIECMSearchServiceWSDLNMSp=http://searchservice.ecm.abi

ABIECMSearchServicePckgNM=abi.ecm.searchservice

Usage Considerations

The following topics cover usage considerations for ECM.

- How to Upload an Attachment in the ECM System
- How to Download an Attachment from the ECM System
- How to Attach ECM Content to Advantage Pages

How to Upload an Attachment in the ECM System

Navigate to the transaction or table record where you want to attach the content using the attachment feature. When an attachment is stored in CGI Advantage using the **Upload** actions, it is stored in the Advantage database at first, Then ABI poller picks up the corresponding IN_OBJ_ATT_STOR entry from Advantage Audit Log. ABI calls web service on Advantage to gather the attachment and its metadata. ABI then sends an Import web service request to the ECM system or uses ECM system's API to store the content.

How to Download an Attachment from the ECM System

There are two ways in which a user can download an ECM attachment in Advantage: using the **Download** action or using the **Direct View** link.

- The **Download** action utilizes ABI services to retrieve the attachment content.
- The **Direct View** action, if supported by the ECM system (for example, Filenet), displays the attachment in the native viewer of the respective ECM system without utilizing any ABI services.

Storing Attachments in a File System

The Advantage application allows users to store attachments in a database, in an ECM system, or in a file system. In order to store the attachments in a file system, the following properties should be set in the ADV30Params.ini file.

Parameter	Delivered Value	Parameter Description
UseCustomAttachment	false	This property indicates if Advantage attachments are processed by custom attachment handlers. If enabled (true).
CustomAttachmentHandler	com.amsinc.gems.adv.attachmenthandlers.AdvFileRepositoryHandler	This property holds the fully qualified class name that is being utilized to handle attachments.
FileRepositoryRootFolderName		This property holds the file path to store attachments when using AdvFileRepositoryHandler implementation. For example: =/apps/CGIADV/RTFiles/VLS1/Temp/

Accessibility

Web accessibility means people with disabilities can perceive, understand, navigate, and interact with the Web. CGI Advantage contains built-in accessibility features meant to support the Federal Government's Section 508 requirements as well as the World Wide Web Consortium's (W3C) Web Content Accessibility Guidelines (WCAG) 2.1. These accessibility features are available for and may be beneficial for all users. However, they were designed specifically to address accessibility issues, and the system has been certified for use with JAWS® for Windows used in conjunction with Microsoft Internet Explorer.

- Working with JAWS in Advantage Applications
- Working with Windows Eyes in Advantage Applications
- Usage Tips

Refer to the "Accessibility" section in the online help for additional information about Accessibility Features in Advantage.

Working with JAWS in Advantage Applications

JAWS (an acronym for Job Access with Speech) is a screen reader, which a software program for visually impaired users. When using JAWS to complete an on-line form, it is necessary to first enter the "Forms Mode". The Forms Mode allows the user to enter text into the form field using the keyboard. If the Forms Mode is not turned on before attempting to enter content, then the form field will not receive the correct keyboard letters.

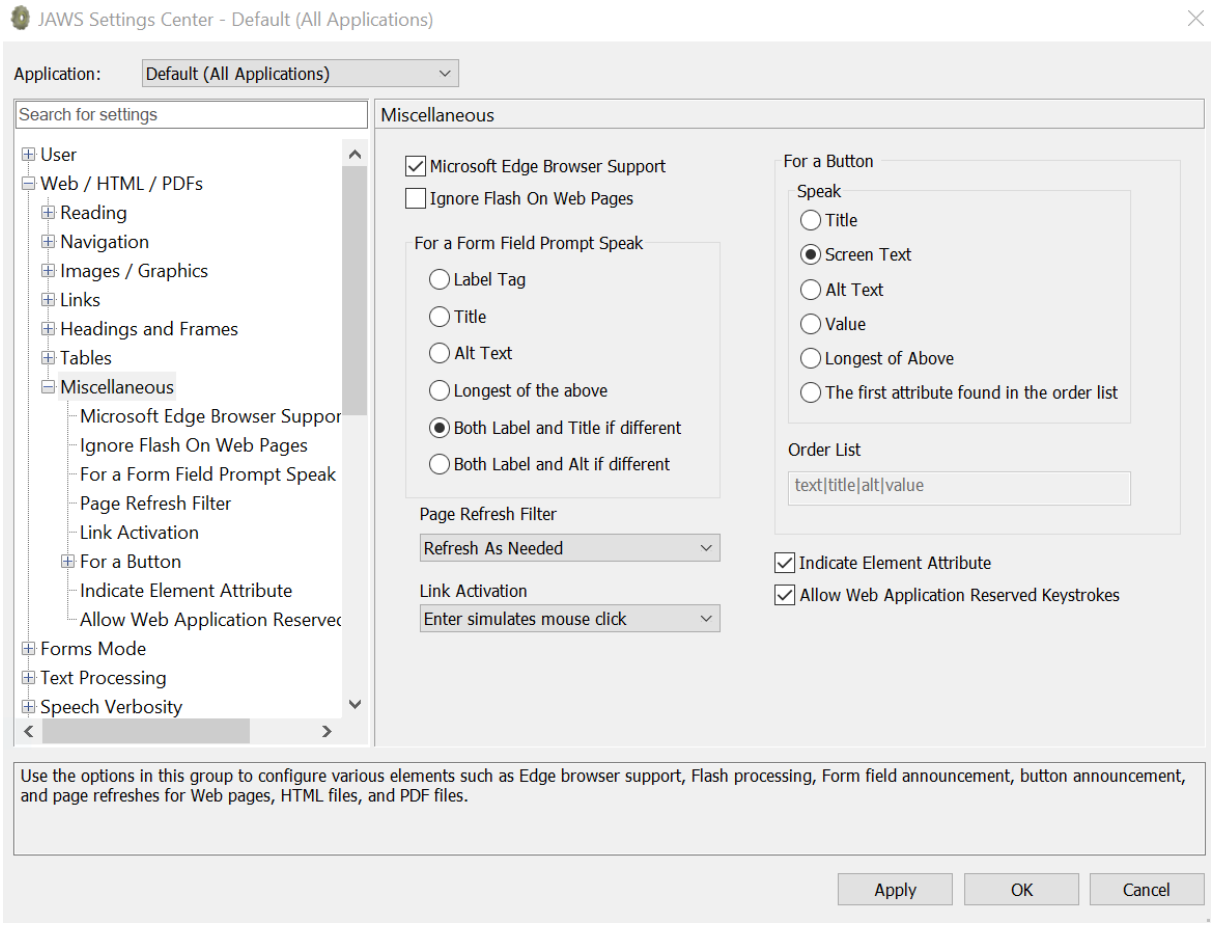
Configuring JAWS for Advantage Applications

There are some settings that must be configured for JAWS to work correctly with Advantage applications.

Reading Form Field Title

These settings configure how JAWS reads form fields. For Advantage, configure JAWS to 'Use Both Label and Title If Different' for Form Fields. In Advantage, some fields, such as Required Fields, would have special key words in the title attribute that does not show in the label. Hence, configuring JAWS to access the title attribute would enable it to provide the user with the relevant information from the title attribute.

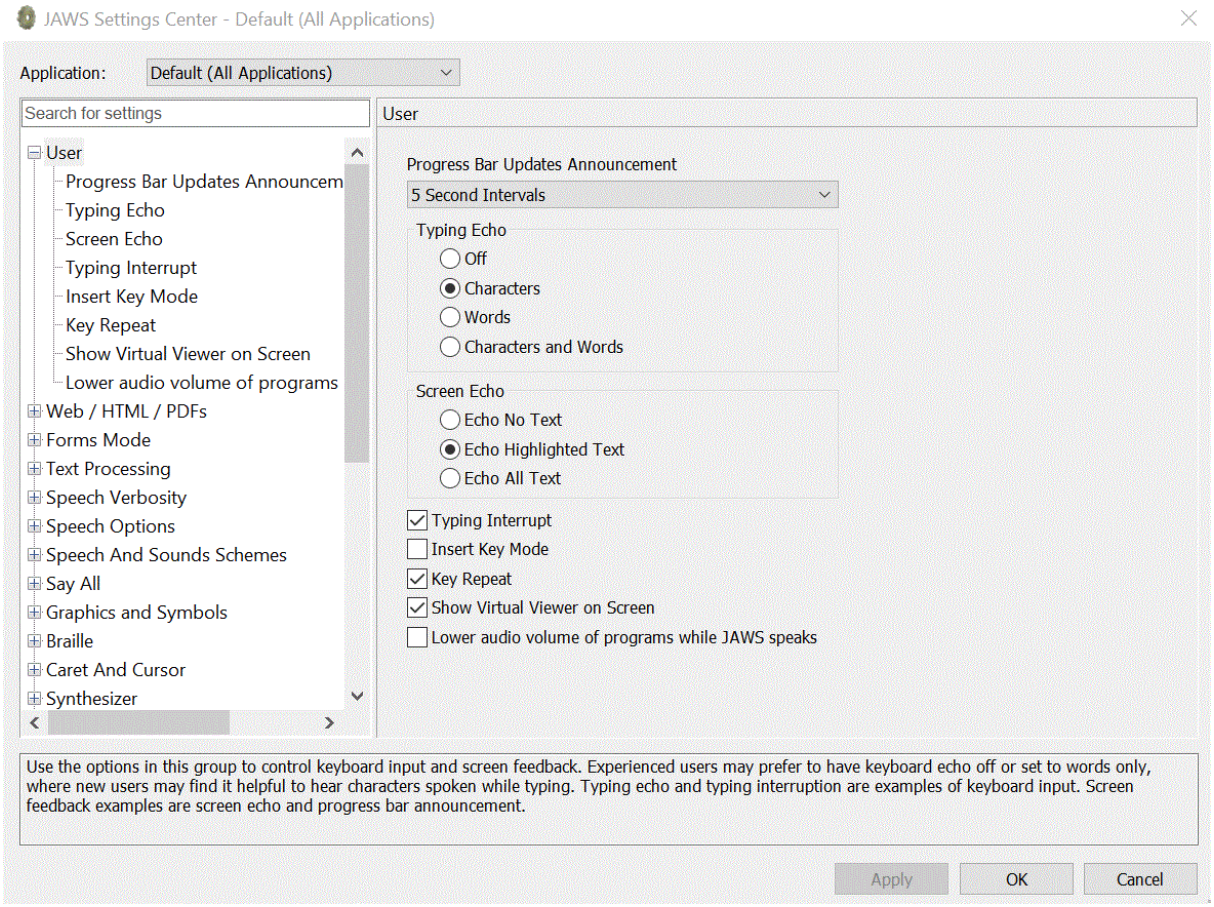
1. From the JAWS menu, select Utilities > Setting Center > Web / HTML / PDFs > Miscellaneous.
2. In the radio button group **For a Form Field Prompt Speak** options, select *Both Label and Title If Different*.
3. Select **Apply**.



Announce Progress Message While Loading Page

The following setup on JAWS enables announcing of progress messages when a page is being loaded to be opened and when the loading progression bar appears (mostly for transaction pages).

1. From the JAWS menu, select **Utilities** > **Setting Center** > **User**.
2. Do the following settings as shown in the following screen shot and then select **Apply**.



Working with Windows Eyes in Advantage Applications

Window-Eyes allows people to surf the Internet, send and receive e-mails, and to use typical business productivity software. Window Eyes has a Browse Mode, which if turned on, allows Window-Eyes to work more closely with applications without needing to access its OSM (off screen model). Either the Browse Mode option in the General menu can be used or the Browse Mode Toggle hot key (CTRL-SHIFT-A by default) can be pressed. By default, Window-Eyes will automatically load Browse Mode whenever a new page is loaded. Browse Mode will automatically be turned on even if Browse Mode is off when a new page loads.

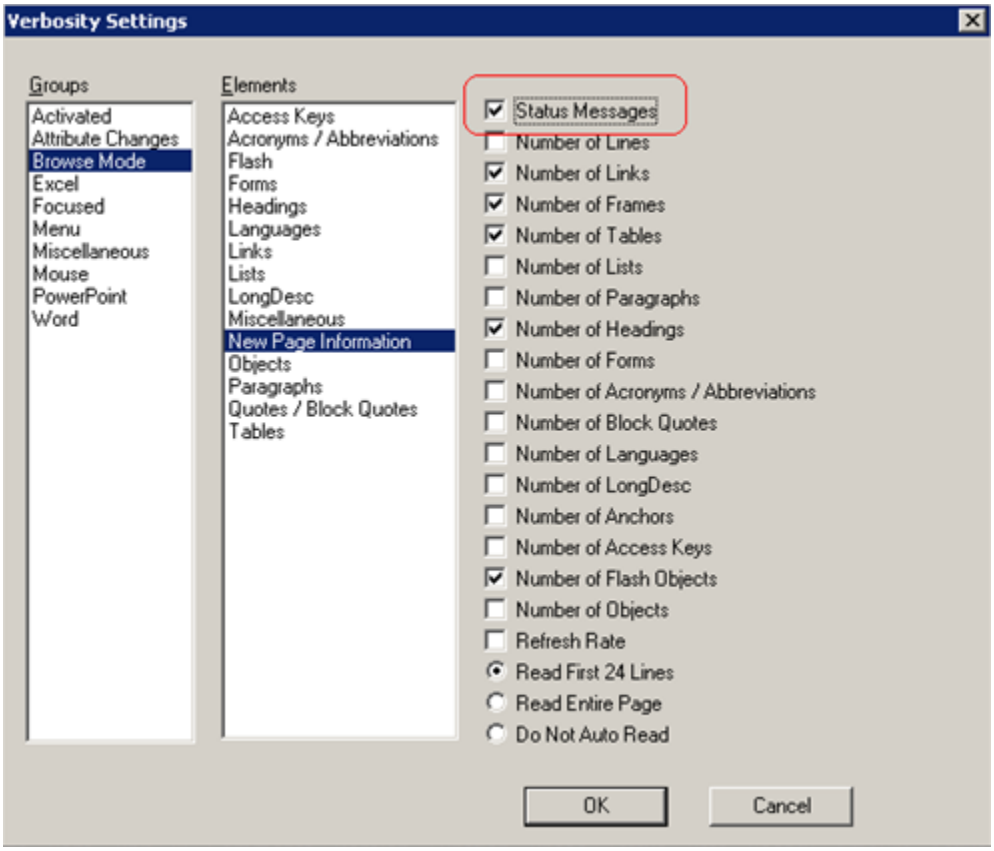
Configuring Windows Eyes for Advantage Applications

Announce Progress Messages While Loading Page

The following setup on Windows Eyes enables announcing of progress messages when a page is being loaded to be opened and when the loading progression bar appears (mostly for transaction pages).

1. Select the **Status Messages** check box for New Page Information (element) for Browse Mode (group) on the Verbosity Settings window.

2. Select **OK**.



Usage Tips

Refer to the following for usage tips for accessibility.

Reading Non-Editable Check box and Combo box

Non-editable check boxes and combo boxes are only read when Browse mode is ON for JAWS and Windows Eyes and they are not read when Browse mode is OFF. For JAWS use keystrokes ALT + UP ARROW (Say prior Sentence) and similarly ALT + DOWN ARROW (Say Next Sentence) to read them. Use the same keystrokes to read the accordion band on transaction pages.

Reading Error Messages

When a user selects the “View All” link on the error message bar then the error messages are displayed in tabular form. Refer to JAWS and Windows Eyes documentation to look up keystrokes that allows the user to navigate a table and its cells. Use those keystrokes to read the error messages which are in tabular form.

Listing Sub-menu Items on Menu of Transactions page with Jaws

The following steps explain how to list sub-menu items on Transaction pages using JAWS:

1. JAWS User can select a menu on a Transaction page by clicking the “Activate Link” for the menu on the Links List (Insert F7) dialog on a Transaction page.
2. The menu is displayed with the main menu items. Some of these main menu items have sub-menu items associated with it.
3. When a sub menu item is selected on Links List dialog, JAWS reads out that sub-menu items exist for it. In order to obtain the sub-menu items for a menu item on the Links List dialog, select the sub menu item and then click the “Move to Link”(and not Activate Link) on the Links list dialog. This will then list sub-menu items for the menu item below the menu item on Links List dialog on the next invocation.

Selecting Check box, Radio Button and Action Buttons with Windows Eyes

With Windows Eyes running, radio buttons, check boxes, and buttons can all be manipulated while either in or out of Browse Mode by using the SPACE BAR. Note: The Enter key should not be used for selecting them.

Application Navigation Setup

The following pages include setup for navigation with the Advantage application.

- Application List
- Application Page Registration
- Leaf Fields
- Context Sensitive Search

Application List

The Application List page displays the list of all applications in the system. A record is maintained for each application specifying information such as the unique ID assigned to the application, the name of the application, whether the selected application is active, and whether the application can stand-alone. The Stateless flag indicates if the application checks the system state when logging in. Stateless applications must also be stand-alone.

Application Page Registration

The Application Page Registration page is used to register all pages within the system. All pages must be registered here so that they can be accessed easily and the online help files can be located by the system. Note: The Enterprise Search Lite Indexer job must be run after making changes to this page, in order for the changes to appear as part of the Global Search. The job only selects records from this table that have the **Searchable?** check box selected.

› Field Information

- Page Code – A unique code that identifies a page in the system. This code can be used when searching by Page Code in the Global Search field.
- Category – The sub system such as, Procurement, Accounts Payable, Benefits Administration, and so forth to which the page belongs.
- Page Type – Indicates if the page is an Activity Folder, Transaction, Reference Table, Inquiry, or a Widget.
- Description – A description for the page. This description is displayed to the user when using Global Search or Advanced Search.
- UI Package – The name of the CGI Advantage Studio client application to which the page belongs. For example, the name of the application for the page that displays the Bank page is Accounts_Payable, hence this is the value entered in this field while setting up the entry for the Bank page.
- Application - This is the CGI Advantage application to which the page belongs. A drop-down list of all applications within the system is displayed. The administrator is responsible for selecting the application to which the page belongs. If no application is selected, the page is considered common and is available to all non stand-alone applications.

- Dest Page – The name of the Java class for this page. For example, the name of the Java class that displays the Bank page is pR_AP_BANK_ACCT_Generic, hence this is the value that would be entered in this field while setting up the entry for the Bank page.
- Dest Frameset – If a page is made up of multiple frames then this field contains the name of the Java class that makes up the frameset that contains the frames. In most cases this field is left blank in the system.
- Dest Frame – The name of the HTML frame to which the initial transition occurs if a page is made up of multiple frames. This field is only populated if the Dest Frameset field is populated.
- Target Frame – The name of the browser window or frame in which the page is displayed when a user clicks navigates to the page from Global Search or Advanced Search. Normally, this field has a value of Display to indicate that the page is displayed in the main display area.
- Help Sub Directory – The name of the directory under the main help folder where the HTML help file for this page is stored. The location of the main help folder is identified by the property MainHelpFilesLocation in the system's default help configurations file (adv__conf/help.conf) for generic purposes.
- Help File Name – The actual name of the HTML file that contains the help information for this page. This file is physically located under a help sub directory as discussed above.
- Initial Mode – The mode in which the page is displayed when the user accesses this page from Global Search or Advanced Search. The possible values are Query (the page is displayed in query mode), Browse (the page is displayed with data) and Add (a blank record is inserted).
- Order By – Indicates the order in which data is sorted on this page when a user accesses the page from the Page Search page. The order is specified in the form of a valid SQL ORDER BY clause without the key word ORDER BY, but can contain the key words ASC or DESC to indicate the direction in which the data must be ordered. For example, DOC_CD DESC indicates that the data has to be sorted by transaction code in descending order.
- Where Clause – Indicates the filter that is used to display data on this page when a user accesses the page from the Page Search page. The where clause is specified in the form of a valid SQL WHERE clause without the key word WHERE. For example, DOC_CD = 'PO indicates that the data must be filtered to only display Purchase Orders.
- Searchable? – Indicates if this page must be accessible from Global Search or Advanced Search. This is required as the information for certain pages is stored in the Application Page Catalog so that help information for this page can be retrieved, but these pages should not be accessed directly from Global Search or Advanced Search and should be accessed through a link on some other page.
- Organization Codes subsection – Pages can be secured by entering organizational elements information. For filtering based on organizational authority to be enabled, row level security must be enabled for the data object IN_PAGES. For more information on how to setup row level security for a data object please refer to the "Application Resources" topic in the Security Administration User Guide.

Leaf Fields

The Leaf Fields (LEAF) page allows you to set up and maintain leafing information for all of the application pages.

The following information can be set up on this page:

- UI Package – The name of the application.
- Page – The name of the page that you are leafing to/from.
- Record Source – The name of the attributes record source.
- Leaf Name – The logical name of the leaf column.
- Attribute Name – The name of the attribute/column/field to be used for leafing to a different page.
- Export Only – If selected, then only leafing out is possible from this page.
- Leaf Destination Usage – Indicates whether the attribute's value is required, optional, or not used to leaf.
- Not Used - if the Not Used selection is made, the Leaf Attribute is not used in the leaf action, but leafing will still occur. The default value is Not Used.
- Required - if the Required selection is made, the Leaf Attribute must be provided by the source page, otherwise leafing will not occur.
- Optional - if the Optional selection is made and the Leaf Attribute is provided by the source page, then the value provided is used to filter appropriate data on the destination page. Otherwise, if no value is provided, the Leaf Attribute is not used in the leaf action, but leafing will still occur.
- Full Key vs Partial Key Leafing – Both full key and partial key leafing is supported. For full key leafing, the full set of primary key(s) for the destination page must be provided by the source page in order for relevant data to be displayed. If a destination page requires two keys and the source page only provides one of the keys, then this leafing will not occur. For example, the primary key for the Title (TITL) page is the Title code (TITL_CD), and the primary key for the Sub Title (STTL) page is the Title code (TITL_CD) as well as the Sub-Title code (SUB_TITL_CD). Full key leafing provides leafing functionality from STTL to TITL, since STTL can provide the full key required by TITL. However, if you wish to leaf from TITL to STTL using only the Title code, full key leafing does not allow this to happen as STTL requires two keys to display matching records.

  For partial key leafing, the source page may provide only partial key information of the destination page. For example, the LEAF setup for the STTL table would be as follows. The key fields are Title and Sub-Title. Title is set up as Required, whereas Sub-Title is set up as Optional. This means that if the Sub-Title code is provided by the source page, then it is used, together with the Title code, to filter appropriate data on the STTL page to bring up only those records that match both the Title and Sub-Title code selected. If only the Title code is available and no Sub-Title code is available (such as when leafing from TITL to STTL), then only the Title code is used to perform the leafing. STTL displays all records containing the selected Title code.

- Leaf Attribute Prefix – Accommodates query objects on the target page. If a column exists on more than one table in a query object, specify the Leaf Attribute Prefix with the table name to avoid an ambiguous column error.
    - For example: If a query object is made of the EMPL and the EMPL_ASGNMT table and has an attribute (INTERNAL_EMPL_ID) that is present in both of the tables, Architecture needs to know form which table the column should be referenced. In this case, the Leaf Attribute Prefix can be given as either EMPL or EMPL_ASGNMT.
- Filter Method – Valid options for this field are Default or Equals.
- Filter Method – Valid options for this field are Default or Equals.
- Leaf Group/Entry Set – This is a required field that defaults to 0. This field is used to group Leaf entries together for the purposes of building a primary key to use for lookup to the target page for Show Description and Show Details. The Leaf Name and Group/Entry set combination must be unique within the UI Page, Package, and Record Source.

You can transition to the Context Sensitive Search page by clicking the **Context Sensitive Setup** Related Page link.

Context Sensitive Search

The Context Sensitive Search table and LEAF table are linked by the common Leaf Name field. There is a 1-to-many relationship from Context Sensitive Search to LEAF. This page determines the key fields required to do a lookup to the target page.

The following information is set up on this page:

- **Leaf Name** – This is a required field. This is the common field between the LEAF and Context Sensitive Search pages.
- **Target Page** – This is a required field. This field provides a pick to the Application Pages (APGS) table. This value is used to look up the key fields for the target page. It is used to select all LEAF entries where the Page is equal to Target Page and the Required for Leaf check box is selected.

You can transition to the Leaf Fields table by selecting the **Leaf** Related Page link.

Configurable Validation

The Configurable Validation (BORULE) page is available in Advantage Administration, Financial, Vendor Self Service and Human Resource Management applications. The Configurable Validation pages are maintained by a System Administrator to define validations/preconditions/rules specific to a transaction or a reference page. Edits defined on these pages will work as an additional validation for a specific reference page/transaction in addition to what is indicated at the application code level.

These edits are applied to business objects and are not page specific. For example, if in the Financial application a validation is added to the Purchase Order Header component (PO_DOC_HDR) then it is evaluated for each transaction (including clones) referring to the Purchase Order Header component.

The following key fields can be defined on this page:

- **Active** – Only validations that are marked active will be considered for evaluation.
- **Priority** – Defines the priority or sequence in which the validation will be evaluated. If there are more than one validation defined for a business object, then evaluation will be based on Priority. Records with a low number gets a higher precedence (that is, a record marked with Priority 1 will be evaluated before a record marked with Priority 2). If two records for the same business object are added with identical Priority, then the one added first takes precedence (based on Validation ID). Default Priority is set as 999.
- **Validation Type** – Indicates the type of record.
    - **Validation** – Records marked as a Validation are edits, which on evaluating (expression as part of Validation) as successful will issue an error message as configured.
    - **Precondition** – Records marked as a Precondition are just verification points and on evaluating (expression as part of Validation) as successful will continue processing the next set of records (Validation or Precondition). Otherwise, evaluation of additional validations will stop for the selected record.

    For example: If there are 5 validations defined for PO_DOC_HDR and the first (Priority) one has a Precondition for Transaction Department Code 010, then a PO transaction with any other Transaction Department Code will not process beyond this first rule.
- **Business Object** - Validation will be applied to the provided business object. This is the technical table name.

    For example, if applying a validation for the Financial application's Purchase Order Header component then use PO_DOC_HDR.
- **Trigger Action** - The Trigger Action specifies the system event validation associated with and activated. This is an optional field and is not applicable if the Validation Type is *Precondition*.

    Please note: For validations configured on transaction business objects, even if the Trigger Action selected is *Inserted*, *Updated*, or *Inserted or Updated*, the validation is still invoked on the Validate and Submit actions.
- **Validation** – Defines the actual edit or expression that needs to be evaluated to validate a record for the given business object. The expression used should evaluate as true or false.

For a Validation Type of Validation, the record will be rejected (to issue error message as configured), if the expression is evaluated as successful.

For a Validation Type of Precondition, the record will be accepted (to continue evaluating the rest of the Validations for the Business Object), if the expression is evaluated as successful.

The following fields capture the user feedback details associated with the validation. This section is only displayed if the Validation Type is Validation.

- **Message Code** - Allows you to select an existing Code defined in the system on the Messages table and associate it to the validation. When an existing Code is selected, the system will use the Severity and Text defined on the Messages table. This field cannot be populated if a value is provided in the Validation Severity and/or Validation Text field.
- **Validation Field** - Specifies the attribute name of the field that should be validated on the selected Business Object. This information will be used show a visual error indication on the field to the user when validation fails and the record is rejected.
- **Validation Severity** – This field allows you to provide a validation severity instead of choosing an existing Message Code. This field cannot be populated if a value is selected in the Message Code field. Validation severity works in conjunction with Validation Text to render a custom user feedback message when validation is not met.
- **Validation Text** – This field allows you to provide custom validation text instead of choosing an existing message code in the system. This field cannot be populated if a value is selected in the Message Code field. Validation Text works in conjunction with Validation Severity to render a custom user feedback message when validation is not met.

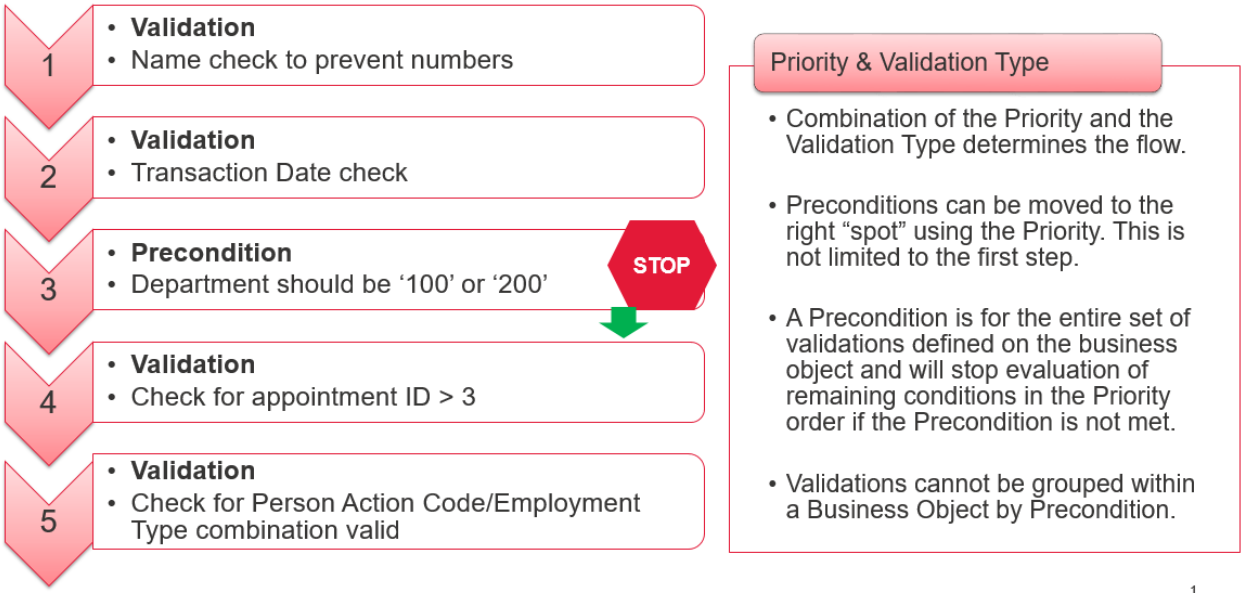
Refer to the following topics for additional information:

- Precondition vs Validation
- Validation Expression Syntax
- Writing a Configurable Validation
- Examples
- Deployment Considerations
- Frequently Asked Questions

Precondition vs Validation

The combination of the Validation Type and Priority selected on the Configurable Validation table determines the flow and controls execution of validations. Precondition comes in play when a large set of validations are defined on a business object and belongs to a common condition that can be checked as a precondition before evaluating validations. This option is very useful and avoids negative performance impact by allowing execution of validations only when necessary.

The following diagram depicts the control flow and use of preconditions.



Validation Expression Syntax

The Configurable Validation functionality implements the Apache JEXL expression evaluation library and expression syntax must comply with the JEXL rules. Though JEXL supports advanced dynamic scripting features, the following sections list basic operators, literals and special keywords supported by the Advantage application.

- Literals
- Operators
- Functions
- Keywords

Literals

Listed below are literals support by the Advantage application and can be used as part of the expression provided in the Validation field on the Configurable Validation table.

Item	Description
Array Literal	Use **square brackets** with one or more values separated by a comma. Example: [1, 2, 3] **Note:** It is always recommended to use all the entries in the array of same class for correct comparisons.

Item	Description
Big Decimal Literals	Use 1 or more digits from 0 to 9, followed by a decimal point and then one or more digits from 0 to 9 suffixed with **b** or **B**. Example: 442.0b 42.0B
Big Integer Literals	Use 1 or more digits from 0 to 9 suffixed with **h** or **H.** Examples: 42h or 42H
Boolean Literals	Use the word **true** or **false**. Examples: ACTIVE_FL == true or ACTIVE_FL == false
Double Literals	Use 1 or more digits from 0 to 9, followed by a decimal point and then one or more digits from 0 to 9 suffixed with **d** or **D**. Examples: 42.0d 42.0D
Float Literals	Use 1 or more digits from 0 to 9, followed by a decimal point and then one or more digits from 0 to 9, optionally followed by **f** or **F**. Examples: 42.0 42.0f
Integer Literals	Use 1 or more digits from 0 to 9. Examples: 42 100
Long Literals	Use 1 or more digits from 0 to 9 suffixed with **l** or **L**.

Item	Description
	Examples: 42l 42L
Map Literal	Use curly brackets ({ }) with one or more sets of **key**: **value** pairs separated by a comma (**,**). Example: { "one" : 1, "two" : 2, "three" : 3, "more": 4 } **Note:** It is always recommended to use all the values in the map of same class for correct comparisons.
Null Literal	Use the word **null**. Example: MY_COLUMN == null
Real literals - exponent support	Use standard Java exponent notation (that is, suffix the number with **e** or **E** followed by the sign **+** or **-** followed by one or more decimal digits). Examples: 42.0E-1D 42.0E+3B
Set Literal	Use curly brackets ({ }) with one or more values separated by a comma (**,**). Example: { "one" , “two”, "three"} **Note:** It is always recommended to use all the entries in the set of same class for correct comparisons.
String Literals	Use either the single quote (**'**) or double quotes (**"**). For example: **"Advantage"** and **'Advantage'** are equivalent.

Operators

Listed below are operators support by the Advantage application and can be used as part of the expression provided in the Validation field on the Configurable Validation table.

Operators	Description
Boolean and	Use **&&** or the word **and**. COLUMN_1 == null and COLUMN_2 != null COLUMN_1 == null && COLUMN_2 != null
Boolean or	Use **||** or the word **or**. Examples: COLUMN_1 == null or COLUMN_2 != null COLUMN_1 == null || COLUMN_2 != null
Boolean not	Use **!** or the word **not**. Examples: !(COLUM1_1 == null) not (COLUMN_1 == null)
Bitwise and	Use **&** operator. Example: 10 & 4
Bitwise or	Use **|** operator. Example: 3 | 4
Bitwise xor	Use **^** operator. Example: 33 ^ 44
Bitwise complement	Use **~** operator.

Operators	Description
	Example: ~33
Ternary conditional?:	Use standard JAVA style conditional operator condition? if_true **:** if_false. Example: val1 ? val1 : val2 and val1 ?: val2 are equivalent. **Note:** The condition evaluates to false when the value referred is an undefined variable or null.
Equality	Use **==** or the abbreviation **eq**. Examples: COLUMN_1 == COLUMN_2 COLUMN_1 eq COLUMN_2 **Note:** Comparing **null** to any **non-null** value evaluates to false.
Inequality	Use **!=** or the abbreviation **ne**. Examples: COLUMN_1 != COLUMN_2 COLUMN_1 ne COLUMN_2
Less Than	Use **<** or the abbreviation **lt**. Examples: COLUMN_1 < COLUMN_2 COLUMN_1 lt 10
Less Than Or Equal To	Use <= operator or the abbreviation **le**. Examples: COLUMN_1 <= COLUMN_2 COLUMN_1 le 10
Greater Than	Use **>** or the abbreviation **gt**.

Operators	Description
	Examples: COLUMN_1 > COLUMN_2 COLUMN_1 gt 10
Greater Than Or Equal To	Use **>=** or the abbreviation **ge**. Examples: COLUMN_1 >= COLUMN_2 COLUMN_1 ge 10
In or Match=~	Use **=~** operator to check that a string matches a regular expression. It also works like an “in” condition and checks if any collection contains a value or not. Examples: NAME =~ ‘.*\\d+.*’ TYPE =~ [‘MAX’,’MIN’, ‘AVG’]
Not-In or Not-Match!~	Use **!~** operator to check that a string does not match a regular expression. It also works like “not in” condition and checks if any collection contains a value or not. Examples: NAME !~ ‘.*\\d+.*’ TYPE !~ [‘MAX’,’MIN’, ‘AVG’]
Starts With=^	Use **=^** operator. Example: To match a Department code that starts with letter ‘C’ DEPT_CD =^ ‘C’
Not Starts With!^	Use **!^** operator. Example: To match a Department code that does not starts with letter ‘C DEPT_CD !^ ‘C’

Operators	Description
Ends With =$	Use **=$** operator. Example: To match a Department code that end with letter ‘C’ DEPT_CD =$ ‘C’
Not Ends With!$	Use **!$** operator. Example: To match a Department code that does not end with letter ‘C’ DEPT_CD !$ ‘C’
Addition	Use **+**. Example: AMOUNT_1 + AMOUNT_2
Subtraction	Use **-**. Example: AMOUNT_1 - AMOUNT_2
Multiplication	Use *****. Examples: AMOUNT_1 * AMOUNT_2 AMOUNT_1 * 10
Division	Use **/** or **div**. Examples: AMOUNT_1 / AMOUNT_2 AMOUNT_1 div 10
Modulus (or remainder)	Use **%** or **mod**. Examples:

Operators	Description
	AMOUNT_1 mod 2 AMOUNT_1 % 2
Side-effect operators	The following side-effect forms supported. • += • -= • *= • /= • %= • &= • |= • ^=
Negation	Use **-** operator. Example: AMOUNT_1 > -12
Array access	Use **square brackets** or a **dotted numeral**. Example: Refer to an item in a collection: VALUES[0] or VALUES.0
Map access	Use **square brackets.** Examples: Refer to a map entry using index: MAP_VALUES[0] or MAP_VALUES.0 Refer to a map entry using key: MAP_VALUES['KEY']

Functions

Listed below are the default functions supported by the Advantage application and can be used as part of the expression provided in the Validation field on the Configurable Validation table to facilitate readable expressions.

Functions	Description
empty	Evaluates whether argument passed is 'empty'. Returns **true** when the argument is: • null • An empty string • An array of length zero • A collection of size zero • An empty map Examples: empty(FUND_CD) !empty(TRANS_AMT)
Size	Evaluates the 'size' of the argument passed. This method returns: 1. 0 if the argument is null 2. Length of an array 3. Length of a string 4. Size of a Collection 5. Size of a Map 6. The result of calling a method 'public int size()' defined by the argument class 7. This returns 0 in other cases. Example: size(CODE)

Keywords

Listed below are keywords are supported by the Advantage application and can be used as part of the expression provided in the Validation field on the Configurable Validation table.

Keyword	Description
applicationDate	Represents the date that the Advantage application is using. This is the date value configured on the Application Parameter table. For example, the following expression compares the effective date to the application date. EFFECTIVE_DT > applicationDate
applicationDateTime	Represents the date and time that the Advantage application is using. This is the date value configured on the Application Parameter table plus the server time value at the time the value is requested. For example, the following expression compares the Effective Date to the Application Date Time. UPDATE_DT > applicationDateTime
systemDate	Represents the Server Date on which the Advantage application is running. For example, the following expression compares the Batch End Date with the Server Date. BATCH_END_DT > systemDate
systemDateTime	Represents the Server Date Time on which the Advantage application is running. For example, the following expression compares the Batch End Date to the Server Date Time. BATCH_END_DT > systemDateTime
systemTime	Represents the current time. For example, the following expression compares the time to the server time. END_TM < systemTime

Writing a Configurable Validation

Advantage supports simple field references and more advanced scripting options within the expression that allow sites to utilize system capabilities to the full extent. Please note that implementors should be familiar with Advantage Framework and Application API to reference JAVA methods inside Advantage Framework classes. The following section explains different types of conventions supported within the

expression in the Validation field on the Configurable Validation page that helps implementors to define readable and maintainable expressions.

- Field References
- Constant Values
- JAVA Methods
- Business Object Instance Methods
- Advantage Helper Class Methods
- External JAVA Helper Classes Support

Field References

You can reference fields on the Business Object being validated simply by the name within the validation expression in the Validation field on the Configurable Validation page. The Advantage application retrieves the field values and makes them available to the validation framework at run time.

For example, if a business object has a field named FUND_CD, then the rule expression should be able to refer to the field by name as shown below.

```
FUND_CD == '010'
```

or

```
FUND_CD != null
```

To reference an old value of a field, simply add the ".old" prefix before the field name. To refer to the old Fund Code value in previous example, the expression should be:

```
old.FUND_CD == '020' and FUND_CD == '010'
```

Field values can be referenced via Advantage Framework provided JAVA methods using the "this" prefix as shown below and achieve the same result as above. It is highly recommended to use field references by name where possible to create readable and maintainable expressions.

```
this.getFUND_CD() == '010'
```

or

```
this.getData('FUND_CD').getString() == '010'
```

Constant Values

You can reference constants defined inside the Advantage classes within the expressions for comparisons. This will make expressions more readable. To use a constant in the expression in the Validation field on the Configurable Validation page, you must be aware of JAVA class implementation and values mapped to constant labels inside the JAVA class.

To compare a field value with a constant, the following convention can be used within the expression in the Validation field on the Configurable Validation page:

```
ACTIVE_FL == CVL_YES_NOImpl.YES
```

or

```
DOC_ACTN_CD == AMSCommonConstants.VALIDATE
```

JAVA Methods

You can refer to all public allowable JAVA methods inside the Business Object instance and public static methods in supported Advantage helper classes within the expression provided in the Validation field on the Configurable Validation page. This may have an adverse impact when certain methods referenced within the expression changing the state of data being validated while executing the expression. To prevent side effects, the Advantage application enforces additional checks and allows only certain method patterns in the expression assuming those methods are built using JAVA standards and are safe to use.

The Advantage application allows methods starting with the following patterns from Advantage classes to refer within the validation expression. This limitation is specifically for Advantage Helper:

- get
- has
- is
- does
- check
- chk
- exist
- strIsEmpty
- strEqual
- strInsensitiveEqual
- addDays
- addMonths
- addYears

By default, Advantage supports reference of public static methods from all the Advantage helper classes inside the advantage.jar module. In addition to that, the following internal and external classes outside the advantage.jar module are supported to utilize extended helper functions within the expression.

Class Name	Internal / External Package
AMSCommonConstants	com.amsinc.gems.adv.common

AMSSecurity	com.amsinc.gems.adv.common
AMSSPARConstants	com.amsinc.gems.adv.common
StringUtils	org.apache.commons.lang3

Business Object Instance Methods

To refer to business object instance JAVA methods within the expression in the Validation field on the Configurable Validation page, simply use the method name with the “this” prefix.

For example, if the business object being validated has a method named getRunDate() that needs to be referenced in the validation expression, then the expression could be written as shown below:

```
this.getRunDate() < applicationDate
```

or

```
this.getRunDate() < EFFECTIVE_DT
```

Advantage Helper Class Methods

To refer to Advantage helper class methods (mostly static methods) in the Validation field on the Configurable Validation page, simply use method invocation by providing required parameters.

For example, if advancing Effective Date by 5 days to compare with the Application Date or if advancing Effective Date by 2 years to compare with the End Date in validation:

```
AMSUtil.addDays( EFFECTIVE_DT, 5 ) < applicationDate
```

or

```
AMSUtil.addYears( EFFECTIVE_DT, 2 ) < END_DT
```

External JAVA Helper Classes Support

To refer to external Java helper class methods (like StringUtils from Apache commons library) in the Validation field on the Configurable Validation page, simply use method invocation by providing required parameters.

For example, to check if String field is populated or blank in validation:

```
StringUtils.isBlank( this.getFUND_CD() )
```

or

```
StringUtils.isBlank( FUND_NM )
```

Examples

The following two scenarios provide different ways to formulate the validation expression in the Validation field on the Configurable Validation page. Given the ability to reference JAVA classes and methods within the expression, there can be several ways an expression can be written to achieve the same results. Rule designer can be creative in choosing optimal techniques to build efficient expressions.

Scenario 1

Prevent numbers in a Code field that accepts both letters and numbers in baseline:

Validation can be written using StringUtils helper methods.

```
StringUtils.containsAny(CODE, '0','1','2','3','4','5','6','7','8','9')
```

Validation can be written using Regex matching.

```
CODE=~ '.*\\d+.*'
```

Validation can be written using the Advantage helper method if one is available.

```
AdvantageUtils.hasNumbers(CODE)
```

Scenario 2

Implement a complex validation on an ESMT transaction where the functional requirements are: If the personnel Action code entered is 29, the 4th digit in the payroll number must be equal to 3 and pay class code must be equal to MONTH and table driven pay must be equal to N.

```
PERS_ACTN_CD == '29' and StringUtils.substring(PAYROLL_NO_CD, 3, 4) != 3
and PAY_CLS_CD != 'MONTH' and TBL_DRV_PY_RT_FL != 'N'
```

Deployment and Considerations

The Configurable Validation feature is a System Administrator function and needs careful attention to develop, test and deploy configurations to production. Configurable Validations are stored in the R_BO_RULE_CNFG database table in each Advantage product schema. It is highly recommended that implementors develop and test validations in a pre-production environment and once satisfied with the changes, deploy configurations to production environments during down time.

This feature can be controlled by an application feature flag with two properties that allow sites to enable/disable the feature as needed.

```
tableBasedConstraints

{

   enabled=false

   flushCacheOnConfigRuleUpdates=false

}
```

To enable this feature in an environment, set enabled=true. If a site is using this feature, the value must be set to true in all environments including production. After enabling the feature, the application recognizes configured validations and consumes during system processing.

flushCacheOnConfigRuleUpdates is applicable ONLY to test environments and enables the capability to immediately reflect the changes made to configurable validation records without a need for an application bounce. This property, when enabled, clears cached validation records and reloads updated configurable validation record(s). In a production environment, flushCacheOnConfigRuleUpdates must remain false all the time.

Sites implementing multiple Advantage products need to apply the above settings and considerations in each product environment.

Frequently Asked Questions

The following is a list of frequently asked questions and answers in regards to the Configurable Validation functionality.

- Are validations supported on all type of pages/tables?

  Yes, configurable validations are applied at the business object level and are not necessarily required to map to a physical table. For example, validations supported on mediation and custom data connector business objects that are not bound to a physical table and utilizes underlying business object processing events.

- Can I specify when to trigger the validation?

  Yes, you can choose an appropriate system event using Trigger Action options available on the validation record. Advantage will activate validation when the associated system event is executed.

- Is there a required syntax to define validation expression?

  Yes, Advantage uses a third party library, Apache JEXL, for expression evaluation and syntax must comply with JEXL rules.

- Can I use Configurable Validation to enforce valid or invalid combinations?

  Yes, it is possible; however, we do not recommend using this feature for enforcing valid or invalid combinations.

- Can I use complex validations with multiple fields?

  Yes, there is no limitation. Advantage supports any number of fields (of referenced business object) within the expression including "added" fields created via extensibility.

- What happens if configurable validation is added for a field that already has a baseline validation?

  The primary purpose of Configurable Validation is to provide the ability to make Advantage more restrictive, not to replace/override baseline validation. In this scenario, both validations are executed to maintain data integrity.

- Can I use Configurable Validation to make a field required?

  Absolutely; however, please keep in mind that an external rule engine comes with added overhead. Therefore, we recommend using data model extensions for this kind of

boilerplate type of validation so they will be enforced efficiently using Advantage native framework. This can be achieved using a simple data model extension by simply flipping the Required flag on the field to true.

- Can I use Configurable Validation to enforce a condition only on the page but not during an offline batch process?

  No. Configurable Validation is applied at the data model level and integrated into business object processing framework. Validation is enforced for both online and offline activity.

Configurable Formula

The Configurable Formula (BOFRMLA) page is available in the Administration, Financial, Vendor Self Service, and Human Resource Management areas of Advantage. The Configurable Formula page is used to define additional attribute-level formulas on the business object. Formulas defined on the Configurable Formula page are evaluated during business object processing in addition to, but not replacing/overriding those formulas coming from native metadata and data object extensions. The formulas are applied to attributes for a specific business object. For example, if in the Financial application a formula is added to the DOC_CREAT_DT attribute on the Purchase Order Header component (PO_DOC_HDR) then it is also defined for each transaction (including clones) referring to the Purchase Order Header component.

- Field Information

Field	Required?	Description
Formula ID	Required	This field defaults to give each record a unique identifier.
Name	Required	A descriptive name of the formula to allow for a quick reference to the formula.
Active	Required	Only records marked as active will be considered during business object processing.
Business Object	Required	This is the database table name where the formula applies. For example, if applying a formula to a field on the Purchase Order Header component then use PO_DOC_HDR.
Priority	Required	Indicates the execution order of formula records.
Description	Optional	A lengthy description to record any details about the formula, reason for the formula, etc.
Trigger Action	Required	Indicates the action on the specified business object that will trigger the formula. Carefully, select the action to match the type of business object, as not all actions are available to all business objects.
Formula Field	Required	The database field name on the selected business object for the formula.
Script	Required	(Hidden) For future CGI use. Defaulted to *True*.

Formula	Required	The formula used during business object processing. For example, if changing the date used to set the Created On (DOC_CREA_DT) from the system (server date) to the current application date as defined on the Application Parameter, then application date is entered.

Refer to the following topics for additional information:

- What is Supported?
- Formula Expression Syntax
- Writing a Configurable Formula
- Examples
- Deployment and Considerations
- Frequently Asked Questions

What is Supported

Advantage supports configurable formulas only on stored data fields. The formula can be applied on a baseline field or an “added” field created via extensibility. The formula expression can refer to baseline and “added” fields in the formula expression to calculate the value of another field on the business object.

The configurable formula functionality is not intended to override/replace baseline formulas. Advantage will not allow configurable formula on a field if there is already a formula defined in baseline using Derivation Types Sum, Count, Parent Replicate, or Formula. This will avoid inconsistent application behavior with multiple formulas applied on the same field.

One exception to that is a configurable formula can be used to override a field default value behavior defined in the baseline.

Formula Expression Syntax

The Configurable Formula functionality was built on same syntax used by Configurable Validation, refer to the “Validation Expression Syntax” topic under Configurable Validation for more information. In addition to the information in that topic, Configurable Formula functionality supports advanced scripting capabilities with JAVA style coding within the formula expression.

The following sections describe additional syntax and capabilities supported by the Configurable Formula functionality.

- Language Elements
- Conditional Statements

Language Elements

The following language elements are supported by the Formula field on the Configurable Formula page.

Item	Description
Comments	Use ## or **//** to define a single line comment within the expression. Use **/*…*/** for multi-line comment. **Examples**: ## Doing something // Run for loop **/*** Multi-line Comment */
Identifiers/Variables	Must use **a-z**, **A-Z**, _ or **$.** Can be followed by **0-9**, **a-z**, **A-Z**, _ or **$**. **Examples**: var1, myValue, **Notes:** • Variable names are case-sensitive. For example, MyValue and myValue are different variables. • The following keywords are reserved and cannot be used in variable names: or, and, eq, ne, lt, gt, le, ge, div, mod, not, null, true, false, new, var, and return.
Local Variables	Use keyword **var** followed by the variable name, as per the standards. **Examples**: • Basic Declaration: **var value;** • With Assignment: **var value = 25;** Local variables scope is the entire scope and they take precedence on resolution over contextual variables.
Statements	A statement can be the empty statement, the semicolon (;), block, or an expression. Statements are optionally terminated with a semicolon.
Block	A block is multiple statements inside curly braces.

Item	Description
Assignment	Assign the variable value with a literal or use a context as initial resolver.
Method Calls	Use standard JAVA standard to call the method of an object.

Conditional Statements

The following conditional statements are supported by the Formula field on the Configurable Formula page.

Operator	Description

If	Use classic **if/else** statement.
	Example:
	To evaluate a field value conditionally comparing another field value.
	if (RATING > 30){
	'High Rating';
	}else if (RATING > 20){
	'Medium Rating';
	}else{
	'Low Rating';
	}
	RATING in above scenario is a field on the referenced business object.

for	Use classic **for** statement to loop through items of a Collection (Array, Map, Iterator or Enumerator).
	Example:
	To sum a field value on parent table using a child collection:
	var sum = 0.00;
	for (ChildClass child : this.getChildren(){
	sum = sum + child.getAMOUNT()

Operator	Description
	} return sum In the above scenario, getChildren() is an existing JAVA method on the referenced business object and returns an array of child objects.

while	Use the classic **while** statement to loop through items of a Collection until a condition is satisfied (Array, Map, Iterator or Enumerator).
	Example:
	To evaluate a field value on a transaction header component looping through commodity records:
	var commodities = this.getCommodityLines();
	while (commodities.hasMoreElements()){
	if (StringUtils.isNotEmpty(commodities.getCOMM_TYP())
	&& StringUtils.equals(
	commodities.getCOMM_TYP(),'SPECIAL')){
	return true;
	}
	}
	In the above scenario, getCommdityLines() is an existing JAVA method on the referenced business object.

Continue	Skips to the next iteration within loops.
break	Breaks from a loop.

Writing a Configurable Formula

Configurable Formula functionality uses the same expression syntax and conventions supported in Configurable Validations. Please refer to the "Writing a Configurable Validation" topic for various conventions supported by the system. In addition, formula expression can include simple JAVA style if/else and for/while loop statements to run logic and derive a final field value.

Formula expression is a collection of statements and when multiple statements are included in the expression, the result evaluated from the last statement execution will be used as the field value where the formula is configured.

Examples:

- If a business object has RATE, QUANTITY, DISCOUNT, and FINAL_AMOUNT fields and formula defined on FINAL_AMOUNT as shown below, it is considered a single statement expression. In the following example, the value derived by evaluating the expression will be set on the formula field.

  (RATE * QUANTITY) – (RATE * QUANTITY * DISCOUNT)

- If a business object has RATE, QUANTITY, and FINAL_AMOUNT fields and formula defined on FINAL_AMOUNT to calculate an amount using variable discount it is considered a multiple statement expression as shown below.

  If (QUANTITY > 500) {

  (RATE * QUANTITY) - (RATE * QUANTITY * 0.10)

  } else if (QUANTITY > 250) {

  (RATE * QUANTITY) - (RATE * QUANTITY * 0.08)

  } else {

  (RATE * QUANTITY) - (RATE * QUANTITY * 0.05)

  }

- In the above example, the field value is evaluated to the last statement executed based on the QUANTITY value. In multiple statement scenarios, you can optionally use a JAVA style **return** keyword with a semi-colon to break the expression execution. The previous expression can be written as shown below and works perfectly fine.

  If (QUANTITY > 500) {

  return (RATE * QUANTITY) - (RATE * QUANTITY * 0.10);

  } else if (QUANTITY > 250) {

  return (RATE * QUANTITY) - (RATE * QUANTITY * 0.08);

  } else {

  return (RATE * QUANTITY) - (RATE * QUANTITY * 0.05);

  }

Considerations

When configuring the Formula on a field, it is very important to understand and make sure that the data type of the field and the value evaluated from the formula expression are of the same class. Not paying attention to that will result in non-compatible class cast errors.

Configurable formula functionality does not handle implicit ordered execution of formulas when calculations configured on multiple fields of a business object are inter-dependent. To mitigate the dependency, formula designers must use formula priority value to dictate the order of execution to ensure field values are calculated and made available for subsequent dependent calculations.

For example, consider formulas configured on AMOUNT_1 and AMOUNT_2 fields of a business object. The formula on AMOUNT_2 field references the AMOUNT_1 field in the calculation. In this scenario, the formula designer should use a higher priority on the AMOUNT_1 formula compared to the AMOUNT_2 formula (for example, a Priority of 1 is a higher Priority value than 2). This will ensure the AMOUNT_1 field value is calculated before the formula on the AMOUNT_2 field is invoked.

Examples

The following section lists a few scenarios and formula expression examples that can be used in the Formula field on the Configurable Formula page.

Scenario 1

Default a user field to the current user adding/updating a record. You can use the following expression in the Formula field.

```
this.getUser()
```

Scenario 2

Default a date field to the current application date. You can use one of the below expressions in the Formula field.

Simply specify the supported key word.

```
applicationDate
```

You can also use the Utility method to achieve the same result.

```
AMSUtil.getApplControlDate(this.getSession())
```

Scenario 3

Calculate an amount field value using simple arithmetic operations based on other fields on the business object. You can use an expression with simple field references.

```
LN_AM - RTG_LN_AM - DISC_LN_AM + PNLTY_INTR_LN_AM - WHLD_LN_AM -
INCT_LN_AM - DFLT_INT_FEE_AM - SUPP_INT_FEE_AM - CNTRC_WHLD_AM
```

Scenario 4

Calculate the Posting Type field value conditionally on a transaction using the if/else condition with simple field references and Advantage Helper class combination.

```
if (DOC_CD ==  'IPO' OR
R_GEN_DOC_CTRLImpl.getDocSubTyp(this.getSession(), DOC_CD) == 'IRQ')

{

     if ( EVNT_CAT_ID =~ ['IP1', 'IP2', 'IP3', 'IP4', 'IP5', 'IP6',
'IP7', 'IP8', 'IP9', 'IP10'])
```

```
        {
           return 'B'
         } else {
           return 'A'
        }
    }
```

Deployment and Considerations

The Configurable Formula functionality is a System Administrator function and needs careful attention to develop, test, and deploy configurations to production. Configurable Formulas are stored in the R_BO_FORMULA_CNFG database table in each Advantage product schema. It is highly recommended that implementers develop and test formulas in a pre-production environment and once satisfied with the changes, deploy configurations to production environments during down time.

This feature can be controlled by an application feature flag with two properties that allow sites to enable/disable the feature as needed.

```
configurableFormula
{
   enabled=false
   flushCacheOnConfigFormulaUpdates=false
}
```

To enable this feature in an environment, set enabled to *true*. If a site is using this feature, the value must be set to *true* in all environments including production. After enabling the feature, the application recognizes configured formulas and consumes during system processing.

flushCacheOnConfigFormulaUpdates is applicable ONLY to test environments and enables the capability to immediately reflect the changes made to configurable formula records without a need for an application bounce. This property, when enabled, clears cached formula records and reloads updated configurable formula record(s). In a production environment, flushCacheOnConfigFormulaUpdates must remain false all the time.

Sites implementing multiple Advantage products need to apply the above settings and considerations in each product environment.

Frequently Asked Questions

The following is a list of frequently asked questions and answers in regards to the Configurable Formula functionality.

- Are formulas supported on all type of page/table fields?

  No, Configurable Formulas are only supported on persisted fields defined on the business object.

- Can I specify when to trigger the formula?

  Yes, you can choose an appropriate system event using the Trigger Action options available on the formula record. Advantage will activate the formula when the associated system event is executed.

- Is there a required syntax to define formula expression?

  Yes, Advantage uses a third party library, Apache JEXL, for expression evaluation and syntax must comply with JEXL rules. Please refer to the Formula Expression Syntax topic for additional details.

- Can I use a complex formula with multiple fields referenced in the formula expression?

  Yes, there is no limitation. Advantage supports any number of fields (of referenced business object) within the expression including baseline and "added" fields created via extensibility. In addition, Advantage supports simple JAVA style conditional elements to run logic and calculate field value. Please refer to the Formula Expression Syntax topic for additional details.

- What happens if a Configurable Formula is added for a field that already has a baseline formula?

  The system will not allow you to create a Configurable Formula on a field if there is already a formula defined in baseline using Derivation Type Sum, Count, Parent Replicate, or Formula.

- Can I override a default value defined on a field in baseline?

  Yes, a Configurable Formula can be used to override a field’s default value to meet site requirements.

- Can I use Configurable Formula on a field and restrict user updates?

  No. The ability to edit a field is not controlled by Configurable Formula and you need to use other extensibility techniques to make the field read-only.

- Can I use Configurable Formula to calculate a field value only on the page from online data entry but not during an offline batch process?

  No. Configurable Formula is applied at the data model level and is integrated into business object processing framework. The formula is evaluated from both online and offline activity.

Configurable Inquiries

Configurable Inquiries provide easy and quick access to online information within an Advantage application. Configurable Inquiries also provide the flexibility to save and share the inquiry, as well as expand the reporting capability to users who do not have access to create and configure reports. Refer to the “Configurable Inquiries” topic in the *CGI Advantage Page/Table User Guide* for information on creating a configurable inquiry, accessing an inquiry you created or that was shared with you, and executing a configurable inquiry. Refer to the “Inquiry Area” topic in this guide for information on the Inquiry Area setup table, which provides input for the Inquiry Area field when creating a configurable inquiry.

Configurable Inquires provide a seamless integration between the Advantage application with Insight, based on Insight Inquiry Area name and Business View. Integration with Insight would be an extension to already available Insight tools. Configurable Inquires provide a better user experience for the user, which allows you to configure the search filters, grid columns, visualization fields, and formatting setup based on the Insight catalogs. All naming conventions are as per the Advantage standards.

The following pages exist online in the Advantage application:

- **Inquiry Area (CIINQAR):** This page maintains the Insight Area mapping details such as the Inquiry Area Name, Business View, Inquiry Page Name, and Target Qualified Name.
- **Create Inquiry (CREATCI):** This wizard allows you to create new inquires based on the Inquiry Area. Authorized users can configure the search filters, columns, visualization, and format settings. Refer to the “Create Inquiry” topic in the *CGI Advantage Page/Table User Guide* for more information.
- **My Inquiries (CIMYINQ):** All inquiries created and shared with the user can be accessed from this particular page. Refer to the “My Inquiries” topic in the *CGI Advantage Page/Table User Guide* for more information.

Refer to the following topics in this guide for required setup:

- Inquiry Area
- Configuration Setup
- Reporting User Setup

Inquiry Area

The Inquiry Area (CIINQAR) reference page is used to define entries related to the Advantage Insight Catalog (model), Business View (Perspective) and Inquiry Pages that are used for creating Configurable Inquiries.

Field Name	Field Description
Inquiry Area Name	The Advantage Insight reporting catalog (model) name to which the user is connecting for creating inquiries (for example, FIN – Accounts Payable, HRM – Personnel).

Business View	The sub-set of a catalog in which you are making an inquiry (for example, Receivables and Cash Receipts are some of the Business Views defined under the FIN - Accounts Receivables catalog).
Inquiry Page Name	The Inquiry page name defined on application(s) related to the specific inquiry.
Target Qualified Name	It identifies the target inquiry page of an application.

Configuration Setup

The Configurable Inquiries Feature (config_inquiry.conf) file includes an inquiryconfig parameter with the following properties that must be set up in order to use the configurable inquiries functionality.

Parameter	Default Value	Parameter Description
uri		The URI of the Advantage Insight Server.
maxDataFetchSizeInBytes	5242880	Provides a size limit of response data from Insight for each of the Advantage applications request.
insightUser	SSASTest	Insight Server User Name (Different from the Advantage Insight Application User)
insightPassword	<SSASITestInsightPassword>	Insight Server password associated to the user (Different from the Advantage Insight Application User - Password)
minSearchFieldsLimit		Indicates the minimum number of search fields that must be added at the time of creating or saving the inquiry. If a value is not provided it defaults to *3*.
maxSearchFieldsLimit		Indicates the maximum number of search fields that can be added at the time of creating or saving the inquiry. If a value is not provided it defaults to *50*.
minGridFieldsLimit		Indicates the minimum number of columns that must be added at the time of creating or saving the inquiry. If a value is not provided it defaults to *1*.

maxGridFieldsLimit		Indicates the maximum number columns that can be added at the time of creating or saving the inquiry. If a value is not provided it defaults to *50*.

The Insight configuration details can be found in the *CGI Advantage Insight Third Party Software Installation and Configuration Guide*.

Reporting User Setup

When utilizing the Configurable Inquiry functionality, a User Mapping table needs to be populated with the Advantage User Name mapped with the Insight Application User Name, so that all the security related aspects of the Insight users are included while fetching the data from Insight.

Reporting user details can be populated from the Reporting User field on the User Information (SCUSER) page, which then populates the BI_USER_MAPPING table and the same would happen while a new user is being created.

Sites can populate the values on the User Information (SCUSER) page so that the BI_USER_MAPPING table is populated with the appropriate mappings.

Table Name: R_SC_USER_INFO

REPORTING_USER	A formula derived value from the REPORTING_USER column of the BI_USER_MAPPING table.

Table Name: BI_USER_MAPPING

USER_ID	Advantage Application User Id
REPORTING_USER	Advantage Insight User that can be attached to the Advantage User.

Configure Page

Configure Page (DESIGNER) allows you to apply extensions for pages in Advantage. Each of the major areas of the Advantage application (Administration, Human Resource Management, Financial, Performance Budgeting, and Vendor Self Service) has a separate Configure Page. It is very important that if you are creating an extension for a page, that you have accessed the correct Configure Page.

The page-level menu includes four Customized Application Pages links that open an adjusted view of Configure Page (DESIGNER), showing only those pages extended in the particular area of Advantage: Administration, Financial, Human Resource Management, or Performance Budgeting. The listings represent updates done online as well as file-based extensions. The Pending Synchronization page-level link gives you pages that are extended online, but to be synchronized.

Once you have found the page in the inquiry results, select **Configure** from the row-level menu. This action transitions you to a page that displays one tab for each view on the page. Each tab is comprised of three or four grids. Each grid presents different levels of page metadata for the current view.

- **Properties** - shows high-level properties of the view itself, such as its title and relative order to other views (when applicable, such as for a transaction page or multi-tab reference data or inquiry page).
- **Groups** - appears only when the view is capable of displaying multiple data sets, usually originating from different application data sources. For example, on the Purchase Order transaction, the Header view only displays data from one data source so “Groups” is not shown. The Summary view of the Purchase Order, however, uses four data sources to show information from the Header, Vendor, Commodity, and Accounting data sources so “Groups” is shown so that properties such as the titles and relative order of these data sets can be modified.
- **Sections** - lists the field sets contained within the view, typically corresponding to small sets of scalar fields seen within a collapsible scalar section or as a tab embedded within a grid row. Properties such as the titles and relative order of the sections can be modified.
- **Fields** - shows the fields on the page, organized by Group (when applicable) and Section. If the same field (from the same backend data source) appears in multiple places on the page during normal page usage, the field will appear just once in this list. Several field properties can be modified.

Applying Extensions

If you make changes directly on the initial record, (that is, Business Role, Business Process, Transaction, and Department all show “FOR ALL”), then the extension applies globally. If you want to specify an extension conditionally, you need to use the **Extend Same...** action available on the row-level menu. This will create a new row for the same Group, Section, or Field (depending on which grid you are using) where you must specify one condition (Business Role, Business Process, Transaction, or Department). You can use the Extend Same… action additional times for the same record, as long as the combination of Business Role, Business Process, Transaction and Department is unique.

Reverting Extensions

You can revert an extension using the **Revert Extension** action via the row-level menu. Note that certain extensions may require manual update to the corresponding page extension file in order to revert properly. You can revert all extensions for a Configure Page record by selecting **Undo Pending Extensions** from the page-level menu.

Add to / Remove from Migration Queue

If the enabled property is *true* for the markPendingExtensionsForSynchronization feature flag in the feature.conf file, then a record must be added to the migration queue before it is selected by the Designer to XML Synchronization Process. The following page level actions assist with adding to / removing from the migration queue and are only available if the enabled property is *true*:

- **Add to Migration Queue** – Selecting this action marks a record as complete, which indicates it will be selected by the Designer to XML Synchronization Process. This action hides the row-level menu for all rows of all grids of all the tabs, and prevent users from making any changes to any rows by making them read-only.
- **Remove from Migration Queue** – Selecting this action removes a record from the migration queue that was previously added to the migration queue. This action makes the record editable again and allows you to make any additional changes before it is again added to the migration queue.

If the enabled property is *false* for the markPendingExtensionsForSynchronization feature flag in the feature.conf file, then only pending extensions are processed by the Designer to XML Synchronization Process.

File Based Extensions

You need to use file-base extensions for supported extensions that are outside of the functionality of the Configure Page. This involves editing the page extension xml file manually. Sometimes you also need to update the application xml file. If competing extensions arise between the Configure Page and file-based extensions, the extension from the Configure Page take precedence.

Verifying Extensions

The online.extension.db.mode in the server_bootstrap.properties file controls whether an extension is immediately available after you save it on the Configure Page. The setting does not apply to file-based extensions, which always requires PalDb regeneration before the extension is seen online.

- Sites can set online.extension.db.mode to *true* when prototyping to immediately view changes. It is not recommended to use this setting during production, because the system needs to check the table storage for Configure Page extensions and merge them with the page layout coming from PalDb before presenting the page to the user. Also, if there is concurrent file-based extension development, you cannot see both Configure Page and file-based extensions in the online application unless you run the Designer Sync job followed by PalDb regeneration.
    - Sites should set the online.extension.db.mode to *false* for production environments. When this setting is used, the system sees what is in PalDb via RTFiles XML. Changes made via Configure Page are not applied to the application pages until the Designer Sync job is run to merge the extension to RTFiles, PalDb is regenerated, and the system is bounced.

Once an extension is available online, you can navigate to the page and verify that the extension appears as desired. Depending on the scope of the extension (global, role, process, transaction, or department), you may need to select the appropriate role, navigate to the page a certain way, or choose records that satisfy the transaction or department condition in order to see the extension.

Power BI Analytics

A limited number of Power BI analytics are available in the Artifact Library. Refer to the “Delivered Power BI Analytics” topic in the CGI Advantage Business Role User Guide for more information. The Power BI Analytics can be downloaded and tweaked, as necessary. Sites can then convert the analytic to a widget to add to the widget library, so the analytic can be used on a home page, accessed from an application page or added to a carousel tile. Your site will need a Power BI license in order to use Power BI Analytics on the home page, application page or carousel tile.

How to add a Power BI Analytic to an Application Page

After an Embedded Analytic has been added to the widget library, links to the analytic can be added to the page-level menu for the following types of pages:

- Transaction pages
- Activity folders
- Reference pages
- Inquiry pages

To allow a user to access a Power BI Analytic from the page-level menu on an application page the following changes need to be added to the page’s metadata so that the analytic name can be displayed under Related Pages links.

For example:

```
<nav name="analyticsForOrgCashFlow" title="FIN - Organizational Cash Flow" widgetId="3"
targetComponentType="StandardFormModalPage" targetLocation="viewManagerModal"
targetQualifiedName="admbase.page.Advantage.AnalyticsApplicationPage"
applicationUrl="{{applicationUrl}}" order="10"/>
```

In the above nav element, widgetId is the widget created on the WIDLIB page for displaying analytics. TargetComponentType is StandardFormModalPage, which specifies that a modal will be shown once we open analytics from the page-level menu. TargetQualifiedName is the page used to render analytics.

Powerbi_config.properties should be deployed to VLS\bin and all required properties should be correct.

How to add a Power BI Analytic to a Carousel Tile

After an Embedded Analytic has been added to the widget library, the analytic can be added as a carousel tile on a carousel page. To allow a user to view an analytic on a carousel tile for any carousel page, the following changes need to be added to the page metadata:

For example:

```
<nav name="analyticsForOrgCashFlow" title="FIN - Organizational Cash Flow" widgetId="3"
targetLocation="noDisplay" targetComponentType="PowerBIReportPage"
targetPage="adm.apgs.REPORT" applicationUrl="{{applicationUrl}}" order="10"
targetMode="browse"/>
```

In the above nav element, widgetId is the widget created on the WIDLIB page for displaying analytics. TargetComponentType is PowerBIReportPage, this page renders analytic on the UI side. TargetPage is the page used to render analytics, in this case adm.apgs.REPORT points to ApplicationAnalyticsPage.

Powerbi_config.properties should be deployed to VLS\bin and all required properties should be correct.

Alerts

Alerts are used to send messages in CGI Advantage. Alerts can be used to notify all users, groups of users, or specific users. Refer to the following topics for additional information about the Alerts functionality:

- Alert Management
- Purging Expired Alert Notifications
- Alert Email Notification
- Testing Email Functionality

Alert Management

Alerts are managed through a single Alerts (ALRT) page. The page opens with three tabs of information (*Received*, *Sent*, and *Draft*) and a Create button to create a new alert. Users without security access to create and send alerts will only see the Received tab and will not see a Create button. A user is granted access to create and send alerts if the **Administer Alerts** check box is selected on Access Control for the Alerts Resource Group and Security Role assigned to the user.

Refer to the following topics for more information:

- Create Alerts
- Received Alerts
- Sent Alerts
- Draft Alerts

Create Alerts

The Create Alerts page is accessed by selecting the **Create** button on the Alerts page.

› Field Information

Field	Required?	Description
Recipient(s)	Conditionally Required	Indicates the recipient of the alert. Valid options are: *User ID*, and/or a *Business Role*, and/or a *User Group*. The choice will display the correct field to recording recipient information. • The **User ID** to whom the alert message should be sent. An alert can be sent to multiple users with values separated by commas. • **Business Role** – The Business Role or business roles to which an alert message will be

		sent to all the User IDs that have the Business Role associated. • **User Group** – The User Group(s) to which an alert message will be sent to all User IDs associated with that User Group. A choice must be made unless the **All Users** indication is selected.
All Users	Conditionally Required	The selection of this indication will result in the alert being sent to all users in the system.
Type	Required	This field identifies the type of alert with respect to what the reader should do with the information: • *Info* – The alert provides information to the user, but does not require any actions on the user's part (for example, "The system will come down for maintenance at 6 pm"). • *To-Do* – The alert indicates an action that the user needs to perform (for example, "Change your password").
Priority	Required	The level of priority associated with the alert that is intended to convey an urgency the reader of the alert needs to act. Valid values are: *Critical*, *Medium*, or *Low*. Of these choices, *Critical* has the special function of becoming a pop up page for users already logged in or immediately upon logging in for those not actively in CGI Advantage.
Message	Required	This field captures the message text that is the alert. Use the items in the tool bar just above the field as needed to format the alert or include items in the alerts such as an external link, movie or picture.
Start Date and Time	Required	Each alert must begin at a certain date and time. The system will default the Server Date and Time if not entered. The format for this field is HH: MM. The alert message is available from the first minute of the Start Date until the End Time on the End Date.

End Date and Time	Required	Each alert must end at a certain date and time. Allow enough time here to account for users receiving the message.
Send Email	Optional	The selection of this indication triggers an email to the recipient/s with the alert in addition to the online alert message. However, to receive the email, the user must have setup their alert email address in their Accounting Settings page of the User Profile.

After the alert is successfully saved, the record appears on the Draft tab of the Alerts page for the sender if the Start Date and Time is in the future. If the Start Date and Time was the current date and time, the record appears on the Sent tab for the sender. Also, once sent, the alert can be viewed on the Received tab for the receiver.

Note: The Active Alert Interval (ACTIVE_ALERT_INTERVAL) Application Parameter (APPCTRL) defines the interval in minutes that is used to determine how often to trigger the Active Alert check. The Active Alert Interval parameter represents the number of minutes that should elapse before an Active Alert check should look for new messages. The value for this entry must be a positive integer.

Received Alerts

All “active” alerts are listed on the Received tab of the Alerts page for Administrative (authorized) users. Please see the "Accessing Alerts" section in the *Getting Started Guide* for information on the tab.

Sent Alerts

The Sent tab on the Alerts page lists all alerts that have been sent. This tab is only visible to authorized users (that is, those associated with an administrative security role).

- Field Information

Field	Description
1st Column (Alert Priority)	Orange Circle with Down Arrow – Low priority Red Triangle with Left/Right Arrows – Medium priority Red Diamond with Up Arrow – High priority
2nd Column (Read Status)	Red Circle with Dash - Unread Green Circle with Checkmark - Read
Message	The message text of the alert, which is also a transition to open a window to review all the text.

Type	This field identifies the type of alert with respect to what you should do with the information: • *Info* – The alert provides information to the user, but does not immediate on your part (for example, "The system will come down for maintenance at 6 pm"). • *To-Do* – The alert indicates an action that you need to perform now or in the near future (for example, "Change your password").
Start Date and Time	The Date and Time the alert was started (1st attempt at sending).
End Date and Time	The Date and Time the alert expires and will be removed.

› Record Level and Grid Actions

- **Copy** – Opens the Create Alert page with the transferrable parts of the selected past alert in order to send a similar or same alert again.
- **Delete** – Removes the alert from the Sent Alerts page for the sender of the alert.

Draft Alerts

Alerts are stored on the Draft tab of the Alerts page until the Start Date and Time arrive. For example, if an Alert is created on 11/1 with a Start Date and Time of 11/2 13:45, then the Alert message will be held on the Draft tab until 11/2 at 13:45. The layout of this tab is the same as the Sent tab but without the read/unread indication as it has not yet been sent.

› Record Level and Grid Actions

- **Edit** – Opens the Edit Alert page, which resembles the Create Alert page, for changing information on the draft alert.
- **Copy** – Opens the Create Alert page with the transferrable parts of the selected past alert in order to send a similar or same alert again.
- **Delete** – Removes the draft alert from system.

Purging Expired Alert Notifications

A batch job has been provided to Admin users for purging/deleting alerts by specifying a date for removal of expired alerts.

- The process removes any alerts (from all users) that have an End Date and Time earlier than the specified date and time. These alerts are considered expired.

- The process sends an alert message to all affected users indicating the date prior to which expired alert messages have been removed. Note: This type of alert message can also be removed the next time the batch job is run or can be individually deleted.

Alert Email Notification

This feature provides the ability for users to receive emails containing Alert messages in addition to the online message. When an Alert is sent, an email is sent to the recipient user's email address. Note the email is only sent when the Alert Email Notification selected and an email specified in their Account Settings page of their User Profile.

In order to send any alerts as an email, the Enable Email Alert Notifications (ENABLE_EMAIL_ALRT_NOT) on the Application Parameter (APPCTRL) page must be set to true.

Testing Email Functionality

The CGI Advantage application allows you to test email functionality without actually sending emails to recipients in a production or non-production environment. This feature allows you to test functionality as designed without the need to disable POP servers or scrub (replace email addresses with a dummy email address) email addresses from non-production or production environments. To enable the "dummy email" functionality for a specific CGI Advantage application, two parameters must be defined on Application Parameter (APPCTRL).

- SEND_DUMMY_EMAIL (Send Dummy Email) – When set to *true*, the CGI Advantage application sends all system-generated emails to the dummy email address defined in the SEND_DUMMY_EMAIL_ID parameter instead of the recipient email address. If set to *false*, the emails are sent to the recipient email address.

- SEND_DUMMY_EMAIL_ID (Dummy Email Address) – Used to record an email address for testing various email notification features.

Advantage Transaction Setup

Refer to the following topics for required setup by system administrators for transactions in Advantage. Refer to the *Transactions User Guide* for information on transactions.

- Calendar Date
- Future Transaction Trigger
- Transaction Category
- Transaction Type
- Transaction Code (Transaction Control)
- Transaction Creation Action
- Transaction Upload Templates
- Transaction Download Template
- Transaction Attachments
- Maximum Line Limit for Transactions
- Working with Transaction Signatures
- Transaction Archiving

Calendar Date

Every date for a fiscal year is defined on the Calendar Date page. This data is not the calendar functionality found throughout the application, but exists as a defaulting mechanism to give the current accounting period and fiscal year for a given date.

The fields on Calendar Date (CLDT) are as follows:

Field Name	Field Description
Calendar Date	Each date is defined MM/DD/CCYY.
Fiscal Year	The default fiscal year value to be inferred for the calendar date.
Fiscal Period	The default accounting period value to be inferred for the calendar date.
Holiday	An indication a date is considered a holiday.
Federal/Bank Holiday	An indication a date is considered a federal or bank holiday.

Weekend	An indication date falls on a weekend.
Billing Month Last Day	An indication date is the last day of a month for billing purposes. The last day of a month that is not a holiday or a weekend usually has this distinction.

Future Transaction Trigger

The Future Transaction Triggering chain uses data setup from transaction processing and entries created by users on the Future Transaction Triggering (FDT) page as input for scheduling and creates transactions on a scheduled basis. The chain will not only create and load transactions; it also creates later FDT entries when necessary as well as cleaning off transaction triggers past the Expire Date. The transactions created by the chain are not submitted by the chain. Some require manual intervention while others will be picked up by a System Maintenance Utility job defined to submit transactions.

On Transaction Control (DCTRL) there is an indication to control whether or not a transaction code can be established as triggered transaction - (Recurring Transaction must be checked). Any transaction type that cannot be copied cannot work successfully as a recurring trigger.

There are three types of triggers. The most common is the recurring one where the system copies either the most current version or the version specified on the trigger. This is the only trigger allowed in Advantage Administration. Refer to the *Financial Administration User Guide* for information on the other two triggers, which are allowed in Advantage Financial.

The following are the important pages related to this Chain job:

- Future Transaction Triggering (FDT)

  The fields on the Future Transaction Triggering (FDT) page are as follows with common ones omitted. Many of these fields cannot be updated after the Triggered Transaction.

Field Name	Field Description
Start	The required date for which the page will apply frequency settings to determine when the next trigger should occur.
End	The date that triggering should stop.
Recurring JV Reversal Reclassification	The type of trigger generated: • *Recurring* – Will create one or more copies. • *JV Reversal* – Will create a copy of the source journal voucher where the debit and credit amounts on the Accounting tab are switched and memo references are created back to the source transaction.

	• *Reclassification* – Certain transactions allow this action to create a subsequent version and make pre-defined updates (CR, RE, PRLLL, PRLCE, PRACR)
Frequency Type	Choices of: • *One Time Future* – The default for JV Reversal and Reclassification. • Weekly • Monthly by Date • Monthly by Day
Frequency Type 2	A second level of frequency required for Weekly, Monthly by Date, and Monthly by Day. Values in this field change for each of those three.
Frequency Type 3	A third level of frequency required for Weekly, Monthly by Date, and Monthly by Day. Values in this field change for each of those three.
Active	When *true*, the record is eligible for selection by the Future Transaction Triggering Process. When false, the trigger is paused until activated again.
Carry Forward Amount	When *true,* the record will create the triggered transaction with the Line Amount found on the accounting line of the source transaction. When *false*, the Line Amount field will be $0.00 with the intention of a user completing that amount once it is known. This option only applies to Recurring triggers.
Hold Transaction	When *true*, the transaction will be created with a Transaction Status of Hold with the intention of the transaction being skipped by an automated submit process.
Bypass Approvals	When allowed by setup on Future Transaction Triggering Options page, the indication of *true* will result in the Bypass Approvals action being applied to the triggered transaction. The idea is that the original transaction went through approvals so copies of it do not need to be approved.

Transaction Prefix	An optional prefix that will be used in Automatic Transaction Numbering for the creation of the target transaction. If specified in Future Transaction Triggering Options setup, that prefix will default and cannot be changed.
Expire Date	A date at which time a successful trigger record will be deleted by the Future Transaction Triggering Process. The default is six months after the Start Date, but that can be overridden.

Future Transaction Triggering Options (FDTO)

The Future Transaction Triggering Options (FDTO) page allows you to specify different options for different Transaction Codes or specific Transaction Code/Department combinations. Setup on this page is optional. FDTO edits are only performed when records are being inserted or updated on the Future Transaction Triggering (FDT) page. The FDT page allows you to set up, maintain and trigger future transactions.

The fields on the Future Transaction Triggering Options (FDTO) page are as follows with common ones omitted.

Field Name	Field Description
Transaction Code	The first part of two in the definition of a Future Transaction Triggering Option.
Department	The second part of two in the definition of a Future Transaction Triggering Option.
Bypass Approval Option	A required indication to control the Bypass Approvals field found on Future Transaction Triggering. Valid values for this field are: • *Do Not Allow* – Users are not allowed to bypass approvals; therefore, the Bypass Approval indication will always be *false.* • *Allow* – Users can make the choice for individual triggers to bypass approvals or not; therefore, the Bypass Approval indication will default to false but can be changed to *true.* • *Require* – The Bypass Approvals indication will default to *true* and cannot be changed. Note: The Allow and Require values cannot be selected unless the Approval Override Allowed indication on

	Transaction Control (DCTRL) is selected for the specified Transaction Code.
End Date Limit	A required number of days to limit the time between the Start Date and the End Date. A value of 0 means there is no limit.
Copy Final Phase Only	When *true,* only a transaction version can be specified as the source in a trigger is the Transaction Phase is *Final* or *Historical Final.*
Transaction Prefix	An optional prefix, already setup with the Transaction Code, Department and current Fiscal Year on Automatic Transaction Numbering that must be used for all triggered transactions. That verification will use the wildcard of **** when the Department is ALL and after any specific Department did not match a record.

For detailed information on the job (such as when to run, input, output, and process parameters) refer to the associated run sheet in the *CGI Advantage General Accounting Run Sheets* guide.

Transaction Category

The first and highest level in the transaction hierarchy is Transaction Category. Values established at this level are not used by the application for any processing rules, but serve as a reporting tool. The level is also available for custom edits or us on a Chart of Accounts Combination Validation, Required Element, or Inference page.

The fields on the Transaction Category (DCAT) page are as follows with common ones omitted:

Field Name	Field Description
Transaction Category	The required transaction category for which a transaction types will be associated.

The following category is delivered with the Administration part of CGI Advantage. Refer to the *Financial Administration User Guide* for the list of Transaction Types delivered with Advantage Financial.

Category	Name
SC	Security

Transaction Type

The second level in the transaction hierarchy is Transaction Type. It is at this level that the application associates database tables for data storage. Most application edits found on a transaction are set at the transaction type level with a few exceptions for transaction sub type.

Changing the delivered transaction type codes should not be done, as the application will not function properly. Changing other information on this page is allowed. New records inserted in this page will not function without considerable development.

The fields on the Transaction Type (DTYP) page are as follows with common ones omitted:

Field Name	Field Description
Transaction Type	A unique identification of a transaction type to which transaction codes will be associated.
Transaction Category	The required transaction category for which a transaction type is associated.

The following transaction type is delivered with the Administration part of CGI Advantage. Refer to the *Financial Administration User Guide* for the list of Transaction Types delivered with Advantage Financial.

Type	Name	Category
USER	User Maintenance	SC

Transaction Code (Transaction Control)

The lowest level in the hierarchy is the Transaction Code. All transaction codes are defined on the Transaction Control (DCTRL) reference page. All transaction codes must be unique within the application and not just within a transaction type.

Additions and changes to Transaction Control are expected in any application. Care should be taken when establishing custom transaction codes because the same value may appear as a delivered code in a future upgrade. Unlike many other warnings about setting up custom data, transaction codes are slightly different. Many of the names of custom transaction codes are those used in prior Advantage or other applications. To change the codes that users have become familiar with, is a decision often decided against. Therefore, adequate documentation should be maintained on any custom transaction code setup so that it will not be lost when an upgrade presents a new baseline code that is identical. One way to make custom codes unique, when the opportunity exists to establish a new code, is to use a delivered code followed by a number. Outside of the budgeting area, which uses numbers to tie to a budget structure, accounting transaction codes delivered will not contain numbers. Delivered transaction codes attempt to serve as acronyms, and will not contain numbers.

The primary fields on the Transaction Control (DCTRL) page are as follows with several common ones omitted.

Field Name	Field Description
Transaction Category	The inferred transaction category value from the transaction type value entered.
Transaction Type	Every transaction code must be assigned to a transaction type.
Transaction Code	Each transaction code is identified by a unique identifier.
Sub Type	The optional transaction sub type to which a transaction code will be defined.
Home Application	One of a variety of choices will designate that a resource (transaction code in this case) belongs to a certain application. This required value will be used when filtering pages such as the Transaction Catalog and Worklists based on the application to which a user has logged into.

As there are many options for controlling different types of transaction functionality disbursed across this setup page, the help for each control has been grouped into the following sections where similar controls are grouped even further. This order is different from that presented online to facilitate the initial setup of data on this page, which is a critical function to all implementations and subsequent upgrades where new Transaction Codes will be used.

- Creation

  The options presented in this section will apply to all transaction types (or nearly all because of certain transaction type exceptions) and are invoked even before an instance of a transaction is presented for data entry or update.

Field Name	Field Description
Online Creation	When *true,* new, modification, or cancellation drafts of a transaction code can be created online. A user's security settings would be evaluated to see if the type of draft can be created.
Offline Required	When *true,* the transaction code should only be created offline. Transaction codes that are for interfaces and created only by batch programs often have this control set to *true.*

Create	When *true,* new draft version is allowed. Only the Transaction Function of *New* is controlled by this option. A user's security settings will still be evaluated for creating the transaction code.
Transaction Unit Required	When *true,* the Transaction Unit field must be used in transaction creation for identification or security purposes.
Auto Numbering	Three settings determine if auto numbering has to be used: • *Prohibited* – Will not allow automatic numbering so a Transaction ID must be supplied. • *Required* – Automatic numbering is required. • *Optional* – A Transaction ID can be supplied or generated. There are no cross edits between this field and the Auto Numbering (ADNT) page.
Minimum Transaction ID Length Maximum Transaction ID Length	Optional controls that will establish a minimum or maximum Transaction ID length to be more restrictive than the limits on the Application Parameter (APPCTRL) records for Minimum Transaction ID Length and Maximum Transaction ID Length. The Auto Numbering (ADNT) page also edits the Format Field Length field against these controls.
Modify	When *true,* modification draft versions are allowed. Only the Transaction Function of *Modification* is controlled by this option. A user's security settings will still be evaluated for modifying the transaction code. Even is marked *true,* those transaction types that do not allow modifications by design will not allow the action.
Cancel	When *true*, cancellation draft version is allowed. Only the Transaction Function of *Cancellation* is controlled by this option. A user's security settings will still be evaluated for cancelling the transaction code. Even is marked *true*, those transaction types that do not allow cancellations by design will not allow the action.
Alternate Page Code	When a transaction has an alternative view, such as a wizard, the page code for that alternate view should be specified in this field. Completion will enable the View link via the Related Action row-level menu in a worklist. With this action an approver can open an alternate view of a transaction instead of selecting the transaction link to open the standard view.

› Processing Actions

The options presented in this action will apply to all transaction types (or nearly all because of certain transaction type exceptions) and are after an instance of a transaction is presented for data entry or update and do not apply to just a single transaction tab.

Field Name	Field Description
Collaboration	When *true,* the Collaborate action for a transaction code. This is the first of several system configurations to use the collaboration function.
Submit	When *true*, the submit action is allowed by users online. A user's security settings will still be evaluated for submitting the transaction code. When *false*, the system must perform the submit action. If a transaction code should be allowed to be submitted online, this control should be set to *True*. If a batch job should only submit the transaction code, this control should be set to *No.* The default for this indication is *True*. A user's security settings would be evaluated to see if submitting the transaction code is allowed at the individual level as this DCTRL option is at the system level.
Submit Phase	A setting of *Pending* means there will be the evaluation of workflow rules with the submit action. A setting of *Final* means workflow rules will not be evaluated.
Workflow Process	A choice of four values control what type of workflow evaluation will be done: • *None* - No workflow evaluation even if the Submit Phase is *Pending.* • *Internal* – Advantage workflow will be evaluated if the Submit Phase is *Pending.* • *External* - 3rd party workflow application integrated into CGI Advantage Financial will be evaluated if the Submit Phase is *Pending.* • *Both Internal and External* – Both Advantage and 3rd party workflow will be evaluated if the Submit Phase is *Pending.*
Workflow Asynchronous Processing	When *false*, after the final approval is applied and there are no new errors, then the transaction will go to final. When *true,* after the final approval is applied and there are no new errors, the transaction does not go directly to final but stays in

	pending. It will take another Submit action for the transaction to leave pending and go to final.
Single Approvals Enforced	When *true*, a user is allowed to apply only one approval to a transaction and the Single Approvals Enforced indication on the Approval Rules (IWF08) page is. When *false*, a user can apply multiple approvals unless the Single Approvals Enforced indication on the Approval Rules (IWF08) page is *true.*
Approval Bypass	When *true*, if the transaction is submitted into workflow, a user with sufficient security settings can choose to bypass all approvals and submit the transaction.
Override Pending Phase	When there are override errors with a transaction code with workflow setup, there are three values that control the override and the approval: • *Allowed only after reaching* - Overrides to be applied in pending and not draft. • *Required before reaching* – Overrides have to be applied in draft. • *Allowed before or after reaching* – Overrides can be applied to either pending or draft
Recurring Transaction	When *true*, records can be added to the Future Transaction Triggering (FDT) page to create copies of the transaction.
Component-Specific Application Resource	When *true,* a tab-specific application resource should be used for tab security so that the system performs security authorization at the transaction tab level as the application resource rather than transaction code.
Log Discard	When *true,* all discards of draft transactions will be tracked on the Transaction Discard Log (DSCRDLOG) page.
Apply Effective Dates Unchanged Edit	When checked, allows users to ensure the effective dates are updated for all transactions that affect the timeline maintained tables. When checked, if transaction affecting the timeline maintained tables are modified without changes to effective dates then an overrideable error message is generated.

Refer to the "Transaction Code (Transaction Control)" topic in the *Financial Administration User Guid*e for information on the options provided on this page in the Advantage Financial application.

Transaction Creation Action

This page lists the available actions for the automatic creation of transactions from various places in the application, including reference data pages (Vendor Customer) and query pages (Invoicing Search). Each action contains the following:

- **Action** – Signifies which fields need to be copied forward into the target transaction from the source table or transaction. Valid values are coded into the application so that new values cannot be defined without development effort.
- **Transaction Type** – The transaction type to perform the action.
- **Transaction Code** – The transaction code to perform the action.
- **Action Name** –A short, textual description of the action which signifies what is to take place on the target transaction.
- **Transaction Sub Action** – The secured transaction sub action to be processed on the document by which it will be populated.
- **Transaction Sub Action Name** – The name for the sub action.
- **Transaction Name** – The name associated with the selected Transaction Code.
- **Default** – This flag indicates whether the current record is to be considered the default record for its action code when the user is presented with a choice of transactions on the Create Transaction page.

Note: The Populate Document Codes process should be run before adding a record to the table for a new cloned transaction.

Transaction Upload Templates

Advantage allows you to upload transactions to Advantage by uploading spreadsheets that have been populated with required information. The spreadsheets populated must come from system-generated templates or from customized templates that have been validated and uploaded to the Transaction Upload Template (DOCUPTEM) page. For information on uploading transactions to Advantage using a template that has been added to the Transaction Upload Template page, refer to the “Transaction Upload Spreadsheets” section in the *CGI Advantage Transactions User Guide*.

Refer to the following topics in this user guide for additional information:

- Create a System-Generated Template
- Customize a System-Generated Template
- Validate a Customized Template
- Upload a Customized Template

Create a System-Generated Template

System-generated transaction upload templates provide a template for users to upload numerous transactions for a specified transaction code into Advantage. The sections and field names are derived directly from the UI of the specified Transaction Code.

The Generate/Upload Template page allows you to create system generated transaction upload templates. This page is accessed by selecting the **Generate/Upload Template** link on the Transaction Upload Template (DOCUPTEM) page. Authorized users can create a system-generated transaction upload template by entering the Transaction Code in the Code field and then by selecting the **Generate Template** action. (Note: The Generate Template check box must be selected under the Application Actions section on the Access Control (SCRACS) page in the Administration application for the user's Security Role.) The system-generated transaction upload template is added to the Transaction Upload Template (DOCUPTEM) page.

You can transition to the Transaction Upload Template page by selecting the **Transaction Upload Template** link. You can then enter search criteria to locate the template that was generated. At this point the system-generated template can be populated with transaction information and then uploaded to the Upload Transaction Spreadsheet (UPDOCS) page. Refer to the "Transaction Upload Spreadsheets" topic in the *CGI Advantage Transactions User Guide* for more information. You can also customize a system-generated template, validate and upload the template to the Transaction Upload Template page. Refer to the "Customize a System-Generated Template" topic for more information.

Transaction Layout Generator Utility

The System-generated template is a replica of the transaction in terms of the field labels and their layout. Transaction Layout Generator Utility is a standalone utility which generates and maintains metadata about the transaction. The CGI Advantage System Administrator should execute this utility whenever there is a change in the transaction layout. Refer to the "" section in the Appendix for instructions on how to run this utility. This utility should be executed in the following situations:

- Site is new to Advantage and is setting up Advantage for the first time.
- Site is upgrading from a previous release.
- Site has custom transactions.
- Sites applying defect fixes specific to transaction layouts

For sites that have cloned transactions and do not have a build machine to run the Transaction Layout Generator Utility, the following steps should be followed to use the spreadsheet upload feature:

- Identify the Transaction Code of a cloned transaction. For example: MB1 which is a clone of the CH transaction.
- Locate the folder that contains the fCH.json file under the RTFiles/VLS1/DocSignatures location.
- For example, the location for the CH transaction template is here:
- C:\CGIADV\RTFiles\VLS1\DocSignatures\Template\Cost_Accounting_Reference
- Create a copy of the fCH.json file in the same folder and name the file fMB1.json.
- Perform the Generate Template action for the MB1 transaction from the Generate Template (DOCUPTEM) page.

- Result: The System generated template is available for download.

Customize a System-Generated Template

System-generated templates can be customized, validated and uploaded prior to adding documentation information. You must first follow the steps in the "Create a System-Generated Template" topic to create a system-generated template. Navigate to the DOCUPTEM page and enter search criteria to find the system-generated template for your Transaction Code. Select the hyperlink in the Template File Name column to open the template in Microsoft Excel. Save the excel file on your computer with a custom file name and an .xlsx extension.

You can make customizations to system generated transaction upload templates; however, some restrictions apply:

- The template must be in Microsoft Excel format with an .xlsx extension.
- The fields marked with an asterisk (*) sign in the spreadsheet are required/mandatory fields in Advantage; therefore, these fields cannot be removed.
- Fields that are not marked with an asterisk (*) sign in the spreadsheet can be removed.
- Field captions can be renamed; for example, Transaction Department Code can be renamed to Transaction Dept Code.
- Attribute Names cannot be changed; for example, DOC_DEPT_CD cannot be changed to DOCUMENT_DEPT_CD.
- The fields within a section can be re-ordered.
- A section cannot be deleted; for example, you cannot delete the Vendor section from the spreadsheet.

Once you have made your customizations, you must validate the excel file on the Generate/Upload Template page. Refer to the "Validate a Customized Template" topic for more information.

Validate a Customized Template

If a system-generated template has been customized, as indicated in the prior section, then the excel file must be validated on the Generate/Upload Template page, prior to uploading the template to the Transaction Upload Template (DOCUPTEM) page. The Generate/Upload Template page is accessed by selecting the Generate/Upload Template link on the Transaction Upload Template page.

The Validate section of the Generate/Upload Template page allows you to validate a customized transaction upload template to ensure that it abides by the restrictions set by the system. You must populate the Transaction Code field that corresponds to your template, select the template file via the Browse button, and then select the **Validate Template** action to validate the template. If there are issues with the template, then the following message is displayed: "The template has errors and needs to be rectified". You must fix the errors and then perform the validation again. These steps must be repeated until the template validates successfully. If the template validates successfully, then the following message is displayed: "The template validated successfully", which indicates you can upload the customized template. Refer to the instructions in the "Upload a Customized Template" topic for more information.

Upload a Customized Template

Once a customized template has been validated successfully, authorized users can upload the template via the Generate/Upload section on the Generate/Upload Template page. The Generate/Upload Template page is accessed by selecting the Generate/Upload Template link on the Transaction Upload Template (DOCUPTEM) page.

To upload a customized template you must enter the Transaction Code in the Code field and you must select the customized template file by selecting the Browse button. You can then optionally populate the Dept, Unit, and Template Description fields before selecting the Upload Template action. (Note: The Upload Template check box must be selected under the Application Actions section on the Access Control (SCRACS) page in the Administration application for the user's Security Role.) If the upload is not successful, then the following message is displayed: "The template has errors and needs to be rectified". If the upload is successful, then a record is added to the Transaction Upload Template page and the following message is displayed: "Template uploaded successfully".

Once the template has been uploaded to the DOCUPTEM page, authorized users can download the excel file, add transaction information, and then upload the excel file to the Upload Transaction Spreadsheet (UPDOCS) page. Refer to the "Transaction Upload Spreadsheets" topic in the *CGI Advantage Transactions User Guide* for more information.

Transaction Download Template

Advantage allows you to download data from any transaction in Advantage into a user-selected template (for example, an Excel spreadsheet), which can then be saved as an external file and is available for use outside of Advantage. Clicking **Download Transaction** from the Action Menu within a transaction transitions you to the Transaction Templates (DTPL) page. This page allows you to search for and select the template that you want to use. For more information about this page, please refer to the "Transaction Templates topic in the *Transactions User Guide*.

Any transaction is eligible for download, regardless of the transaction's Phase, Function or Status. The download functionality is limited; however, by the templates that are available for your site. The templates are not part of the CGI Advantage baseline software package. Sites can create their own templates and then reference the Template File Names on the Transaction Download Template (DDTPL) page.

This chapter of the user guide is divided into the following sections:

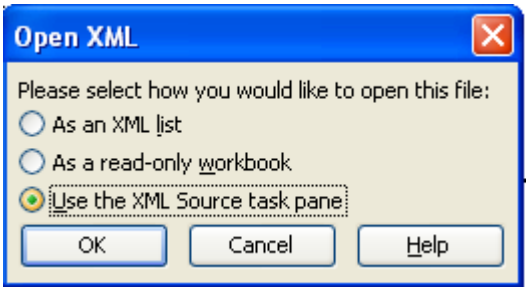
- How to Develop Transaction Download Templates
- How to Setup Transaction Download Templates in Advantage

How to Develop Transaction Download Templates

This section provides instructions on creating the .xlsx template, which can then be referenced by Advantage so that users can download data from transactions in Advantage into the template. Most formatting information will be contained within the Microsoft Excel template. To create a template, the System Administrator should follow these steps.

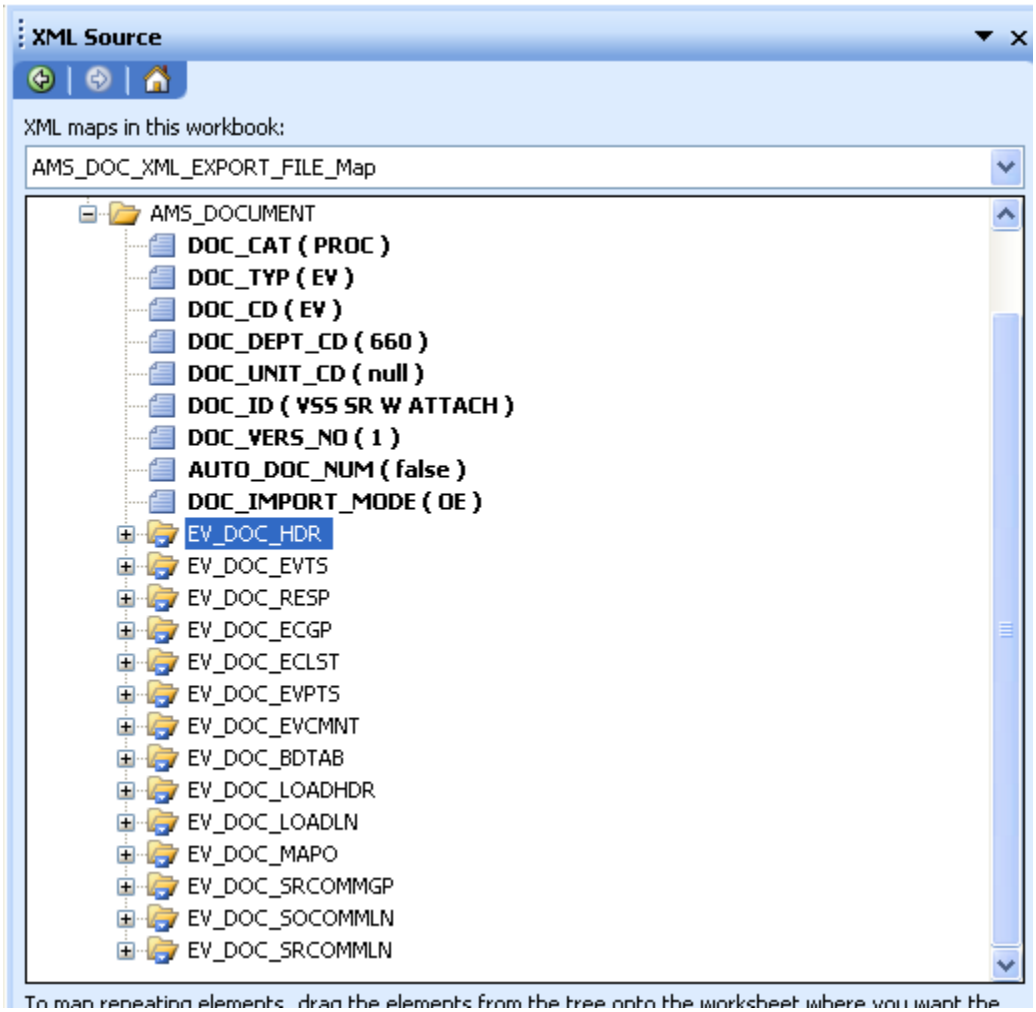
1. Use the System Maintenance utility to export a sample transaction for EACH transaction code that you want to be able to export as .XLSX. The exported transaction should be finalized and have data in all components that will be needed in the template.

2. Open Microsoft Excel (2007/2010) and use: file>>>open to access the .xml file created. Excel will produce the following dialog:



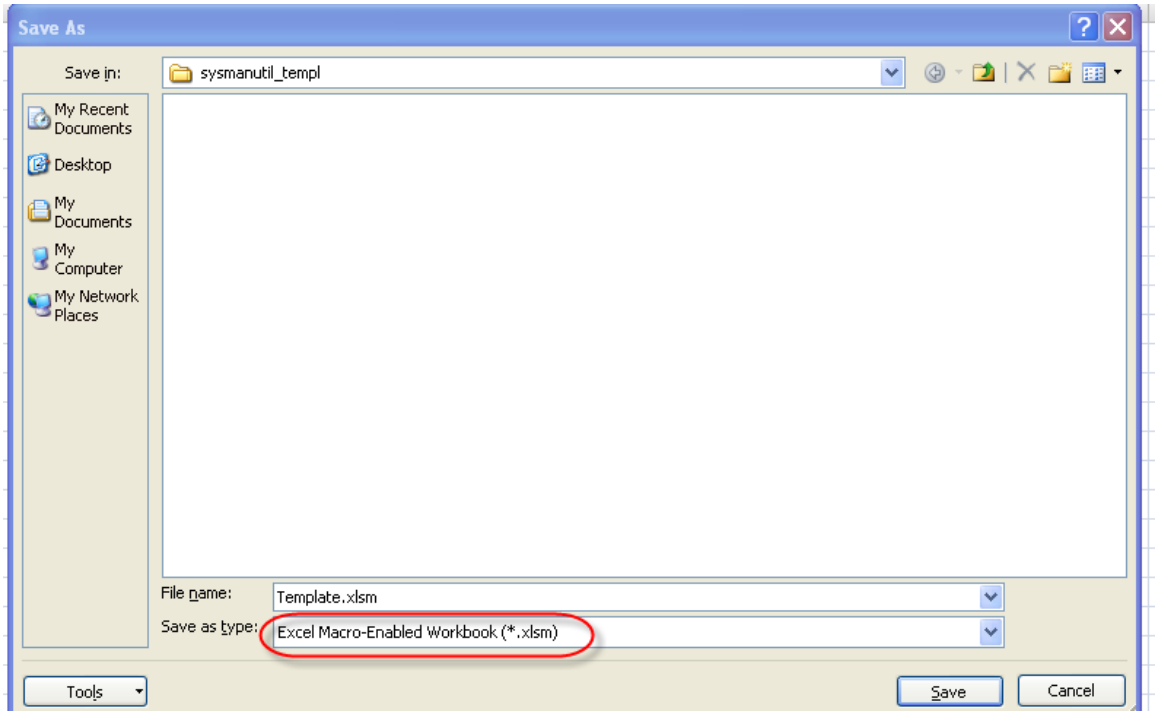
3. Choose the **Use the XML Source task pane** option and click the **OK** button. EXCEL will read the file and produce the "XML Source Task Panel".

4. Using the Source Task Panel you may select individual components/sections of the transactions to be displayed by dragging from the panel to any place in the Excel workbook.



5. Once the required transaction components have been laid out on the appropriate part of the worksheet, the Administrator will record Macros or write Visual Basic for Application code to format the data exported from Advantage. This logic should be recorded in the open event for the file. Advantage will trigger these macros/programs to run when the export occurs.

6. Save the file in the required version of .XLSX. Note: If the Excel file contains Macros then save it as an '.XLSM' file as follows:



7. Once the template has been created it must be placed in the exact directory specified in the adv30params.ini file. An administrator with access rights to the directory location on the server will load the template file to this location.

Special Notes

Templates must be established using the component name and database column name to define each field of data that will be downloaded. However, Advantage will display the field in the external file with its caption name from Advantage. For example: the database definition of 'EV_DOC_RESP.LGL_NM' will appear in the external file as 'Vendor Name'. In the event that the downloaded field does not have a caption (that is, the field is not displayed on the Advantage transaction), then a caption can be defined in the template. In fact, a template caption will always take precedence over an Advantage caption. Similarly, stored values from Coded Valued Lists (CVLs) will be interpreted into their display values. For example, if a 'Business Type' has a stored value in the database of "1", Advantage will display the field in the external file as "Minority Business". If there are any processing errors, an error message is displayed in the external file which results from the download.

Macro Example

This macro creates 3 worksheets: Line Item, Line Item Group, and Total Bid.

- The Line Item worksheet is a vendor by vendor listing of each commodity line item. The Line Item worksheet is grouped by Line Item Number and sorted by line item total bid in ascending order.
- The Line Item Group worksheet is a vendor by vendor listing of each commodity group. It is grouped by “Group No” and sorted within each “Group” by the Group Total Cost in ascending order.
- The Total Bid worksheet is a summary of each Vendor’s bids. Vendors with the lowest total bid amount appear in the leftmost column.

Function Overview

- Sub Main

  Main calls all other procedures. Something unique to Main is that it activates the Line Item worksheet and sets the cell formatting of the Discount Percentage column to ‘General’. This is setup for converting the Discount Percentage values from text to numbers.

- Sub Sorter

  Sorter creates a worksheet called temp which is used as scratch. Sorter counts the number of Line Items and sorts each group. It sorts each group by calling Sort. Sorter passes Sort the column to sort by, the row to begin the sort at, the number of rows, the number of columns, and the bid response column. Sorter has an optional argument ‘bygroup’ that allows for individual sorting as opposed to the group sort default. While sorting, if Sorter encounters a line item with a Line Type of 3, Sorter converts the value of the Discount Percentage field of that line to numbers. After the sorting is complete, Sorter parses the worksheet and replaces the 9.9E+307 values with “No Bid”. Sorter then deletes the temp worksheet.

- Sub Sort

  Sort is called by Sorter and basically sorts a specific range of rows. Sort is passed the column to sort by, the row to begin the sort at, the number of rows, the number of columns, and an optional bid response column. It uses the selection sort algorithm. To handle no-bid rows, Sort places 9.9E+307 in the total amount field. This ensures that the row appears at the bottom of the sorted list. As part of the selection sort algorithm, Sort calls Swap.

- Sub Swap

  Swap is passed two rows and the number of columns. Swap switches the two rows that it was passed.

- Sub SortLine

  SortLine is used to sort rows by item number or group number. To achieve this, SortLine passes hard coded row values to the Sort function. After the sorting is done, SortLine deletes the temp worksheet.

- Sub ReplaceNull

  ReplaceNull parses through the worksheet and replaces all null values with a zero.

- Sub TotalBid

TotalBid places the Total row on the Total Bid worksheet. TotalBid creates the temp worksheet, copies the Vendor names from the Line Item Group worksheet to the temp worksheet and then uses a hidden column containing group information to pull out corresponding line item prices from the Line Item worksheet. The same process is then repeated for line item totals which reside on the Line Item Group worksheet. The data is selected and then transposed to the Total Bid worksheet.

How to Setup Transaction Download Templates in Advantage

Once a template has been created for your site, an entry must be created on the Transaction Download Template (DDTPL) page specifying the Transaction Type, Transaction Code, Transaction Department along with the descriptive name of the template, the exact name of the Template File, and the intended Target Application in which the resulting external file should open. It should be noted that the Target Application can be precluded by a user's local desktop settings. For example, the Target Application will indicate the intended type of output file that will result from the download, such as Excel for a template file ending with an .xlsx /.xlsm extension. However, if the user has a local setting which associates any of the two .xlsx /.xlsm extensions to a different application then that setting will prevail.

Limitations exist for the size of data that can be downloaded. The Transaction Download functionality supports the download of a transaction up to a configurable limit of lines, which is the cumulative total of all transaction component lines. The upper limit is set on the Application Parameters (APPCTRL) page, and can be further limited per Template using the Max Download Lines field on DDTPL. Please note that row and column limits are also set by Microsoft Excel and other similar products.

The directory location (that is, path) of the templates is specified in the adv30params.ini file. An administrator with access rights to the directory location on the server will load the template file to this location. The directory location should not be accessible to regular system users.

Deleting an entry from Transaction Download Template (DDTPL) page will not delete the template file itself but will make that template unavailable for use for all users.

Transaction Attachments

CGI Advantage allows you to upload and attach files created in other applications (for example, Excel spreadsheets and Word transactions), to Advantage transactions. These files are known as attachments. These attachments are associated with transactions they were "attached to" and are then treated as an extension of the transaction.

If the transaction were to be archived, the attachments would be archived as well. Since attachments are treated like an extension of a transaction, CGI Advantage security applies to these attachments. Thus you will be able to access attachments for only transactions which you are authorized to work with.

You may add, delete, and restore attachments to a transaction, as long as a transaction is in *Draft* phase. The attachment information may be viewed at any time from the Attachments page. Once an attachment has been added to a transaction, CGI Advantage keeps track of the attachment information, in the Attachments Log. The "Using Attachments" section in the *CGI Advantage Transactions User Guide*, further describes the Attachments functionality.

The Attachment Count feature can be turned off for all transactions within the application by setting the Bypass Transaction Attachment Counts (BYPASS_DOC_ATT_CNTS) parameter on the Application Parameters (APPCTRL) table to true. If the Bypass Transaction Attachment Counts (BYPASS_DOC_ATT_CNTS) parameter is set to false, then the Attachment Count feature can be turned off for individual transactions by specifying the transaction codes in a comma separated list in the Bypass

Transaction Attachment Counts List (BYPASS_DOC_ATT_CNTS_LST) parameter on the APPCTRL table.

The AttachmentTypesAllowed parameter in the ADV30Params.ini file indicates the Attachment Types that can be uploaded in Advantage. If the value is an asterisk (*) or is null, then all types of files can be uploaded. Otherwise, only the Attachment Types listed for this parameter are allowed. The system interprets the file types written in the AttachmentTypesAllowed parameter in a case-insensitive manner.

The Maxuploadfiles parameter in the adv30parms.ini file controls/restricts the number of files a user can add to the Upload Attachment tab at one time using multiple file selection. The default value of the parameter is set to <= 5. However, there is no restriction with regard to the number of attachments that can be added to the Upload Attachment page or Attachment page. For example, you can use the Browse action multiple times to add more files than the Maxuploadfiles parameter allows you to add at one time to the Upload Attachment page.

Transaction Attachments in Advantage ESS

The Employee/Manager Attachment Options section, on the Transaction Control (DCTRL) reference page in Advantage HRM, allows you to control transaction attachment capability on Advantage ESS transactions. You can enable/disable attachments for a specific ESS transaction, as well as to provide additional attachment criteria, such as limiting the size, type, and number of attachments that can be attached, and selecting the Configurable Text (CTEXT) that will be displayed on the transaction attachment pop up page.

Note: Only the ESS transactions that support attachment functionality can be configured to allow attachments. The Employee/Manager Attachment Options section is only available in Advantage HRM.

The following fields are used to configure the attachment criteria:

- Allow Attachment - This check box allows you to specify whether or not transaction attachments are allowed on an Advantage ESS transaction. If this check box is selected, additional attachment configuration fields in this section are enabled and attachment functionality will be allowed for the specified ESS Transaction.
- Attachment Type Allowed - This field allows you to specify the acceptable file types that can be attached to the selected Advantage ESS transaction. This field is required when the Allow Attachments check box is selected. You can enter multiple valid file type extensions using commas. Allowable/valid types are: TXT, HTM, DOC, PDF, XML, TIFF, XLS, ZIP, DOCX, XLSX, CIF, XLSM, GIF, PNG, JPG, JPEG, and BMP. Note: The entered file type must be globally supported by the ADV30Parms.ini file. If the specified file type does not correspond with global standards as found in the ADV30Parms.ini file, then an error message is issued.
- Number of Attachments- This field allows you to specify the number of transactions that can be uploaded to the selected Advantage ESS transaction. This field is required when the Allow Attachments check box is selected. The maximum number of attachments that can be defined in this field is 99.
- Attachment Size Limit - This field allows you to specify the size limit (in megabytes) for each transaction attachment for the selected Advantage ESS transaction. This field is required when the Allow Attachments check box is selected. The Attachment Size cannot exceed the globally defined file size defined in the ADV30Parms.ini file.

- Configurable Text - This field allows you to enter a Configurable Text (CTEXT) code associated with site configurable instructions regarding how to attach files for the selected ESS transaction. The CTEXT code must be set up on the CTEXT page.

If the Allow Attachment check box is selected, an Attachments button or paper clip icon will appear on the ESS transaction. When the ESS user selects the Transaction Attachment button/icon, an Attach Transaction pop up window is opened, displaying the configurable text instructions, and the user can select the Browse button to locate the file they wish to attach. Once an attachment is added to a transaction, the attachment file name is displayed along with a paper clip icon.

Maximum Line Limit for Transactions

Transactions can be entered online or generated through batch processes. A large transaction with a large number of lines can potentially yield out of memory errors when any action is performed on it. Therefore, in an effort to minimize such situations, a restriction can be placed on transactions in the number of lines it can contain in each transaction component for a single transaction.

Refer to the following topics in this user guide for additional information:

- Setting up DCREQ to enforce Maximum line limit on transactions
- Influencing Factors
- Quantify Maximum line limit property value settings
- Certain Exceptional Cases

You can also refer to the "Line Number Limitations" topic in the *Transactions User Guide* for more information.

Setting up DCREQ to Enforce Maximum Line Limit on Transactions

An entry with property name MAX_LINE_LIMIT can be entered on the Transaction Component Requirements (DCREQ) table for transaction components where the Maximum line limit needs to be enforced. It indicates the maximum number of lines that can be inserted into a single transaction for a given transaction component. Edits are placed in the Transaction Engine that create new transactions to ensure that the number of lines inserted in newly created transactions for various transaction components do not exceed the Maximum line limit property value set on the DCREQ table for those transaction components. If the limit is exceeded then a SEVERE error is thrown and the action is rolled back.

Such limits can also be explicitly placed on batch processes generating new transactions. Refer to the “MaxLineLimitForBatch – Helper class for generating transactions within Maximum line limit” topic in the *CGI Advantage Financial Developer Guide* for details.

These limits are only enforced when the “EnforceMaxLineLimit” property value is set to *true* on ADV30Params.ini. This is a system-wide property and the default provided value for this property is *true*.

Any insert/update/delete done on the Transaction Component Requirements (DCREQ) table will take into effect only after all application servers (VLSs dedicated to online/offline processing) are brought down and then brought up.

Examples of Maximum line limit entries on DCREQ:

Transaction Type	Transaction Component	Property Name	Property Value	What it means?
PR	PR_DOC_COMM	MAX_LINE_LIMIT	200	The maximum number of lines that can be contained on the Commodity component of a single Payment Request transaction is 200.
PR	PR_DOC_ACTG	MAX_LINE_LIMIT	400	The maximum number of lines that can be contained on the Accounting component of a single Payment Request transaction is 400.
XYZ	XYZ_DOC_HDR	MAX_LINE_LIMIT	0	The transaction Header creation for transactions with a Transaction Type of XYZ is not allowed. This means that transactions with Transaction Type XYZ cannot be created.
ABC	ABC_DOC_VEND	MAX_LINE_LIMIT	0	Lines cannot be added to the Vendor component of transactions with Transaction Type ABC.

Influencing Factors

The following factors should be considered while setting up the MAX_LINE_LIMIT property values for various transactions.

- Placing such limits might cost a slight processing overhead when these transactions are created.
- These limits must start at the beginning of the procurement life cycle and not be restricted further down the chain, keeping in mind there are also Receivers, Invoices, Payment Requests, and Fixed Asset transactions that will have commodity lines from these early procurement transactions.

Quantify Maximum Line Limit Property Value Settings

The following factors help to quantify the MAX_LINE_LIMIT property values for various transactions and set them on DCREQ.

- The size of the transaction drives the memory usage of the transaction. There are various factors that drive the size of the transaction.
- Transaction structure is one factor that drives the size of a transaction. The number of components a Transaction Type has drives the size of the transaction. Different Transaction

Types have a different number of components and hence the memory usage will be different for them. For example, the Purchase Order has Header, Vendor, Commodity and Accounting components whereas the Charge transaction has Header and Accounting components. These components may be loaded into memory when a transaction is processed. Further number of columns on a transaction component and extent of peripheral updates (business logic) it makes also drives memory usage.

- The second factor that drives the size of a transaction is memory usage by the connection objects that are used by the transaction. The number of prepared statements cached in the connection module drives the size of the connection object associated with a transaction. Complex transaction transactions such as Validate and Submit uses 2 connection objects when they are performed online and they use 1 connection object when they are performed offline. Two connection objects consume around 50 to 60 MB of VLS memory. Refer to the "Why Connection Routing ?" topic under "Application Server Administration" for more details on connection objects.
- Concurrent processing load on VLS
- Number of concurrent users online at a given point drives the processing load on a VLS. 60-80 users on a VLS with 1GB of memory is a normal figure.
- The setting of Number of jobs that can be concurrently run offline drives the processing load on the VLS. Setting 2-3 jobs to be run concurrently on a batch VLS with 1GB is a normal figure.

Certain Exceptional Cases

Maximum line limit edit was introduced to restrict the number of lines that can be contained in a transaction component in a single transaction. There is a rare possibility, though maximum line limit is enforced on a transaction component, that a component contains (maximum line limit + 1) number of lines. Here is an explanation for such situations.

Assume Maximum line limit for PO_DOC_COMM is set to *100* on the Transaction Component Requirements table.

Case 1: There is an edit in the Transaction Engine that throws error and disallows the insert when a 101st line is being inserted and saved on a PO transaction that already has 100 commodity lines. Here is a rare case where the Transaction Engine on insert and save of the 101st line does not throw any error because of concurrent users adding and saving lines to the same transaction at the same time. Consider the following sequence of events happening in increasing scale of time.

- User A opens a PO transaction in edit mode. The transaction already had 99 commodity lines.
- User B opens same PO transaction in edit mode.
- User A selects "Insert New Line" to add 100th line into Commodity component of the transaction. User A starts entering data into 100th line.
- User B selects "Insert New Line" to add a line (his 100th line) into Commodity component of the transaction. Page (PLS) side edit does not throw an error because it sees the number of existing lines for this component is 99. Note that this is not the 101st line because User A has not yet saved his newly inserted line and hence that line is not available in the database.
- User B enters data into the newly inserted line.

- User B selects Save for his newly inserted line and at the same User A selects Save for the newly inserted line. Behind the scenes Transaction A is started for User A's save and Transaction B is started for User B's save and both transactions are executing concurrently. While performing the maximum line limit edit both the transactions finds the number of existing lines on the Commodity line as 99 and allows the save. As a result the commodity component can end up having 101 lines, that is, (Maximum line limit value + 1).

This is rare situation that happened because of concurrent users adding and saving lines into the same transaction at the same time. There is no added business value to make system implementation (code) complex to avert this situation. The net effect for such rare situations will be that an additional line beyond the maximum line limit gets added to the component.

Working with Transaction Signatures

Transactions collect information into a single form where data is entered and is then submitted for processing. As mentioned earlier in this guide the transaction submission can be done directly by any user or can be routed through a workflow. At every stage of transaction Submission or Approval the system provides a mechanism to electronically sign its exact replica. While it is not necessary to sign transactions, signatures provide a powerful, automated means of certifying and validating what transaction data a user had actually submitted or approved prior to moving to the next phase of pending or finalization. In this respect, transaction signing is typically implemented for critical or highly visible transactions. This section explains how transactions can be configured and setup for signing.

In their basic form, electronic signatures are used to achieve the following results:

- **Intent** - signifies intent, approval of terms and confirmation that the signer reviewed and approved the content.
- **Identity** - identifies the person signing and indicates the action taken by the signer.
- **Integrity** - guards the integrity of the transaction against alteration.
- **Non-Repudiation** - ensures that a party to a transaction cannot deny the authenticity and origination of the transaction.

The system provides two ways of signing transactions electronically; they are *Digital* and *Non-Digital Signatures*.

- **Non-Digital e-Signature** is a light weight mechanism for signing the transactions. The Advantage transaction is exported to a PDF file and is then stored in the system for future reference.
- **Digital e-Signature** provides a cryptographic means for signing the transactions. The Advantage transaction is exported to a PDF file, and the User's Signature stamp is applied to it and then stored in the system for future reference. The signature stamp on the PDF maintains the integrity and authenticity of the PDF. If the signed PDF is tampered then the signature stamp appears invalid.

Some of the common terms of CGI Advantage Transaction Signatures that are used throughout this chapter are explained below:

- **Signature Rule** – Determines which, if any, type of signature needs to be applied to the transaction by the user before submitting it to the next phase.

- **Signature Condition** – Used to construct signature rules. Each condition evaluates up to five fields and values to construct a condition for requiring transaction signature. Refer to the "Manage Approval Conditions" topic in the *Workflow and Collaboration User Guide* to understand how to setup conditions.

- **Signature Type** – For every signature rule, this field determines which type of signature should be applied to the transaction (that is, *Non-Digital e-Signature* or *Digital e-Signature*).

Activating Transaction Signatures

When you set up a transaction for signing, you can define rules that require the transaction to be signed digitally or non-digitally. The following topics explain how transactions can be configured and set up for signing.

- Transaction Layout Generator Utility
- Transaction Signatures Global Switch
- Transaction Signature Rules

Transaction Layout Generator Utility

Signatures are applied to PDF transactions which are exported from the actual transactions. This PDF is a replica of the transaction in terms of the field labels, values and their layout. Transaction Layout Generator Utility is a standalone utility which generates and maintains metadata about the transaction. The CGI Advantage System Administrator should execute this utility whenever there is a change in the transaction layout. Refer to the "Transaction Layout Generator Utility" topic for instructions on how to run this utility. This utility should be executed in following cases:

- Site new to Advantage and is setting up Advantage for the first time.
- Sites upgrading from previous releases
- Sites having custom transactions.
- Sites applying defect fixes specific to transaction layouts.

Transaction Signatures Global Switch

The Transaction Signatures feature can be enabled or disabled globally for all transactions. To enable this feature the DocumentSignaturesEnabled parameter in the CGI Advantage initialization parameter file (Adv30Params.ini) should be set to *true*.

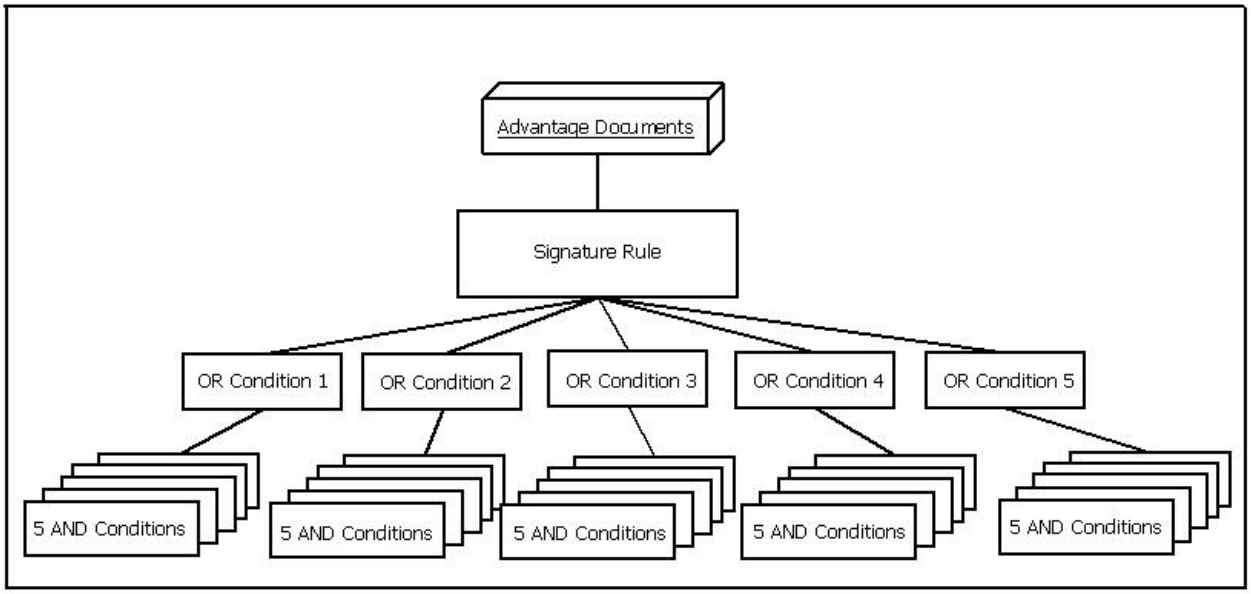
Transaction Signature Rules

The signing of transactions in CGI Advantage is governed by rules defined on the Transaction Signature Rules (DSIGRULE) page. A transaction signature rule is uniquely identified by the Transaction Code, Signature Type and the Organizational elements. For any Transaction Code, if a rule is defined that matches the exact values for the Organizational elements, the transaction is selected for signature processing. Rules can also be defined using wildcard characters for the Organizational elements. If no exact match is found then the rule selection follows the Approval Hierarchy as described in the "Approval Hierarchy" topic in the *Workflow and Collaboration User Guide*.

Once the applicable Signature Rule has been determined, it is evaluated to determine if a signature is required. If an applicable Signature Rule is not found, then the transaction does not require a signature. Signature Rules are evaluated during transaction processing for user initiated actions (for example, Transaction Submit, Transaction Approve and Transaction Bypass Approval). The transaction signing feature is only available for user initiated actions and not for processes that submit transactions automatically, such as, System Maintenance Utility or similar batch processes. Therefore, these rules are not evaluated during automated transaction submission.

The Transaction Signature Rule is uniquely identified by Transaction Code, Signature Type and the Organizational elements. Signature Type can be set to one of the following values: *Digital e-signature* or *Non digital e-signature*. Therefore, there can be two rules for a specific combination of Transaction Code and Organizational elements. In this situation, the DocumentSignatureRulePreference parameter is used to determine the Signature Type that should be used for signing the transaction. This parameter is defined in the CGI Advantage initialization parameter file (Adv30Params.ini).

Each Signature Rule can have up to five OR conditions. Each OR condition can be formed by a set of five AND conditions. A typical Advantage Transaction Signature Rule is shown in the following figure.



Each OR condition is defined on the Manage Approval Conditions (IWF07) page. Refer to the “Manage Approval Conditions” section to understand how the conditions are defined and evaluated.

Signing Transactions

Once a transaction rule is evaluated and the transaction qualifies for signature then the complete transaction is exported to a PDF file. This PDF transaction is an exact replica of the transaction with respect to the field values, their layout and positioning. The PDF transaction is shown to the user in a pop-up panel on the same window. This panel has a legal declaration and a password field. The user will have to accept the legal declaration and enter his/her system password before signing the transaction.

The legal declaration seen on the pop-up window is configurable. The legal declaration title is read from the DocumentSignatureLegalDeclarationTitle parameter and the actual declaration is read from the DocumentSignatureLegalDeclaration parameter. Both parameters are defined in the CGI Advantage initialization file (Adv30Params.ini).

Once the User signs the transaction, it is moved to next phase depending upon the action performed.

Transaction Archiving

Transaction Archiving is an important feature of CGI Advantage as it reduces the volume of transaction data in the system for better usability and performance. It also removes unwanted transaction historical data from the system, allows users to audit, store and retrieve previously archived transactions, and allows the system to reuse transaction IDs when this functionality is required.

Transaction archiving provides the following functionality:

1. Removes unwanted historical transaction data in the system.
2. Reduces the number of transactions to help improve system performance and reduce the size of the system database
3. Provides for the ability to reuse transaction IDs by removing existing transactions with that ID from the system
4. Provides, through the Transaction Archive Catalog, a list of archived transactions available to be retrieved and restored
5. Allows the transaction archive action to be used both as an online action as well as a batch process
6. Allows specification of the archive storage medium as either a database table or the file system
7. Archives transaction attachments along with the transaction when archiving the last transaction version of a transaction collection

The archive action on a transaction copies the selected version(s) of a transaction into a specified storage medium and removes it from the system. The storage medium can either be a file system or a database table. When a transaction version is archived, the transaction header, components, and catalogs of that transaction are archived and deleted from the system. Additionally, any object attachments that are associated with the transaction are also archived if the transaction version being archived is the last remaining version for that transaction collection in the system. When the archive action is completed, an entry is created on the Transaction Archive Catalog to retain a footprint of the archived transaction version. This archive footprint can be used later to search the list of archived transactions for a specific version based on user specified search criteria. From the Transaction Archive Catalog, a previously archived transaction can be restored back into the system by using the unarchive action. This action restores all of the archived data for the selected transaction version(s) from the storage medium back into the system. Finally, the archive historical action is used by the System Maintenance Utility (SysManUtil) job to select and archive all of the historical transaction versions for a given transaction collection. Refer to the "Transaction Archive Catalog" topic in the *Transactions User Guide* and refer to the "Working with System Maintenance Utility" topic in the *System Processing User Guide* for more information.

Refer to the following topics in this user guide, for more information:

- Archiving to the File System
- Archiving to a Database Table

Archiving to the File System

One common archive storage medium in the CGI Advantage system is file-based storage. This option is set in the ADV30Parms.ini system initialization file at installation time. Compressed ZIP files are created and used to store the transaction data and other related information. The ZIP file format specification is used as it is a platform-independent file format that aggregates many files into one and it also allows for compression of the file. The ZIP files contain the XML file for the transaction data and also other files associated with the transaction, including transaction attachments. These files are stored in the file system directory specified in the setup file ADV30Params.ini that was set up at installation time.

When a transaction version is archived to the file system, two possible ZIP files can be created. One zip file contains all of the transaction data, including any related workflow data, in the form of XML and is named as follows: “D” + archive number + “.ZIP”. For example, a transaction data archive ZIP file may be named as “D3560.ZIP”. The other file that is optionally created is the transaction attachments archive file, which contains all of the related attachments for the transaction if the transaction version being archived is the last for the transaction collection. This file is named as follows: “A” + archive number + “.ZIP”. For example, if this file was created with the previous example archive file, it is named “A3560.ZIP”. The archive number that is assigned to the files is sequentially generated using the unique number generator. Since this number is used by the system as a pointer to the physical archive files, the transaction archive files must not be renamed.

Note that the system must be set up to archive to the File System in order to use the Batch Transaction Archiving process.

Archiving to a Database Table

Transactions can also be archived to a database table as another form of archive data storage. This option is set in the ADV30Parms.ini system initialization file at installation time. In this setup, XML data is generated for the transaction version being archived, along with any related transaction attachments, and this is stored in a database table, IN_ARCH_STOR, as BLOB data. This table is used to store archived data for database based archive processes.

For performance and database sizing reasons, this table can be created in another database instance or schema rather than in the production database that contains the rest of the CGI Advantage system tables (the default). This has the added benefit of not impacting the production system when the archive table is backed up/restored and it does not impact the size of the production database. To do this, simply deploy the IN_ARCH_STOR table to another database instance or schema using the CGI Advantage Studio. Also make sure that you change this data object’s data server assignment in the VLS console to point to the new database instance or schema.

Unique Number Generation

This topic describes the unique number generation concept of the CGI Advantage system. This topic details the administration of unique numbers by system administrators. Unique numbers used in CGI Advantage are assigned based on entries maintained in the unique number table. The terms “unique numbers” and “sequence numbers” have been used interchangeably in this section.

Unique numbers are maintained by the system almost transparently to the user. A unique number request is handled at the system level instead of relying on database generated unique numbers or sequences. Unique numbers are assigned from the unique number table and can be set up so that each request for a unique number does not necessarily mean an additional database request. Instead, a set of pre assigned numbers can be reserved in the application server in memory and allocated on request. A database request is made only when the set of numbers has been exhausted or the application server is recycled.

Unique number request servicing Java methods are synchronized and the unique number table row is locked at the database level to ensure that there is no conflict in an environment supporting multiple application servers requesting unique numbers for the same unique number sequence name.

Refer to the following topics for required setup:

- Unique Numbers
- Initialization File Settings

Unique Numbers

The Unique Numbers page allows the system administrator to modify the last sequence number used for a sequence as well as set the block size of unique numbers that should be loaded in the application server.

- The **Sequence Name** specifies the name of the sequence and uniquely identifies the sequence. The reserved Sequence Name of ALL is the default sequence and it is used if any application request for a unique number is made without specifying the Sequence Name.
- **Sequence Last ID** specifies the last assigned unique number for this particular sequence.
- **Sequence Block Count** specifies the set of numbers that should be assigned to the application server on each unique number request to the database. A database request for the next unique number block is only made after all of the unique numbers have been exhausted on the application server.

Unique numbers used in the system are generated based on entries maintained in the unique numbers table. Take care whenever you update the Sequence Last ID to ensure that duplicate unique numbers are not generated by the system. The Sequence Block Count for any unique number sequence should be set after consideration of the transaction behavior in any installation. The Sequence Block Count indicates the range of unique numbers each logical application server keeps in memory before requiring a database call to the TABLE_NM to get a new range. Thus, if the range is set too high and the application server is shut down for any reason, the unused range of numbers kept in memory can no longer be used. For example, if a value of 1000 was used and three logical application servers requested a unique number range and then were immediately shut down, you could waste a range of 3000 unique number values. Thus, you must consider how quickly unique numbers may be wasted, in addition to considering the performance gains achieved by setting a high value for Sequence Block Count.

Initialization File Settings

The TABLE_NM parameter in the system's initialization file, ADV30Params.ini, specifies the table in which the unique numbers are stored. If a sequence name is requested that does not exist, a new sequence is created and the DEFAULT_BLOCK_SIZE is used to store a default sequence block count.

```
##########################################

## Unique Identifier parameters

##########################################

TABLE_NM=R_IN_UNID

DEFAULT_BLOCK_SIZE=1
```

HRM Specific Setup

This chapter includes setup tables that are only included in Advantage HRM.

HRM Data Dictionary Tables

Refer to the following topics (listed alphabetically) for information regarding the Data Dictionary tables that must be set up in HRM:

Column

The Column table is one of the six HRM Data Dictionary tables. This table defines each field in a table, including its position and attributes.

Column by Name

The Column by Name table is one of the six HRM Data Dictionary tables. This table defines the location within a record of a page entry and its corresponding column position.

Index

The Index table is one of the six HRM Data Dictionary tables. This table defines the key structure for each table.

Index Column

The Index Column table is one of the six HRM Data Dictionary tables. This table defines information about table keys.

Table

Table is one of the six HRM Data Dictionary tables. This table contains a list of all of the tables defined to an application by table identifier and physical characteristic.

Table Directory

Table Directory is one of the six HRM Data Dictionary tables. This table contains a list of all of the tables defined to an application keyed by table name.

Application Access Definition

This page allows you to identify the valid Application Access Ids for a particular installation. It also specifies whether each table for each Access ID resides on disk, is memory-resident, or is transient. All fields on the Application Access Definition table should be maintained using the Begin Day utility; other sources of input may generate unpredictable results. Refer to the *Utilities Run Sheet Guide* for your application for information on the Begin Day utility. This table is only applicable to Advantage HRM.

Begin Day

The Begin Day (BDAY) page allows you to identify the table numbers of the system control tables that control information, indicate the locations of various tables and control fields, and provide other default information. This table can also be used to instruct system security to interpret the system Terminal ID and to identify the sizes of the Level 1 and Level 2 Security entries. The records are created using the BEGIN, CNTL, and TABLE control card inputs to the BGNDAY utility. This table is only applicable to Advantage HRM.

Boot Definition

The Boot Definition (BOOT) page contains application start-up (bootstrapping) information needed by DBIO at initialization time. This table is only applicable to Advantage HRM.

Database Extension Table

The Database Extension Table allows you to specify database-specific override information to the data dictionary. This enables overriding of the COBOL-oriented field names, field types, and table names. This information is stored in the Database Extensions Table, and is used to create the changes file required as input to the SQLMAKE Data Definition Script generation process. This table is only applicable to Advantage HRM.

Descending Key Filter Parameters

The Descending Key Filter Parameters page allows you to invert the data in selected fields within a table record so that those fields are stored in descending order. The page is used to identify the fields to be inverted by the Descending Key Filter. This table is only applicable to Advantage HRM.

Error Message Explanation

The Error Message Explanation (EMEX) page allows users with the appropriate privileges to create and define error codes and messages.

Extended Begin Day

The Extended Begin Day page contains information needed to determine subsystem table numbers and table numbers for advanced features such as the Front End and logging in Advantage HRM.

File Definition

This page specifies the physical characteristics of the database files in which the control and application tables are defined. This table also indicates how tables are stored on the database files and defines the structure of the application database. This table is comprised of two pages, File Definition (1 of 2) and (2 of 2); Database-specific information is displayed on database-specific File Definition screens (File Definition for ADABAS, File Definition for DB2, and File Definition for VSAM). This table is only applicable to Advantage HRM.

Filter Parameters

This page allows you to direct I/O through multiple layers of filters and tables. A single DBIO call may pass through multiple filters before it is completed. The initial filter is determined by the table definition of the table referenced in the DBIO call. That initial filter and subsequent downstream filters typically determine the next filter to be invoked by consulting one of the Filter Parameters tables. Filters that need additional parameters, such as Descending Key and Sparse Index, obtain the next filter specification from a Filter Parameter record on a specialized table maintained by dedicated screens such as the Descending Key Filters (DKEY) page. Other filters that do not require additional parameters share a single Filter Parameter Table maintained by the Filter Parameters page. This table is only applicable to Advantage HRM.

Memory Resident Keys

The Memory Resident Keys page allows you to store information about which rows of a table to load into memory while grouping tables into sets by Task ID which the DBSHMGEN utility uses to determine which tables and which rows to load into memory. This table is only applicable to Advantage HRM.

Memory Resident Load-Order

The Memory Resident Load Order page allows you to store information about memory resident tables and group tables into sets by Task ID which the DBSHMGEN utility uses to determine which tables to load into memory. It also indicates the maximum amount of memory to use, which order to load the tables, and whether the table should be entirely or partially loaded into memory. This table is only applicable to Advantage HRM.

Table Definition

The Table Definition page defines the logical characteristics of system control and application tables in Advantage HRM

Memory-Based Data Access

CGI Advantage provides the ability to keep mostly static reference data in memory for fast and efficient access. It supports two varieties of this feature: Memory Resident Table (MRT) and Server Data Object Cache (SDO). The technical implementation of these two features is quite different despite the conceptual similarities.

It is more efficient for the system to load frequently accessed but infrequently updated tables into memory. Any data requests for such tables can be serviced from memory instead of making expensive database calls. Such tables in the system can be marked as being either MRT or SDO.

The following topics describe the MRT and SDO features of the CGI Advantage system. The topics also detail what system administrators need to be aware of when a table has to use either the MRT or SDO feature.

- Memory-Resident Tables (MRT)
- Server Data Object Caching (SDO)

Memory-Resident Tables (MRT)

MRT provides fast access to mostly static reference data. This is achieved by loading the data of all the memory resident tables into memory whenever the first request is received for accessing any MRT data. The table data is maintained in memory at the application server layer which allows the server to bypass database calls altogether whenever a request to retrieve data is made. This one time front-loading of MRT data ensures that no further database calls are required for these tables since the application server can service data requests from memory.

The data maintained in MRT's memory image is static and is loaded from a flat file which reflects a snapshot of the data in the corresponding database table. The memory image is not updated unless a new "snapshot" of data is generated from the database and the MRT is reloaded by restarting the application server. The database table that corresponds to an MRT can be updated under some conditions (described later), but those updates are reflected in the MRT only when a fresh snapshot is taken and the application server restarted, as mentioned above. Therefore, tables that undergo frequent updates are not included in MRT. Only the security tables listed in the table below are added into MRT, and no other tables may be placed into MRT.

The following security-related tables are placed in the MRT file:

Data Structure	Database Name
Application Resource QueryObject	R_SC_APPL_RSRC_QRY
Access Control (SCRACS) table	R_SC_RSRC_ACCS
Security QueryObject used internally, required	q_CombOrgCode
Secured Field Resources (SCFUSE) tables	R_SC_RSRC_FLD

Field Access (SCFACS) table	R_SC_FLD_ACCS
Field Comparison	R_SC_FLD_CMPR
Foreign Organization (SCFORG) table	R_SC_FGN_ORG
User Interface Field Security (SCUIFS) table	R_SC_UI_FLD_SEC

To ensure data consistency between the MRT data and the underlying database, actions such as insert, update, and delete are typically not allowed on data objects that are memory resident. At the same time, there is a need to have pages that allow the data to be set up in those tables. Hence, a mechanism is provided whereby a developer building pages can set the AMSDataSource's MaintenanceString property to Y to indicate to the system that all updates on that page should go to the database, even though it is to a table that is loaded in memory as MRT.

- Pages can be developed which allow users to update the contents of a memory resident table directly on the database. This facility is ideal for tables with infrequent updates where there will not be any application inconsistencies caused by the fact that MRT must be regenerated and the application server recycled before the changed table contents come into effect. This feature is possible only if the AllowMRTUpdate flag is set to true for the specific data object referenced (AllowMRTUpdate flag should be globally set to false in production).
- To bypass running the MRT Generation process and bouncing of the application for any page listed above (excluding Application Resource QueryObject which will always needs to be read from MRT), you must add the corresponding Database Name of the page (listed above in the table) to the BypassMRTLookupList parameter in the ADV30Params.ini file. You must then bounce the application for the changes to take effect. Refer to the "Activating Security Information" topic for more information.

MRT Setup and Usage

If a table is in MRT, care must be taken to verify that any new pages referencing that table function as developed after the new MRT file takes effect. The developer must set the value of the MaintenanceString property on AMSDataSource to Y to ensure that all updates on that page go to the database if this is not a read-only page.

Refer to the following topics for additional setup and usage information:

- Initialization File Settings
- Memory Resident Table File Generation
- Overriding Table Updates on MRT

Initialization File Settings

The system's Initialization Parameter file (ADV30Params.Ini) has entries specifying the location from which MRT data will be read. The same location is used by default when generating the MRT file; although, the user can override and specify a new MRT file location and MRT file name. MRTFileLocation specifies the MRT directory, MRTFileName specifies the actual MRT file name, MRTGenerationRole

specifies any role other than ADMN, which can generate the MRT file. The MRTGenerationRole property in the system's Initialization Parameter file (ADV30Params.ini) is only applicable to the online MRT Generation process and does not apply to Job Based MRT Generation.

```
##########################################

## Memory resident tables parameters

##########################################

MRTFileLocation=<ADV_HOME>/RTFiles/VLS1

MRTFileName=ADVMRT.dat

MRTGenerationRole=
```

An attempt is made to load the MRT file when the application server starts up and a call for data on MRT is made. If no MRT file is available, all data access is done at the database level. Security access is linked to MRT; only users with ADMN roles are able to use the system if a MRT data file is not available at application server start up. An ADMN user can then generate the MRT file and recycle the application server; the new MRT file is then loaded to memory.

Memory Resident Table File Generation

The system has the ability to generate the MRT file from the MRT Generation job in the Advantage job framework and online through the Generate MRT App (GMRT) page. Both methods produce the same output and require that the application server be restarted in order to reload generated MRT file. Refer to the "Generate MRT - Security" topic in the *Security Administration Guide*, for information about the Generate MRT - Security (GMRT) page.

The MRT Generation job allows authorized users to generate the MRT file offline. The MRT Generation job is located in the Administration application under the Administration > Utilities area of the Batch Catalog. This job is re-runable but not re-startable.

The MRT Generation job has the following parameters:

- **Generate MRT File Location** - This parameter defaults to the value from the ADV30Params.ini file during execution if left blank. However, this is an editable parameter and the user can specify a value different from the default.
- **Generate MRT File Name** - This parameter defaults to the value from the ADV30Params.ini file during execution if left blank. However, this is an editable parameter and the user can specify a value different from the default.
- **Generate MRT Mode** – This parameter has an option of two values: *Production* and *Debug*. If *Production* is selected, the data file generated by this process is usable by the application server but is not formatted for easy viewing. If *Debug* is selected, the data file generated in this process is unusable by the application server but is formatted for easy viewing.

Overriding Table Updates on MRT

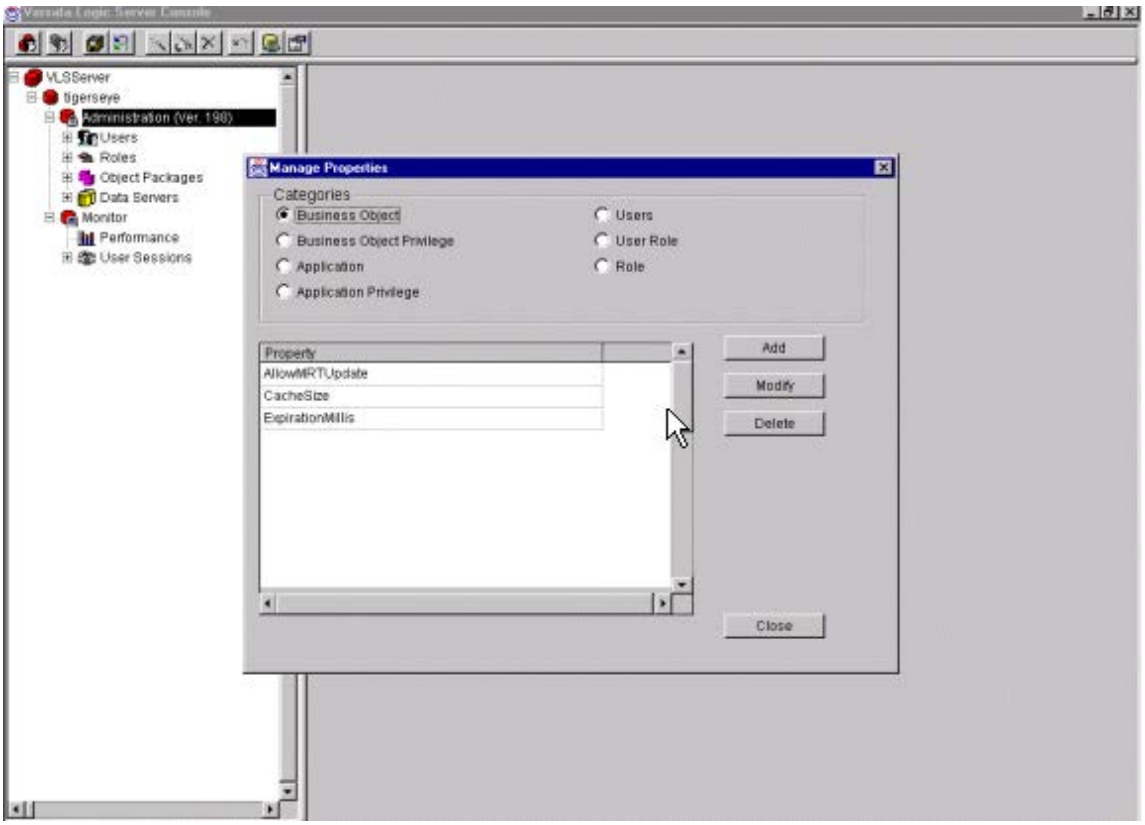
The system administrator can specify that no updates are allowed on a particular memory resident table, though a page has been developed using the AMSDataSource MaintenanceString parameter to redirect updates to the database. This feature is useful when you want to turn off or turn on database updates for

memory resident tables for a fixed duration. Setting or resetting this property requires the application server to be recycled for it to take effect.

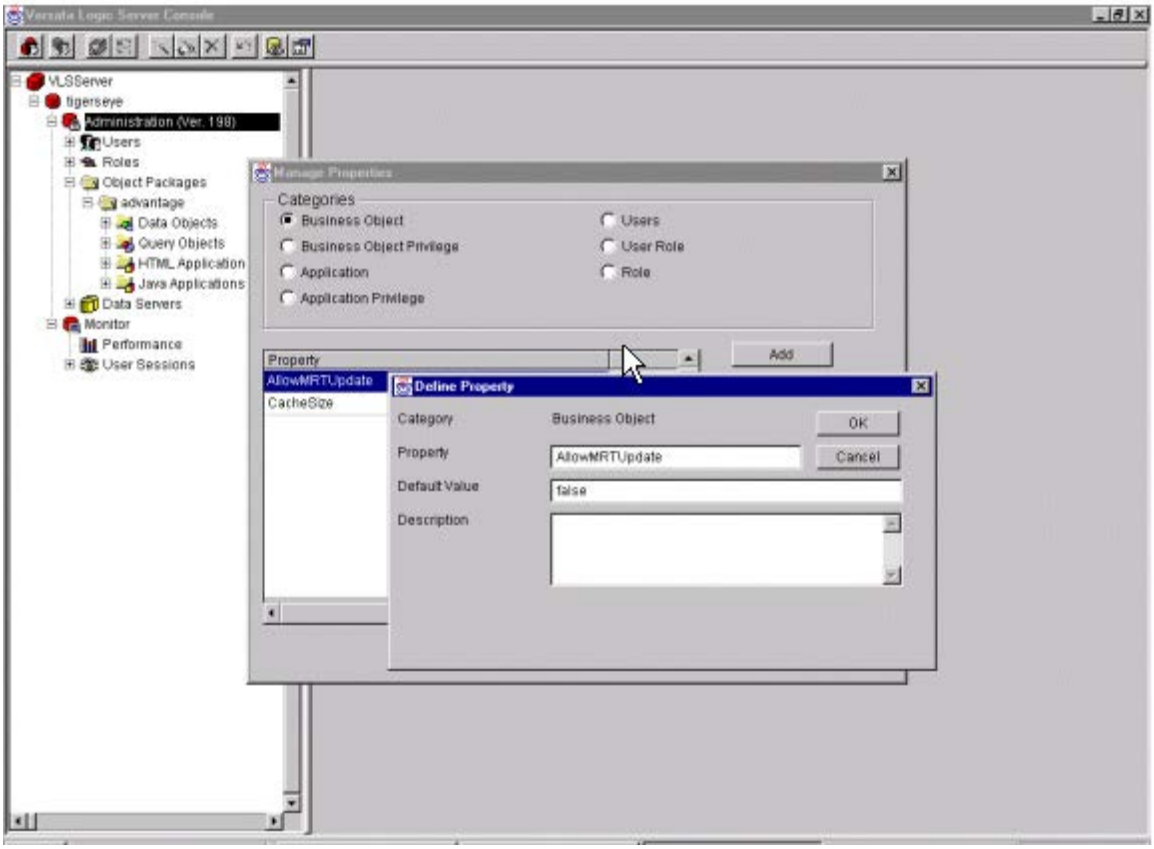
In a production environment, it is recommended to set AllowMRTUpdate to *false* in the application server console for all data objects. This means that in general no updates are allowed on memory resident tables. The only way to update memory resident tables in a production environment is to reset the AllowMRTUpdate flag to *true* for that data object, recycle the application server, modify the data, reset the AllowMRTUpdate flag back to *false* and bring the application server back up again.

The AllowMRTUpdate flag is not applicable for the MRT generation process. The Generate MRT page always retrieves its data from the database when building the MRT file, irrespective of the AllowMRTUpdate setting.

- Click the **Property Definitions** button in the application server console to access the Manage Properties screen.



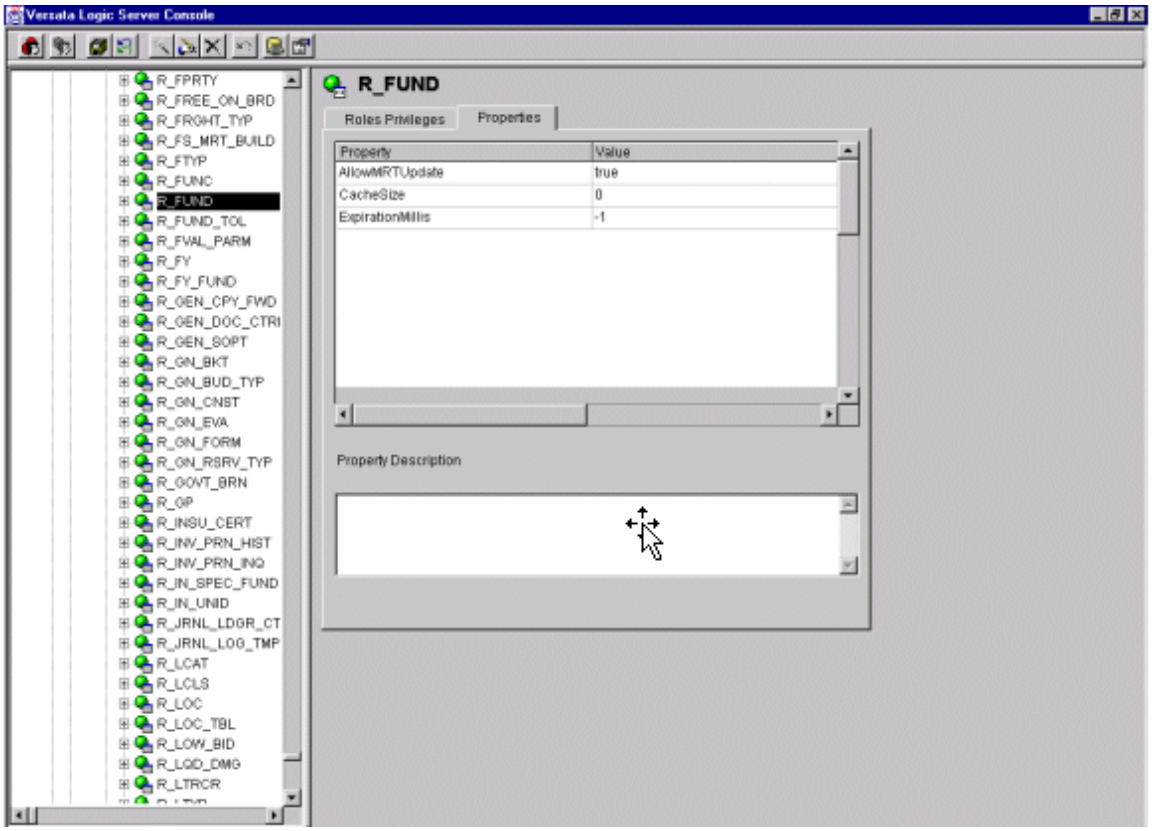
1. Click the Modify button to modify any settings.



It is recommended that the default settings be retained and individual changes made at the specific data object level. The AllowMRTUpdate field is set to false by default.

- To modify the AllowMRTUpdate field setting for a specific data object, expand Object Packages under Administration. Then open the data object under the Advantage application and modify the specific properties for this data object.

- Setting the AllowMRTUpdate field to *true* for a specific data object ensures that updates are allowed on this memory resident table when a page has the AMSDataSource property MaintenanceString set to *true* to redirect updates to the database.



Server Data Object Caching (SDO)

The Server Data Object Caching (SDO) feature of the CGI Advantage system provides fast access to mostly static reference data. This is achieved by maintaining an internal cache of data objects in memory at the application server layer. This allows the server to bypass the database altogether when a request to retrieve data is made.

The SDO cache is dynamic in nature; it is populated as requested data is retrieved from the underlying database. This allows the SDO cache to automatically grow as requests are made for non-cached data. To keep the overall size of the SDO cache in check, a Least Frequently Used (LFU) algorithm manages the SDO cache by removing infrequently accessed data from the internal cache.

To ensure data consistency between the SDO cache and the underlying database, insert, update, and delete actions are not allowed on data objects while they are set to participate in the SDO cache. It is important for system administrators to keep this in mind when identifying a data object that could participate in the SDO cache.

Only data object getObjectByKey() calls are serviced by the SDO cache mechanism. If a matching row is found in the cache it is returned; otherwise, a call is made to the database and that row is then added to the data object's cache to service any future requests. This lazy loading SDO mechanism ensures that the most frequently accessed rows of a table are loaded in memory first.

Refer to the following topics for additional information:

- SDO Considerations
- SDO Setup and Usage

SDO Considerations

After you decide that the system may benefit from having a table loaded in memory, you must decide whether the table should be included in SDO cache. The following guidelines can be followed in this analysis:

- An SDO cache table can be read-only during an application server session. This means that only pure read-only tables are good candidates for SDO as the updating of SDO tables is more involved, requiring that the application server be recycled twice.
- Only data object getObjectByKey() calls can take advantage of the SDO caching mechanism.
- Query objects cannot be set up to be SDO cached.
- Tables with a large number of rows may benefit by following the SDO cache mechanism along with expiration logic as a limited number of rows can be stored in memory forcing database calls for the less frequently referred rows. Incorporating expiration logic is not normally recommended.
- Only a finite number of tables can be marked to be memory resident or be stored on the SDO cache because of memory limitations. Careful estimation of the memory requirements is required before marking a table for SDO. Factors to consider are the number of rows and record size and typical growth in record access rates for SDO.

SDO Setup and Usage

The SDO cache feature has been designed to be transparent to the end user while providing fast access to reference data. Data object getObjectByKey() calls participate in the SDO cache whenever appropriate if the SDO cache feature is activated for the data object.

The SDO cache setup ensures that every row accessed is first checked for in the cache. If it is not found, the row is retrieved from the database and added to the cache. Future requests for that row are serviced from the memory cache instead of an expensive database call. This is especially useful for read-only tables. Transaction processing tables with insert/update/delete requests are not viable candidates for SDO caching.

It is important that the system administrator be aware of the restriction of insert, update, and delete actions when setting up a data object to participate in the SDO cache. Inserts, updates and deletes are not allowed by any application code that interacts with an object if the data object is designated to be included in the SDO cache. A Server Exception is thrown with the appropriate error message whenever any insert/update/delete attempt is made. This is normally not an issue for the system administrator because the cached data objects most likely reference data that is infrequently, if ever, updated. If an update is required at some point in time, the data object setup must change to take the object out of the SDO cache for that period to allow the update.

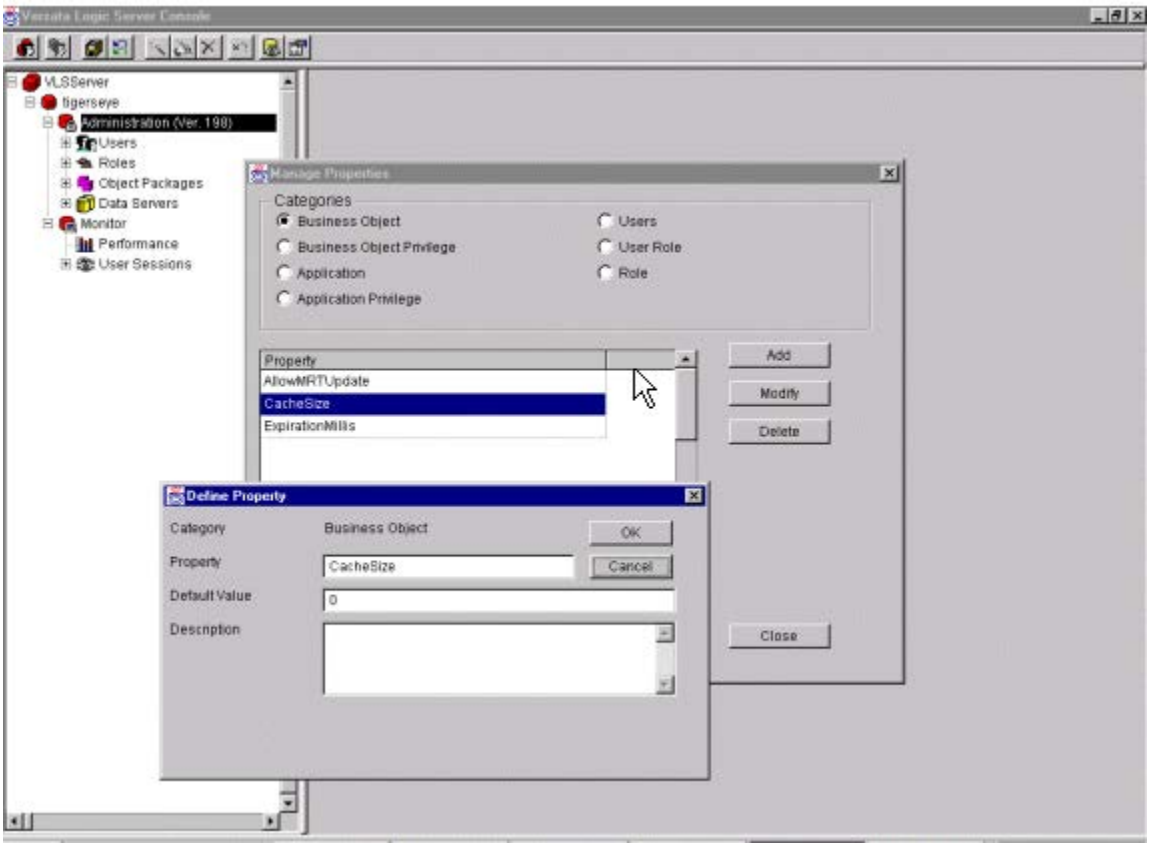
Refer to the following topics for additional information on SDO setup and usage:

- Defining Application Server Console Properties and Settings

- Updating Data in SDO Tables
- Mutual Exclusiveness of MRT and SDO

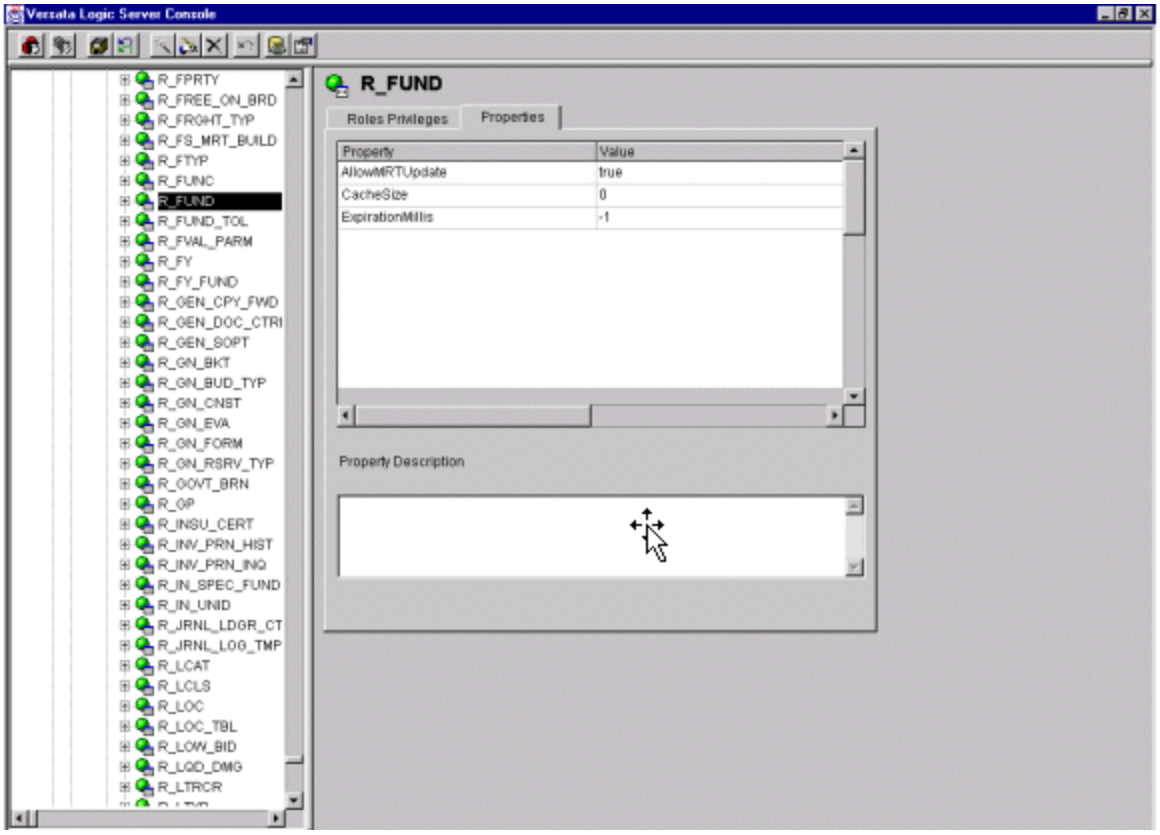
Defining Application Server Console Properties and Settings

A default CacheSize of 0 is specified at the system level which means that SDO caching is not set for any data object. This is set in the Manage Properties section of the Business Object category.



Expand Object Packages under Administration to modify this setting for a specific data object. Then open the data object under the Advantage application and modify the CacheSize setting.

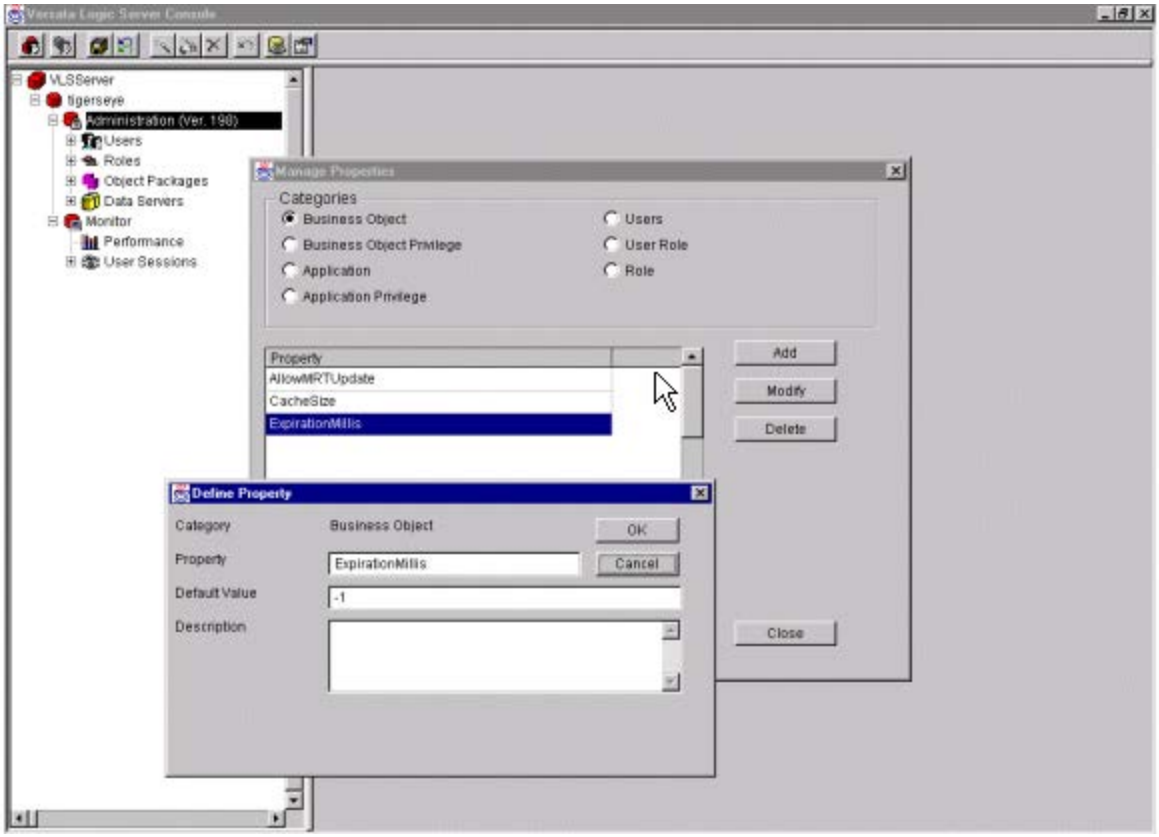
- Setting CacheSize to –1 for a specific data object ensures that all rows in this table are added to the cache. This approach is recommended as being suitable for most environments.
- CacheSize can also be set to a positive number in which case rows are cached until that number is reached after which all new row requests are directed to the database. This is not ordinarily recommended but can be set for specific data objects if required. Take care to ensure that no system functionality is modified because of this setting.
- A default ExpirationMillis setting of –1 is specified at the system level. This means that cached rows never expire during the life of the application server.



Expand on Object Packages under Administration to modify this setting for a specific data object. Then open the data object under the Advantage application and modify the ExpirationMillis setting.

- Setting ExpirationMillis to –1 for a specific data object ensures that no rows in this table are expired in the cache during the life of the application server. This approach is recommended as being suitable for most environments.

- ExpirationMillis can also be set to a positive number in which case all cached rows of the data object are expired after the specified milliseconds. This approach is not normally recommended but can be set for specific data objects if required. Take care to ensure that no system functionality is modified because of this setting.



Updating Data in SDO Tables

Modification of the contents of a SDO cached table is not allowed. The only way to modify the contents of a SDO cached table is to define it as being non SDO by setting CacheSize to 0 for that data object, recycling the application server, modifying table contents, resetting CacheSize back to its original setting and again recycling the application server. Since this procedure involves multiple application server recycles it is recommended that you simulate and adequately test property changes in a test environment before applying them in the production environment.

Mutual Exclusiveness of MRT and SDO

If a table is in MRT, then the AllowMRTUpdate flag is set to *true* by default and the CacheSize must be set to *0* (rather than –1). Setting the CacheSize to anything other than 0 turns on SDO (-1 is unlimited cache size), which is mutually exclusive with MRT.

Database Optimization Hint

Database queries generate result sets that contain all rows, from the requested tables, that match the filtering constraints. When most rows will not be needed, the work to generate the much-larger result set and the resources to handle such a result set are not required. Eligible CGI Advantage pages enable virtual result sets to reduce the high memory overhead associated with open database cursors and connections. When the online user wishes to view a different "window" of records, CGI Advantage queries the table again to retrieve the virtual result set to display the appropriate set of records.

Database servers also provide the ability for queries to specify the number of records to be retrieved. Such a Database Optimization Hint allows the database server to generate a result set with only the records that are currently needed. For example, if a page displays a grid of records from a table, typically only ten to twenty are displayed at any one time. If the page will ultimately allow the user to view 1,000 records (whether the user will or will not), it is really not necessary for the database server to generate a result set with the 1,000 records.

Therefore, combining the Database Optimization Hint with virtual result set-enabled pages will optimize the retrieval of records from the database. The database server will quickly return a smaller result set of records to be displayed on the current page's "window." And when the user chooses "Next" to view the next set of records, the database server will return the next smaller set.

Listed below are sample SQL SELECT statements that incorporate the Database Optimization Hint for the identified database server.

- Oracle

  SELECT /*+ FIRST_ROWS(20) */ * FROM DOC_HDR ORDER BY DOC_CD

- Microsoft SQL Server

  SELECT TOP (20) * FROM DOC_HDR ORDER BY DOC_CD

The following topics discuss how to set up the Database Optimization Hint.

Guidelines for Using the Database Optimization Hint

Use of the Database Optimization Hint in selection queries on a page is recommended for pages that query tables with a large number of records. Incorporating the hint will be successful for pages (including pick pages) that satisfy all of the following conditions:

- The page enables virtual result sets. By default, most of the pages in CGI Advantage enable virtual result sets unless the behavior is explicitly not enabled during the development of the page. For example, the Disbursement Management page in CGI Advantage has explicitly not enabled virtual result sets within the logic of the page. Therefore, this page is not a candidate for using the Database Optimization Hint in selection queries.
- The page is not expected to be secured with row filtering for any of its record sources that will use the Database Optimization Hint during queries.

- Application-specific logic does not filter out records in the result set returned from the database server. (Technically, this logic typically occurs during the processing of the beforeResultSetFillBeforeSecurityCheck and beforeResultSetFillAfterSecurityCheck events.)

If all three conditions are not satisfied when the Database Optimization Hint is used for selection queries on the page, then the page will not display the expected records. Therefore, it is necessary to ensure that the page meets all of the above conditions when one contemplates using the Database Optimization Hint for selection queries on the page.

Enabling the Database Optimization Hint

Refer to the following topics for information on enabling the Database Optimization Hint

- General Setup for Pages
- Exceptional Setup for Custom Pick Pages

General Setup for Pages

To incorporate the Database Optimization Hint on a page that enables virtual result sets, follow these steps:

- Add the page that will incorporate the Database Optimization Hint to Page Preferences (PGPREF).
- Enable the “Use DB Optimization Hint” option for the page entry.
- If the selection queries for the page’s picks should also incorporate the Database Optimization Hint, then enable the “Use DB Hint on Page’s Pick(s)” option for the page entry. There are also special rules for custom pick pages, and they are described in the "Exceptional Setup for Custom Pick Pages" topic.

Note that changes on Page Preferences will not take effect until the application server is restarted.

In addition to the previous steps there are other levels of control for incorporating the Database Optimization Hint on pages that enable virtual result sets. These levels are the application level and the record-source and pick level, and they are described in the following topics.

- Application Level
- Record Source and Pick Level

Application Level

A value of true for the parameter VRS_DB_HINT_ENABLE on Application Parameters will enable the usage of the Database Optimization Hint on the pages appropriately set up on Page Preferences. Any other value will disable the usage of the Database Optimization Hint throughout the application, regardless of the settings on Page Preferences. Note that the application server will need to be restarted for a change to the value of VRS_DB_HINT_ENABLE to take effect.

Record Source and Pick Level

This type of setting is available during development time when a page is designed, and it is generally used to disable the usage of the Database Optimization Hint in selection queries for a particular record source or pick object on the page. As such, the information presented below is more technically oriented.

Each record source and pick object included on a page contains several properties to define the behavior of the record source and pick object. For the Database Optimization Hint the property for both is "UseDBHint," and the property is defined for the corresponding classes AMSDataSource and AMSPick, respectively. The default value for the UseDBHint property is blank which is treated as value of "true". Therefore, a value of blank or "true" means that the Database Optimization Hint will be incorporated in the selection queries for the page's record source or pick object based on the settings on Application Parameters and Page Preferences.

- For example, suppose the value of VRS_DB_HINT_ENABLE on Application Parameters is set to *true*, page X is defined on Page Preferences with "Use DB Optimization Hint" enabled, and page X has three record sources, A, B, and Y. Then a Database Optimization Hint will be added to the selection queries for all three record sources on the page. If the design of a particular record source Y on this page X states that no Database Optimization Hint should be incorporated regardless of the settings on Application Parameters and on Page Preferences, then the AMSDataSource property of UseDBHint for record source Y should be set to *false* in the CGI Advantage Studio.

- As another example based on the previous one, suppose "Use DB Hint on Page's Pick(s)" is enabled for page X on Page Preferences, and suppose that page X has three pick objects, C, D, and Z. Then a Database Optimization Hint will also be added to the selection queries for all three pick objects on the page (that is, when the query is processed to display the corresponding pick page for the selected pick object). If the design of a particular pick Z on this page X states that no Database Optimization Hint should be included regardless of the settings on Application Parameters and on Page Preferences, then the AMSPick property of UseDBHint for pick Z should be set to *false* in the CGI Advantage Studio.

Exceptional Setup for Custom Pick Pages

Handling pick pages is potentially more complicated than standard pages, since pick pages may be either system-generated pages or custom pages. Indeed, the settings on Page Preferences for the page that contains the pick object are still checked first. Then, depending on the settings, an additional lookup to Page Preferences might occur for a custom pick page. CGI Advantage uses the following steps to determine when the Database Optimization Hint should be incorporated in selection queries for a page's picks.

- As described previously in the "General Setup for Pages" topic, CGI Advantage first looks for a page entry on Page Preferences for the source page that contains the pick object selected by the user.

- If the entry exists and the "Use DB Hint on Page's Pick(s)" option is enabled, then the Database Optimization Hint will be incorporated in the selection query for the pick page, whether it is a system-generated or custom pick page.

- If the entry exists but the "Use DB Hint on Page's Pick(s)" option is not enabled, then the Database Optimization Hint will not be incorporated in the selection query for the pick page, again whether it is a system-generated or custom pick page.

- If the entry does not exist, then the Database Optimization Hint will not be incorporated in the selection query for a system-generated pick page. However, for a custom pick page, an additional lookup on Page Preferences is required.
- Based on the results from the previous step, CGI Advantage makes an additional lookup on Page Preferences, using the custom pick page name as the page entry.
- If the entry exists and the “Use DB Optimization Hint” option is enabled, then the Database Optimization Hint will be incorporated in the selection query for the custom pick page.
- If the entry exists but the “Use DB Optimization Hint” option is not enabled, then the Database Optimization Hint will not be incorporated in the selection query for the custom pick page.
- If the entry does not exist, then the Database Optimization Hint will not be incorporated in the selection query for the custom pick page.

Examples

Note that is assumed that all pages listed in the following examples satisfy all the conditions listed in the "Guidelines for Using the Database Optimization Hint” topic to be capable of successfully incorporating the Database Optimization Hint in selection queries. For changes on Page Preferences and changes to parameter VRS_DB_HINT_ENABLE on Application Parameters to take effect, restarting the application server is required.

- Transaction Catalog Page and All of Its Pick Objects
- Transaction Catalog Page but None of Its Pick Objects
- New Page and Some of Its Record Sources and Pick Objects
- No Database Optimization Hint

Transaction Catalog Page and All of Its Pick Objects

Set the following information to incorporate the Database Optimization Hint in the selection queries on the Transaction Catalog page and on the pick pages for Transaction Catalog’s pick objects.

1. Add an entry to Page Preferences for the page name of “Advantage.DocCatalog”.
2. Enable the “Use DB Optimization Hint” and “Use DB Hint on Page’s Pick(s)” options for “Advantage.DocCatalog”.
3. Set the value of parameter “VRS_DB_HINT_ENABLE” to “true” on Application Parameters.

Transaction Catalog Page but None of Its Pick Objects

Update the existing information for the Transaction Catalog page to incorporate the Database Optimization Hint in the selection queries only on the page itself and not on the associated pick pages.

Disable the “Use DB Hint on Page’s Pick(s)” option for “Advantage.DocCatalog” on Page Preferences.

New Page and Some of Its Record Sources and Pick Objects

A new page, TestOne, in the Advantage application will contain three record sources, A, B, and C and will also contain three pick objects for three system-generated pick pages, X, Y, and Z. It is known that during the design of TestOne the incorporation of the Database Optimization Hint is needed in the selection queries for only record sources A and B and only for pick objects M and N. Therefore, set the following information for this more technically oriented example.

- Add an entry to Page Preferences for the page name of “Advantage.TestOne”.
- Enable the “Use DB Optimization Hint” and “Use DB Hint on Page’s Pick(s)” options for “Advantage.TestOne”.
- Set the value of parameter “VRS_DB_HINT_ENABLE” to “true” on Application Parameters.
- In the CGI Advantage Studio, set the AMSDataSource property UseDBHint to “false” for record source C on page TestOne. Deploy the generated and compiled file.
- In the CGI Advantage Studio, set the AMSPick property UseDBHint to “false” for pick object O on page TestOne. Deploy the generated and compiled.

No Database Optimization Hint

Update the following information to disable the incorporation of the Database Optimization Hint in all selection queries for the entire application, without regard to data on Page Preferences.

Set the value of parameter “VRS_DB_HINT_ENABLE” to “false” on Application Parameters.

Listing of Tables Requiring Server Restart

Certain tables in Advantage need a Server restart to reflect any updates. This is because the data from these tables is loaded into memory on first access. Data for the succeeding requests are then fetched from the memory. This greatly helps in reducing the time spent on frequent database access.

Page	Underlying Table
Preferences Setup (PREFSTUP)	R_PREF_STUP
Application Parameters (APPCTRL)	IN_APP_CTRL
Setup Audit Log Control (AUDCTRL)	IN_AUD_CTRL
Leaf Fields (LEAF)	IN_LEAF
Application Servers (APPSRV) – Only a new entry needs a server restart.	IN_APPSRV_LST

Changes to Memory Resident Tables need a server restart. Please refer to the "Memory-Resident Tables (MRT)" section for the list of tables maintained as MRT.

Transaction Layout Generator Utility

Transaction Layout Generator Utility is used to generate and maintain layout and metadata for the CGI Advantage transactions in .json format files. The .json format file is used by the transaction renderer engine to render the transaction to a specified format.

An example of when these layout files are used is while signing the transactions. Transactions need to be exported to PDF format. These PDF files are then signed by the system on behalf of the system. The JSON layout generated using this utility is used to export such transactions to PDF format.

This utility is placed in the <AMSADV3_HOME>\Utils\DocumentLayoutGenerator folder.

The following steps should be used to execute this utility:

1. Open **AdvDocToPDFUtilParams.ini** and update the following paths:
    - **ADVANTAGE_REPOSITORY** => Path to the repository xml
    - **VLS_HOME** = Versata_Home\VLS\bin
    - Example:
    - ADVANTAGE_REPOSITORY = C:\\AMSFIN310\\dev\\advantage\\Source\\advantage.xml
    - VLS_HOME = Versata_Home\VLS\bin
2. Update the **APP_TYPE** property with the appropriate value.
    - 1 - Advantage Administration
    - 2 - Advantage Financial
    - 3 - Advantage HRM
3. Update the **TARGET_JSON_LAYOUT** property with an appropriate value. Make sure the same location is specified in the **Adv30Params.ini** file for the property **DocumentSignatureTemplateLocation**.
4. Specify the database connection properties for the schema.
    - DB_TYPE = ORA <Type of database>
    - JDBC_URL = jdbc:oracle:thin:@162.70.193.24:1521/BASE11G <JDBC connection URL>
    - USER = ADV_FIN310 <Database User>
    - PASSWORD = ADVANTAGE <Database User Password>

    Note: JDBC_URL example for other supported databases like MS SQL is given in the AdvDocToPDFUtilParams.ini file.
5. Open the **_run_DocumentLayoutGenerator.bat** file and update the relevant ADV_HOME and JAVA_HOME paths.
6. Once all the properties are set, execute **_run_DocumentLayoutGenerator.bat**, it should create .json files in the specified output folder.

7. The System Administrator should make sure whenever there is a change in any of the transaction layouts that this utility is executed.

Interpreting Database-related Error Messages

Data Base Management Systems (DBMS) produce errors that can be confusing to the typical Advantage User. Two common examples of these messages are typically the result of a duplicate record on an insert, or a deadlock condition within the database. The format of the message varies depending upon the DBMS that is being used (that is, Oracle or SQL Server). Advantage intercepts these messages and substitutes them with a more meaningful message, as indicated on the Messages table. The following table provides the DBMS Message and a cross reference to the Advantage error message that is displayed to the user.

DBMS Message	Error Code	SQL State	Error Message (MESG)
ORA-00001: unique constraint (Table name) violated SQL Server: Violation of %ls constraint '%.*ls'. Cannot insert duplicate key in object '%.*ls'.	Oracle: 00001 SQL Server: 2627	Oracle: 23000	Q0134 - A unique index was violated. Either a duplicate record was inserted, or a unique attribute was modified: %table name% %SQLCODE% %SQLSTATE%.
ORA-00060: deadlock detected while waiting for resource SQL Server: Transaction (Process ID %d) was deadlocked on {%Z} resources with another process and has been chosen as the deadlock victim. Rerun the transaction.	Oracle: 60 SQL Server: 1205	Oracle: 61000	Q0135 - Two or more transactions deadlocked with each other while waiting to process. Please resubmit the transaction: %SQLCODE% %SQLSTATE%.

Feature Flag Configuration

The Feature Configuration (feature.conf) file includes the following parameters with an enabled property that turns the associated feature on or off in the application. In order to use the associated feature, the enabled property must be set to true and the application must be bounced.

Parameter	Default Value	Parameter Description
Redis	false	When the redis flag is turned on (enabled = true), and redis is setup, then user configuration data (recent search / grid configuration) will read from memory instead of the database. If enabled = false, it will read from the database.
recentSearch maxRecentSearchCardCount maxRecentSearchPinnedCardCount	true 15 10	This parameter allow a site to enable/disable the Recent Search section by setting the enabled property to true or false, respectively. It also provides the ability for a site to configure the maximum recent search card and pinned card count via the maxRecentSearchCardCount and maxRecentSearchPinnedCardCount properties.
userConfiguration	true	This parameter allow sites to enable/disable the Configure option on Search Cards and Column Grids in the Advantage application, by setting the enabled property to true or false, respectively.
favorite	false	This parameter allows a site to enable/disable the Favorites functionality in the Advantage application, by setting the enabled property to true or false, respectively.
alert	false	This parameter allows sites to enable/disable Alert functionality and also to hide/show the Alert icon in the Global Navigation panel, by setting the enabled property to true or false, respectively.
bookmark	false	This parameter allows sites to enable/disable the bookmark functionality and also to hide/show the bookmark icon in the Global

		Navigation panel, by setting the enabled property to true or false, respectively.
transactionSummaryTab	true	This parameter allows sites to enable/disable displaying the Summary tab as the first tab, when a transaction is opened in Pending or Final Phase, by setting the enabled property to true or false, respectively. Note: For all other Phases, the Header is displayed first.
hideSecuredActions	false	When the enabled property is set to true, secured actions that the user does not have access to are hidden on the indicated resource. When the enabled property is set to false, secured actions are visible on the indicated resource.
publish publishHomePageEnable	true false	This parameter allow sites to enable/disable the publish feature of role based UI configuration. When the enabled property is set to true, the Publish button on the PUBLISH page is visible, when the enabled property is set to false, the Publish button on the PUBLISH page is hidden. The publishHomePageEnable property allow sites to enable/disable the publish feature for home page configuration, by setting the property to true or false, respectively.
agGridEnable	false	This parameter allows sites to activate the Advanced Grid functionality on standard grids, by setting the enabled property to true.
gridColumnReposition	false	This parameter allow sites to enforce permanent column repositioning on grids using drag and drop. When the enabled property = false, column reordering is non-persistent; therefore, column reordering is not saved when a user navigates away from the page or performs pagination. When the enabled property = true column reordering is persistent; and the column position changes are saved for a user.
tableBasedConstraints	false	Enables the Configurable Validation feature when the enabled property for this flag is set to true, that is, the system recognizes and

flushCacheOnConfigRuleUpdates	false	consumes table driven validations setup in the application. The flushCacheOnConfigRuleUpdates property enables the real time testing feature when set to true, that is, changes to Configurable Validation setup (additions/updates) are immediately reflected in the application. This property assists with the development of configurable validations in test environments without a need for an application bounce. This property should be set to *true* in conjunction with the enabled property for tableBasedConstraints set to *true*. The flushCacheOnConfigRuleUpdates property must be set to *false* in Production environments.
configurableFormula flushCacheOnConfigFormulaUpdates	false false	Enables the real time testing feature when set to *true*, that is, changes to Configurable Formula setup (additions/updates) are immediately reflected in the application. The flushCacheOnConfigFormulaUpdates property assists with the development of configurable formulas in test environments without a need for an application bounce. This property should be set to *true* in conjunction with the enabled property for configurableFormula set to *true.* The flushCacheOnConfigFormulaUpdates property must be set to *false* in Production environments.
enableDateRangeSearch	false	This parameters enables searching with operators (<, >, :,) in searchable date fields in the Advantage application, when the enabled property is true.
userSpecificDualFactorAuthentication	false	If this parameter is disabled (enabled is set to false), then the user will not see the DFA User flag on the following pages: Dual Factor Authentication User Info (DFAINFO), UDOC transaction, or User Information (SCUSER) page. This feature flag is created only for DFA functionality testing purpose. This flag helps to keep the feature enabled in ADV30Params.ini for the application but not forcing all users to

		perform dual factor authentication when accessing the application.
transactionWorkflowActions	false	If this parameter is enabled, then a user that belongs to a workflow role is not required to perform the Take Task action prior to performing other workflow actions.
markPendingExtensionsForSynchronization	false	If this parameter is enabled, then a record must be added to the migration queue before it is selected by the Designer to XML Synchronization Process. Refer to the "Add to / Remove from Migration Queue" topic for more information.
enableGridPinning gridPinningPageKeys	false	This parameter enables/disables pinning the grid header for certain pages in the Performance Budgeting application (when a user scrolls vertically). The gridPinningPageKeys property controls which pages are impacted by this parameter. The following keys are delivered by default. keys = ["BF0201","BF0200","BF0400","BF0186", "BF0086","BF0067","BF0050"]} If a page is listed as one of the keys for the gridPinningPageKeys property and the enabled property is set to true for the enableGridPinning parameter, then when a user scrolls vertically on the grid on the indicated page, the column headers remain in view.
enableGlobalGridPinning	false	This parameter enables/disables pinning the grid header (for example, the grid actions, column headers and pagination) across the Advantage application for pages and modals. Note: This feature does not work on the following types of pages: • Widgets available on home pages. • Time In / Time Out Modal (TITO) on the Timesheet (TIMEI) transaction. • Popovers, except it does work on switcher popover on transactions.

		• If the Column Freeze action has been applied to a grid, then the grid pinning will not work. • Grids that do not include any pagination or any toolbar.
viewAllRecordsOnGrid	false	The viewAllRecordsOnGrid feature flag allow sites to enable/disable the ability to show more than 100 records in the standard grid, by setting the enabled property to true or false, respectively. The showViewAllonGrid property must also be set to true in the page metadata for the grid and there must be more than 100 records returned in the search results. If the showViewAllonGrid property is not present in the metadata for the grid and/or the viewAllRecordsOnGrid feature flag is set to false, then grid pagination does not include the ability to show more than 100 records at one time. Performance implications should be considered when enabling the showViewAllonGrid property in the page metadata for a grid. Note: The view_all_max_row_count property in the ui_preferences.conf file determines the maximum number of records that can be displayed in the grid. Refer to the “UI Preferences Configuration” topic for more information about this property.
columnTitleWrap	false	This parameter enables/disables the wrapping of column headings for dynamic columns in grids on Budget Request and Query Results pages in Performance Budgeting. The column_title_wrap_count property defined in the ui_preferences.conf file, determines the number of characters after which the next space should then be wrapped.
enableFirstColumnWrap	false	This parameter enables/disables the ability to prevent the label for the first column heading in a grid from wrapping. If this flag is turned off (enabled = false), the first column will not wrap, and if turned on (enabled = false), the first column will wrap (provided the column name is present in the page data.)

UI Preferences Configuration

The UI Preferences Configuration (ui_preferences.conf) file includes the ui_preferences_config parameter with an enabled property that turns the associated functionality on or off in the application. In order to use the associated functionality, the enabled property must be set to true and the application must be bounced. The ui_preferences_config parameter also includes the following properties.

Parameter	Default Value	Parameter Description
view_all_max_row_count	500	This property indicates the maximum numbers of records that can be displayed in a standard grid. The default value for this property is 500. The grid pagination displays a numbered link based on the value of this property and the link is enabled if there are more than 100 records returned from the search results. For example, if the value is 500, then a 500 link is added in the grid pagination and up to 500 records are returned when the 500 link is selected. The value of this property should be greater than 100 and must be numeric. Performance implications should be considered when setting the value for this parameter. Note: If the showViewAllonGrid property is not set to true in the metadata for the grid and/or the viewAllRecordsOnGrid feature flag is set to false, then grid pagination does not include the ability to show more than 100 records at one time. Refer to the "Feature Flag Configuration" topic for more information about the viewAllRecordsOnGrid feature flag.
spinner_lag_time	3000	This property defines the spinner lag time in milliseconds. This property defines how long the UI should wait for response before loading the spinner. The default value for this property is 3000. With the default set to 3000, the UI will wait for 3 seconds, if it receives a response within 3 seconds, then the page will be refreshed without the spinner; otherwise, the spinner would be loaded for the remainder of the time.
spinner_max_time	300000	This property defines the spinner max duration time in milliseconds. This property defines on how long the UI should for a response before closing the spinner. The default value for this property is 300000. With the default set to 300000, the UI will load the spinner and wait for the server response, if the server does not respond back in 5 minutes (300000 ms), the spinner will be closed.

map_service_provider	google_map	This property defines the map service provider (for example, mapquest/google_map). The default value for this property is google_map.
column_title_wrap_count	10	This property define the dynamic Column Wrapping Count for the column title. The default value for this property is 10. It provides the number of characters in a title's column that after the next space, should be wrapped to the next line.