AX Exception Administrator Guide

Contents

AX Exception Administrator Guide

Contents

1. Configuring the AX Exception database

You can configure AX Exception to use either Microsoft SQL Server or Oracle to store application data. The supported configurations are either AX Server with PostgreSQL 9.1 as the database platform and SQL Server as the AX Exception database platform, or Oracle as both the AX Server and AX Exception database platform, as described in Table 1.

Table 1. Supported AX Exception databases

If your AX Server database is	Your AX Exceptiondatabase must be
• PostgreSQL 9.1	• SQL Server 2008 Standard or Enterprise Edition • SQL Server 2012 Standard or Enterprise Edition
• Oracle 11gR2 • Oracle 11gR1 • Oracle 10gR2	• Oracle 11gR2 • Oracle 11gR1 • Oracle 10gR2

Section contents

1.1. Configuring Oracle for AX Exception

The topics in this section outline the configuration steps that need to be completed on the Oracle database server if Oracle will be used as the database platform for AX Exception. This configuration must be completed by an Oracle database administrator with the necessary permissions for the Oracle database server. This configuration must be completed prior to installing AX Exception.

The following tasks must be completed before you install AX Exception:

1. Determine whether you will use an existing Oracle instance, or if you will need to set up a new Oracle instance for AX Exception. The AX Exception and ACL Analytics Exchange databases can run under the same instance, but they must be created with different database users.
2. Determine whether encryption between the AX Exception application server and the Oracle database server is required. If encryption is required, a certificate from a Certificate Authority needs to be installed and configured in the Oracle Wallet Manager.
3. Configure the Oracle listener. A listener configured for non-SSL communications is required to complete the installation, even if you plan to use SSL for communications between the AX Exception server and the database server. The listener must also have a port configured for SSL communications if encryption is required.
4. Configure a Net Service Name to enable connections to the AX Exception database service from the AX Exception server.
5. Run the SQL script provided to create the tablespace for AX Exception.
6. Run the SQL script provided to create the Oracle database user for AX Exception.
7. Optional. You can run the additional SQL scripts provided to create the AX Exception database schema manually if you prefer, rather than using the AX Exception setup wizard to create the database schema.

1.1.1. Running Oracle database scripts

Before you run the AX Exception setup wizard to install the application files, you need to run the supplied SQL scripts to create the required tablespaces and create the AX Exception database user. You can also, optionally, run the set of SQL scripts that create the database schema manually if you do not want to complete this step using the setup wizard.

Important

You must also ensure that the necessary configuration for AX Exception has been completed in the ACL Analytics Exchange database. The security roles required by AX Exception must be created in the ACL Analytics Exchange database by running the AX Exception roles SQL script for the language you have installed (`EMRoles_<language>.sql`).

To run the Oracle database scripts:

1. Download the AX Exception installer from the ACL Launchpad and extract the setup files.
2. Open Windows Explorer and navigate to the `DBScripts\Oracle` subfolder where you extracted the setup files.
3. Copy the SQL script files to the Oracle server, or a location you can run the scripts from.
4. Open `Create_Tablespaces.sql` and update the variables defined at the top of the script to match your environment. The path defined in the *F_Ex_root* variable must exist before you run the script. Save and close the files after you have made the necessary changes.
5. Open `Create_DB_User.sql` and update the variables defined at the top of the script to match your environment. Save and close the file after you have made the necessary changes.
6. Run `Create_Tablespaces.sql` and then run `Create_DB_User.sql` using an account running under the SYSDBA role.
7. If you want to create the database schema manually, run `SimpleCreateSchema.sql` or run the remaining scripts in the following order using the AX Exception database user account:
   - `Create_Tables.sql`
   - `Create_Views.sql`
   - `Create_Triggers.sql`
   - `Create_Packages.sql`

   If you run the AX Exception schema scripts manually, and someone else will run the AX Exception setup wizard, you must instruct them to choose the **Configure connection settings only** option in the **AX Exception Database** page.

1.1.2. Modifying Oracle connection settings

If your database configuration changes, you need to update the AX Exception configuration files with the new settings. The connection settings for the AX Exception database are stored in three configuration files:

- The AX Exception database username and password are stored in the encrypted `axExceptionConnect.xml` file.
- The database connection settings for the AX Exception database are stored in the `axException.xml` configuration file on the server where AX Exception is installed.
- The setting for the database connection protocol to use is stored in the `wrapper.conf` configuration file.

For information about modifying connection settings from AX Exception to the ACL Analytics Exchange database, see the *ACL Analytics Exchange Server Administrator Guide*.

To modify your connection settings:

1. To modify the username or password for the database, you need to update the values in an unencrypted copy of `axExceptionConnect.xml`. For details about this process, see Modifying axExceptionConnect.xml.
2. To modify database connection settings, such as the IP address of the database server, complete the following steps:

a. Open `axException.xml` in a text editor and update the exceptionmgmt.jdbc configuration settings with the new values you want to use to connect to your database. For details about the exceptionmgmt.jdbc settings you can change, see axException.xml.

b. Save and close the file.

c. Restart the ACL Analytics Exchange Service.

3. To modify the database connection protocol setting, whether SSL is used or not, complete the following steps:

a. Open `wrapper.conf` in a text editor and enter "tcp" for non-SSL connections or "tcps" for SSL connections, as appropriate. The value entered must be lowercase. If "tcps" is set for Oracle, the required configuration must be completed on the database server to allow SSL connections to the Oracle database instance.

b. Save and close the file.

c. Restart the ACL Analytics Exchange Service.

1.2. Configuring SQL Server for AX Exception

The AX Exception setup wizard provides two options for creating the AX Exception database:

1. Set up the database manually before running the AX Exception setup wizard by running the set of SQL scripts provided with the installation package. For more information about this option, see Running SQL Server database scripts manually.
2. Set up the database automatically by entering the required Microsoft SQL Server settings in the setup wizard to create the database as part of the installation process. This option requires the user installing AX Exception to have the required permissions for the SQL Server instance to create the database. Membership in the "sysadmin" role is required to create the database. The built-in "sa" database user and Windows users that belong to the Administrators group on the server are assigned this role by default. For information about this option, see the AX Exception InstallationGuide.

For information about the supported versions of Microsoft SQL Server and other requirements, see AX Exception server requirements.

1.2.1. Running SQL Server database scripts manually

Important

You must also ensure that the necessary configuration for AX Exception has been completed in the AX Server database. The security roles required by AX Exception must be created in the AX Server database by running the AX Exception roles SQL script for the language you have installed (`EMRoles_<language>.sql`).

To run the SQL Server database scripts manually:

1. Start **SQL Server Management Studio** (**Start** > **All Programs** > **Microsoft SQL Server *<version>*** > **SQL Server Management Studio**).
2. If you are prompted to log in, enter the required information to connect to the AX Exception database in the **Connect to Server** dialog box and click **Connect**.
3. Open and run the `CREATE_EM_DBASE.sql` script with a database or Windows user account that belongs to the "sysadmin" role. For example, you can use the built-in "sa" account.

   The database is created with the name EM_DBASE by default. If you want to specify a different database name, use the Quick Replace command to replace all occurrences of EM_DBASE with the name you want to use. The script also creates a user account named EM_USER1 and assigns it the "dbowner" role for the AX Exception database. You can use the Quick Replace command to modify the username, and edit the CREATE LOGIN command in the script to change the password assigned to the account.
4. Open and run the scripts in the following order using the user account that has been assigned the "dbowner" role for the AX Exception database:
   - `CREATE_TABLES.sql`
   - `CREATE_VIEWS.sql`
   - `CREATE_TRIGGERS.sql`
   - `CREATE_UDT.sql`
   - `CREATE_FUNCTIONS.sql`
   - `CREATE_STORED_PROCEDURES.sql`
5. Exit **SQL Server Management Studio**.

1.2.2. Modifying SQL Server connection settings

If your database configuration changes, you need to update the AX Exception configuration files with the new settings. The settings for the AX Exception database are stored in two configuration files:

- The AX Exception database username and password are stored in the encrypted `axExceptionConnect.xml` file.
- The database connection settings for the AX Exception database are stored in the `axException.xml` configuration file on the server where AX Exception is installed.

For information about modifying connection settings from AX Exception to the ACL Analytics Exchange database, see the *ACL Analytics Exchange Server Administrator Guide*.

To modify your connection settings:

1. If you want to update the username or password for the database, you need to update the values in an unencrypted copy of `axExceptionConnect.xml`. For details about the process, see Modifying axExceptionConnect.xml.
2. If you want to modify database connection settings, such as the IP address of the database server, complete the following steps:

a. Open `axException.xml` and update the exceptionmgmt.jdbc configuration settings with the new values you want to use to connect to your database. For details about the exceptionmgmt.jdbc settings you can change, see axException.xml.

b. Save and close the file.

c. Restart the ACL Analytics Exchange Service.

2. Administering the AX Exception service

The topics in this section cover the information you need to know to manage the ACL Analytics Exchange Service that runs the AX Exception application. If you installed AX Exception and AX Server on the same server, both applications run using the same service. For information specific to AX Server, see the *ACL Analytics Exchange Server Administrator Guide*.

Section contents

2.1. Starting the service

The ACL Analytics Exchange Service used by AX Exception should start automatically when Windows starts, and remain running at all times. If the service fails to start for some reason, such as a configuration file error, or if you previously stopped the service, you can start it manually.

To start the ACL Analytics Exchange Service manually:

1. Open the **Services** window:
   - On Windows 2008, select **Start** > **Control Panel** > **Administrative Tools** > **Services**.
   - On Windows 2012, navigate to the **Start** page, select **Administrative Tools**, and then double-click **Services**.

2. Right-click the ACL Analytics Exchange Service in the list of services and select **Start**.

2.2. Stopping the service

The ACL Analytics Exchange Service used by AX Exception should start automatically when Windows starts, and remain running at all times. In some cases, you may need to stop and restart the service before configuration changes take effect, or you can stop the service if you want to prevent users from accessing AX Exception for a period of time.

To stop the ACL Analytics Exchange Service manually:

1. Open the **Services** window:
    - On Windows 2008, select **Start** > **Control Panel** > **Administrative Tools** > **Services**.
    - On Windows 2012, navigate to the **Start** page, select **Administrative Tools**, and then double-click **Services**.
2. Right-click the ACL Analytics Exchange Service in the list of services and select **Stop**.

2.3. Viewing the service logs

You can view the log information recorded for the ACL Analytics Exchange Service any time you need to troubleshoot errors, or when you want to check the status of the application. Table 1 lists the service logs and the information they contain. The `<ex_dir>` value in the "Default location" column is the location where AX Exception is installed. The default location for `<ex_dir>` is `C:\ACL\App`.

Table 1. AX Exception Logs

Log file name	Default location	Description
ActivationDLL.log	windows\system32	Records information about communications with the ACL Activation Server.
axException.log	<ex_dir>\geronimo\var\log	Records errors and information specific to AX Exception. You can configure the settings for this log file in the axException-log4j.properties file in the geronimo\var\config folder.
axException-functional.log	<ex_dir>\geronimo\var\log	Records functional errors encountered by AX Exception.
axException-technical.log	<ex_dir>\geronimo\var\log	Records technical errors encountered by AX Exception.

Log file name	Default location	Description
axExceptionAdmin.log	<ex_dir>\geronimo\var\log	Records errors and information specific to AX Exception Administration. You can configure the settings for this log file in the axExceptionAdmin-log4j.properties file in the geronimo\var\config folder.
axExceptionAdmin-functional.log	<ex_dir>\geronimo\var\log	Records functional errors encountered by AX Exception Administration.
axExceptionAdmin-technical.log	<ex_dir>\geronimo\var\log	Records technical errors encountered by AX Exception Administration.
geronimo.log	<ex_dir>\geronimo\var\log	Records errors and information from the Tomcat application server, which hosts the AX Exception application. You can configure settings for this log file in the server-log4j.properties file in the geronimo\var\log folder.
geronimo_service.log	<ex_dir>\geronimo\var\log	Records errors and information from the ACL Analytics Exchange Service. You can configure the settings for this log file in the wrapper.conf file in the geronimo\var\config folder.
deployer.log	<ex_dir>\geronimo\var\log	Records errors and information about applications that are deployed to the Tomcat application server. You can configure settings for this log file in the deployer-log4j.properties file in the geronimo\var\log folder.

2.4. Modifying log4j2 configuration file settings

The ACL Analytics Exchange applications that are hosted by the Tomcat application server use a standard logging mechanism called Apache log4j2. The configuration settings for each of these application logs can be modified individually. The most common reason to modify the configuration settings is to increase the amount of information written to the log to assist with troubleshooting application errors. The level of logging information can be modified by adjusting the values of the **log4j.appender.*<logfile_value>*.Threshold** properties in the file. You can also adjust other values such as the maximum size of the log file, and the location where the log file is created. The following table lists the ACL Analytics Exchange log4j2 property files that you can configure.

Log configuration file	Description
axAdmin-log4j2.xml	Specifies the log settings for the AX Server Configuration web application. This configuration file is only found on the AX Server. The default log file name is axAdmin.log. The default maximum size of the log file is 5MB (5120KB) and the 10 most recent log files will be archived. The default threshold is INFO.

Log configuration file	Description
axCore-log4j2.xml	Specifies the log settings for AX Server and AX Engine Node. The default log file name is axCore.log. The default maximum size of the log file is 5MB (5120KB) and the 10 most recent log files will be archived. The default threshold is INFO.
axException-log4j2.xml	Specifies the log settings for the AX Exception web application. This configuration file is only found on the server where AX Exception is installed.
axExceptionAdmin-log4j2.xml	Specifies the log settings for the AX Exception Administration web application. This configuration file is only found on the server where AX Exception is installed.
axGateway-log4j2.xml	Specifies the log settings for ACL Add-In. This configuration file is only found on the AX Server. The default log file name is axGateway.log. The default maximum size of the log file is 5MB (5120KB) and the 10 most recent log files will be archived. The default threshold is INFO.
cas-log4j2.xml	Specifies the log settings for AX Server authentication. This configuration file is only found on the AX Server. The default log file name is cas.log. The default maximum size of the log file is 10MB and the 10 most recent log files will be archived. The default threshold is INFO.
axWebclient-log4j2.xml	Specifies the log settings for AX Web Client. This configuration file is only found on the AX Server.

To modify log4j2 configuration properties:

1. Open Windows Explorer and navigate to the `TomCat/logs` subfolder.
2. Open the property file for the log you want to modify the configuration for in a text editor.
3. Make any necessary modifications, and then save and close the file.
4. Stop and restart the ACL Analytics Exchange Service.

2.5. Ports used by ACL Analytics Exchange Service

The ACL Analytics Exchange Service is installed with the default communications port settings used by the application server. Table 1 lists the default ports used by specific Tomcat application server components.

Table 1. ACL Analytics Exchange Service default ports

Port Number	Component Name	Description
8009	Tomcat Connector AJP	Port used to connect to the Tomcat web server.
80	Tomcat Connector HTTP	Port used for unencrypted HTTP communication with the server.
443 or 8443	Tomcat Connector HTTPS	Port used for encrypted HTTP (HTTPS) communication with the server. If you are performing a new installation of AX Server, the default port is 443. If you are upgrading an earlier version of AX Server, the default port is 8443.

2.6. Modifying ports used by AX Exception

If you encounter a situation where ports used by the Tomcat application server conflict with other applications or services, you can modify the ports used. In AX Exception version 5, the default HTTPS port value is 443. In earlier versions, the default value was 8443. On systems that have been upgraded to version 5 from an earlier version, port 8443 is automatically retained. Users on systems that use port 8443 must specify the port number when typing in AX Exception URLs. For example, https://exception.acl.com:8443/exceptionmgmt. Users on systems that use port 443 do not have to specify the port number when typing in AX Exception URLs. Regardless of which HTTPS port is in use, users still need to specify "https" when entering the URL to securely access AX Exception web applications.

To modify a port used by AX Exception:

1. Stop the ACL Analytics Exchange Service on the AX Exception server.
2. In Windows Explorer, open the `geronimo\var\config` subfolder in the directory where you installed AX Exception.
3. Open `config-substitutions.properties` in a text editor.
4. Edit the value of the port setting you want to change, and save and close the file.
5. If you are modifying the HTTPS port, you need to complete the following additional steps:
    a. Open `aclAuditExchange.xml` and update the *AXHttpsPort* property setting with the new value, and save and close the file.
    b. Open `aclCasClient.xml` and update the *cas.securityContext.casServerPort* property setting with the new value, and save and close the file.
    c. Open `axExceptionAdmin.xml` and update the *useradministration.exceptionmgmt.url* property setting with the new value, and save and close the file.
    d. Open `axException.xml` and update the *exceptionmgmt.url* property setting with the new value, and save and close the file.
6. Modify the **Data upload URL** value in the AX Server Configuration web application, so that the correct port is specified. Alternatively, you can modify this setting by updating the *EmDataLoadUrlRoot* property in the `aclAuditExchange.xml` file on the AX Server.
7. Restart the ACL Analytics Exchange Service on the AX Exception server.

2.7. Editing the service configuration settings

The settings for the ACL Analytics Exchange Service are stored in the `wrapper.conf` file in the `\geronimo\var\config` subfolder where AX Exception is installed. Some of the settings are configured with the values required to start the service and should not be changed, but there are a number of global settings that can be modified to change the behavior of the application. If AX Exception is installed on the same server as AX Server, any changes made in the `wrapper.conf` file will affect both applications.

To edit wrapper.conf configuration settings:

1. Open Windows Explorer and locate the `wrapper.conf` file in the `\geronimo\var\config` subfolder where AX Exception is installed.
2. Open the file in a text editor and modify the configuration settings as necessary.

   The `wrapper.conf` file stores the following types of settings:

   - Java configuration settings including memory settings, file locations, and temporary directory settings.
   - AX Exception database type (Oracle or SQL Server) and protocol (SSL or non-SSL).
   - Security certificate configuration settings and keystore location.
   - Logging settings for the ACL Analytics Exchange Service.

   For a complete list of configuration settings you can modify, see system.properties.
3. Save and close the `wrapper.conf` file.
4. Restart the ACL Analytics Exchange Service.

2.8. Understanding service account configuration

AX Exception is installed with a Windows service that performs most of the application functions on the server. The service is configured during installation with the Windows user account you select to run the service.

The ACL Analytics Exchange Service is installed on the AX Exception server, and the AX Exception and AX Exception Administration web applications are installed as components in the Tomcat application server. You can use either a local user account on the server or a domain account to run the Windows service. The setup wizard assigns the local permissions on the server required to run the service. Using the built-in Local System account to run the service is not supported. If AX Server and AX Exception are installed on the same server, the Windows account used to run the ACL Analytics Exchange Service is selected during the AX Server installation and the same account is used for AX Server and AX Exception.

2.9. Understanding security certificate configuration

When you install AX Exception, you are required to configure a security certificate. By default, a self-signed security certificate is installed. In some cases, you may want to replace the self-signed security certificate with a certificate issued by a third-party certificate authority (CA).

The certificate configured on the AX Exception server allows clients connecting to the AX Exception and AX Exception Administration web applications with their web browsers to

securely access the applications through HTTPS. Connecting to the web applications using HTTPS ensures that data transmitted between the web browser and the server is encrypted.

If you choose to use the self-signed certificate, each user that accesses the AX Exception web server will encounter a warning page indicating that the security certificate was not issued by a trusted certificate authority. They will need to install the self-signed certificate in their browser to stop this warning page from being displayed. Certificate installation is not typically required if you replace the self-signed certificate with a certificate purchased from a CA because Internet Explorer supports certificates issued by most CAs automatically.

2.10. Configuring Certificate Authority signed security certificates

The topics in this section provide information about configuring your AX Exception server to use a security certificate from a Certificate Authority (CA) to secure HTTPS connections to the server. During installation, a self-signed certificate is created for use during testing and for organizations that do not want to purchase a certificate from a CA. If possible, this self-signed certificate should be replaced with a certificate signed by a CA.

You need to complete the following six steps to configure a security certificate from a CA:

1. Back up your existing Geronimo configuration file.
2. Create a new keystore.
3. Create a certificate signing request (CSR) and send the request to your CA.
4. Import the certificate, and any required intermediate or root certificates, returned from your CA into your keystore.
5. Configure Geronimo to use your signed certificate for all secure HTTP requests.
6. If necessary, install the certificate in the web browser on each computer that will access ACL Analytics Exchange web applications. This is not necessary if the certificate is provided by a CA listed in the Trusted Root Certification Authorities list in Internet Explorer. Large commercial CAs, such as VeriSign, are included in this list.

2.10.1. Backing up your Tomcat application server configuration

Before you change the security configuration of the Tomcat application server to use a Certificate Authority signed certificate, you should back up configuration files so you have a copy to restore from if you run into any unexpected problems.

To back up the configuration files:

1. In Windows Explorer, open the `TomCat\conf` subfolder in the directory where you installed the ACL Analytics Exchange server application you are updating the keystore configuration for.
2. Copy the `conf\tomee.xml` and `conf\server.xml` files to a safe backup location. If you run into any issues while you are configuring the security certificate, you can restore your original configuration by stopping the ACL Analytics Exchange Service, restoring these files, and then restarting the service.

2.10.2. Creating a keystore and importing your certificate

A keystore file is a special type of file that stores information about security certificates. The following procedure outlines the process for creating a Java Key Store (JKS) with a certificate signed by a Certificate Authority. If you purchased your security certificate from a commercial CA, such as VeriSign, consult the documentation they provide for information on configuring your keystore. If your certificate is in a format that cannot be imported into a keystore (e.g. PKCS12), and you cannot convert it to the PEM format, contact ACL Support Services for assistance with configuring the certificate in Tomcat.

In order to use the keytool command without specifying the full path each time you use it, you need to add the Java `bin` subdirectory your path. You can do this permanently by updating the *Path* Environment Variable to include the full path to the Java `bin` subdirectory, or add it temporarily on the command line using the following syntax:

```
Set PATH=<java_bin_path>;%PATH%
```

For example:

```
Set PATH=C:\Program Files(x86)\Java\jdk1.7.0_67\bin\;%PATH%
```

To configure a new keystore with an SSL certificate:

1. Open a command prompt on the server.
2. Use the following syntax to create the new keystore with a self-signed certificate:

   `keytool -genkey -alias <alias> -keyalg RSA -keystore <keystore_filename>`
3. Create a certificate signing request using the following syntax, and then send the certificate request to the Certificate Authority you are using.

   `keytool -certreq -alias <alias> -keyalg RSA -file <csr_output_file> -keystore <keystore_filename>`
4. Depending on the CA you are using you may need to import an intermediate certificate and/or root certificate into your keystore. Use the following syntax to import one or both of these certificates:

   `keytool -import -alias <alias> -keystore <keystore_filename> -trustcacerts -file <certificate_filename>`

   If you are importing both certificates the *alias* specified for each certificate should be unique. You need to first import the root certificate, and then run the `keytool` command again to import the intermediate certificate.

5. Use the following syntax to import your security certificate:

   `keytool -import -alias <alias> -keystore <keystore_filename> -trustcacerts -file <certificate_filename>`

   The *alias* specified must be the same value specified in step 2 when you generated the keystore. The imported certificate will replace the default self-signed certificate created in the keystore.

6. Copy the keystore file to the `App\keystores` subfolder.

2.10.3. Configuring the Tomcat application server to use your certificate

You need to configure the Tomcat application server so that all HTTPS communications are secured using the security certificate you created.

To configure the Tomcat application server to use your new certificate:

1. Locate server.xml in the `TomCat\conf` subfolder and open it in a text editor.
2. Update the following settings:
   - **keystoreFile** – Enter the name and path to the keystore file you created. You need to replace `myKeystore` with the name of the keystore you created. The value in the text box should be in the following format: `C:\ACL\App\keystores\<your_keystore_name>`
   - **keystorePass** – Enter the password you specified for the keystore when you created it. The password must be enclosed in double quotation marks (" ").
3. Save and close `server.xml`.
4. Restart the ACL Analytics Exchange Service.

Related reference

system.properties

2.11. Configuring Internet Explorer to use your certificate

To access the AX Exception and AX Exception Administration web applications without encountering certificate errors, you may need to install the security certificate in Internet Explorer.

This step needs to be completed on each computer users will access the web applications from if you are using a self-signed certificate, or if your certificate is provided by a CA that is not listed in Internet Explorer's Trusted Root Certification Authorities list. This configuration must be completed by a user with Administrator rights for the computer. You can check if your

issuing CA is in the list by opening Internet Explorer, selecting **Tools** > **Internet Options**, and then clicking the **Certificates** button in the **Content** tab.

To install your certificate in Internet Explorer:

1. Open Internet Explorer on the server and navigate to the Geronimo Console on the AX Exception server. The default location is https://<server_name>/console. If you are working on an AX Exception that has been upgraded from a version earlier than 5.0, you may need to specify port 8443 in the Geronimo Console URL. For example, https://<server_name>:8443/console.
2. Click **Continue to this website (not recommended)**.
3. Click the **Certificate Error** button next to the browser address bar.
4. Click the **View Certificates** link.
5. In the **Certificate** dialog box, click **Install Certificate**.
6. Complete the following steps in the **Certificate Import Wizard**:
    a. Click **Next**.
    b. Select **Place all certificates in the following store**, and click **Browse**.
    c. In **Select Certificate Store**, select **Trusted Root Certification Authorities** and click **OK**.
    d. Click **Next**.
    e. Click **Finish** to import the certificate.
    f. In the **Security Warning** dialog box, click **Yes** to install the certificate.
    g. Click **OK** in the confirmation dialog box to complete the wizard.
7. Click **OK** to close the **Certificate** dialog box.
8. Restart Internet Explorer and navigate to the Geronimo console page.

   If the certificate is configured correctly, the console page will load without displaying the certificate error page. You will also be able to access the other AX Exception web applications from the computer without encountering the certificate error page.

3. Web application administration

The topics in this section cover the information you need to know to administer the AX Exception and AX Exception Administration web applications.

Section contents

3.1. Managing the AX Exception web application

You can access the web-based Geronimo Console by entering the following URL in your web browser: http://<*server_name*>/console where <*server_name*> is the name or IP address of the server where AX Exception is installed.

Note

If your AX Exception server has been upgraded from a version earlier than 5.0, you may need to specify port 8443 in the address. For example, `https://axexception.acl.com:8443/console`.

Starting the web application

To start the web application:

1. Open the Geronimo Console in your web browser and log in.
2. Click **Applications** > **Web App WARs** in the **Console Navigation** list.
3. Complete the following step to start the appropriate application:
    - To start AX Exception, locate the **com.acl/exceptionmgmt/<version>/car** entry and click **Start** in the **Commands** column.
    - To start AX Exception Administration, locate the **com.acl/ax-em_admin/<version>/car** entry and click **Start** in the **Commands** column.

You can also start the web applications on the command line using the following syntax:

AX Exception:

```
"%GERONIMO_HOME%/bin/deploy.bat" --user <uid> --password <pwd> start exceptionmgmt
```

AX Exception Administration:

```
"%GERONIMO_HOME%/bin/deploy.bat" --user <uid> --password <pwd> start ax-em_admin
```

The required *uid* and *pwd* values are the username and password you use to log in to the Geronimo Console and AX Server Configuration.

Stopping the web application

To stop the web application:

1. Open the Geronimo Console in your web browser and log in.
2. Click **Web App Wars** in the **Console Navigation** list.
3. Complete the following step to stop the appropriate application:
    - To stop AX Exception, locate the **com.acl/exceptionmgmt/<version>/car** entry and click **Stop** in the **Commands** column.
    - To stop AX Exception Administration, locate the **com.acl/ax-em_admin/<version>/car** entry and click **Stop** in the **Commands** column.

You can also stop the web applications on the command line using the following syntax:

AX Exception:

```
"%GERONIMO_HOME%/bin/deploy.bat" --user <uid> --password <pwd> stop exceptionmgmt
```

AX Exception Administration:

```
"%GERONIMO_HOME%/bin/deploy.bat" --user <uid> --password <pwd> stop ax-em_admin
```

The required *uid* and *pwd* values are the username and password you use to log in to the Geronimo Console and AX Server Configuration.

3.2. AX Exception security roles

AX Exception includes a number of user roles that grant access to particular parts of the application. User role assignments are administered using the AX Exception Administration web application. For information on administering AX Exception security, see the AX Exception Administration User Guide.

The following roles are available in the system by default:

- **Primary Reviewer** – Users that belong to this role can work with new exceptions in the system for entities that they have been assigned rights for. New exceptions are automatically assigned to the "Assigned to Primary Reviewer" workflow state. The Primary Reviewer can either assign the exception to the Secondary Reviewer, or they can close the exception.
- **Secondary Reviewer** – Users that belong to this role can work with exceptions in the "In Review" state for entities that they have been assigned rights for.
- **EM Admin** – Users that belong to this role have access to the Purge commands in the **Exception Details** page. They can delete exceptions from the system for the entities they have been assigned rights to using other roles that give them write access to the entities (i.e. not the "Read Only EM" role or any custom roles you add that only provide read-only access). For example, if they have been assigned the "Primary Reviewer" role for five entities, they can purge exceptions from these entities based on their combined "EM Admin" and "Primary Reviewer" rights. They cannot purge exceptions from any other entities in the system.
- **Show All Entities EM** – Users that belong to this role have read-only access to all entities in the system. They can view exceptions for all entities, but cannot edit them or move them to new workflow states. Users that belong to other user groups that give them rights to modify particular entities will still be able to modify those entities, and they will have read-only access to all other entities.
- **Access All Entities EM** – Users that belong to this role can view and edit all entities in the system.
- **Read Only EM** – Users that belong to this role have read-only access to entities they have been assigned rights for. They can view the exceptions for these specific entities, but cannot edit them or move them to new workflow states.

The last four roles are built-in system roles that are available regardless of the workflow being used. The "Primary Reviewer" and "Secondary Reviewer" roles are available as part of the default workflow.

AX Exception can be customized to include additional roles where required. Customized roles are typically added to the system during the implementation phase with assistance from the ACL Professional Services Group. Contact the Professional Services Group or ACL Support Services for more information if you need to add additional roles to the system.

3.3. Understanding the Dataloader process

The Dataloader is an application called by the AX Engine publishing process to transfer analytic results to AX Exception. The Dataloader is installed as a component of AX Server, and as a component of AX Engine Node. If you have one or more instances of AX Engine Node configured, the Dataloader installed with each of these instances is used to publish results to AX Exception for analytics processed on that server.

The Dataloader processing is controlled in an Apache Ant build file called `DataLoader.xml`. This file includes the following targets that perform specific tasks:

- **complete** – This target runs the complete data loading process by calling each of the following targets in turn: metadata, data, parameter, and post. This is the default target that runs if you call `DataLoader.xml` without specifying a target.
- **metadata** – This target checks and loads metadata into the AX Exception database. If new columns are encountered the corresponding metadata columns are created in the database.
- **data** – This target loads data into the AX Exception database.
- **parameter** – This target loads parameter values into the AX Exception database.
- **post** – This target checks whether the metadata, data, and parameter tests ran successfully, and updates the AX Exception database with the status of the tests.

Each time the Dataloader runs, it creates a new folder in the `Dataloader\sessions` folder, on the server where the analytic is processed, with the format `Job<`*`job_number`*`>`. If the job runs successfully, this folder is deleted. If any of the tasks in the `DataLoader.xml` file are not completed the folder is saved and can be used to determine where the publishing process failed. One or both of the following log files may be available to assist you with troubleshooting the publishing process:

- ACLProject.log – This log file provides information about the analytic processed by the AX Engine. It provides detailed information about each *ACL*Script command that was executed in the analytic. The log file is created in the `output\ACL` subfolder for the job.
- dataLoader.log – This log file provides information about errors encountered while running the `DataLoader.xml` file. It indicates why a data upload could not be completed. The log file is created in the root folder for the job.

3.4. Modifying the Dataloader configuration settings

Each time the Dataloader runs, it uses the current settings in the `Dataloader\template` subfolder to select the configuration settings to use to upload exception data from AX Server to the AX Exception database. You can modify the settings in the `dataLoaderCommon.properties` file whenever necessary. You can set the locations of files and security certificates, and specify if debug or test mode should be turned on. You should not modify any of the other files in the `template` subfolder.

To modify the Dataloader configuration settings:

1. Navigate to the folder on AX Server , or AX Engine Node, where the Dataloader configuration file is located. The default location is the `DataLoader\sessions\template\conf` subfolder.
2. Open the `dataLoaderCommon.properties` file in a text editor and make the necessary changes. For information on the properties you can configure, see dataloaderCommon.properties.
3. Save and close the file.

   The next time the Dataloader runs, it will use your updated settings.

3.5. Limits for exception publishing

The information below outlines the limits that exist when publishing analytic results to AX Exception.

Value	Description	Limitation
Table to publish field	In the Publish Results page in the AX Client scheduling wizard there are drop-down lists where you select the result tables to publish to AX Exception. The name of the table is set in the analytic using the RESULT tag.	31 characters
Field name	The name assigned to a column in the results table. This limit is specific to AX Exception.	28 characters
Field label	The label assigned to a column in the results table. This limit is specific to AX Exception.	30 characters
Entity name	In the Publish Results page in the AX Client scheduling wizard there is a text box where you can specify the entity name. This limit is enforced by the AX Client user interface.	30 characters
Entity ID	The entity ID is autogenerated based on the entity name.	30 characters
Analytic name	In the Publish Results page in the AX Client scheduling wizard there is a text box where you can specify the analytics name. This limit is enforced by the AX Client user interface.	26 characters
Analytic ID	The analytic ID is autogenerated based on the analytic name.	26 characters

The following field types are not supported for analytic results published to AX Exception:

- ACCPAC
- BASIC
- COMPUTED

- HALFBYTE
- IBMFLOAT
- PCASCII
- UNSIGNED
- UNISYS
- VAXFLOAT

If the data files you are accessing in your analytic access these data types, you need to convert them to supported data types before you can output them to an analytic results table for AX Exception.

Results tables published to AX Exception have the following additional limits:

- The result table for an analytic cannot exceed 900 columns (a maximum of 300 of which can be numeric, or a maximum of 600 string or date).
- The maximum row size cannot exceed 8060 bytes in SQL Server databases. This is a limit imposed by SQL Server. For more information, see "Maximum Capacity Specifications for SQL Server 2008" (http://msdn.microsoft.com/en-us/library/ms143432(v=sql.100).aspx)
- String columns have a 2000 character limit.
- Numeric columns use the SQL Server float data type if SQL Server is the database platform.

3.6. Modifying axExceptionConnect.xml

The username and password settings for the AX Exception database and the mail server account used to send escalation emails are securely stored in an encrypted file named `axExceptionConnect.xml`. If you need to modify any of these settings, you need to enter the new information in an unencrypted copy of the file, and move it to the `geronimo\var\config` subfolder where it will be encrypted automatically the next time the ACL Analytics Exchange Service starts.

To modify encrypted configuration settings:

1. Stop the ACL Analytics Exchange Service in Windows Services.
2. Navigate to the folder where the unencrypted copy of `axExceptionConnect.xml` is located. By default it is located in the following subfolder where AX Exception is installed: `ACL\installation_backup\configuration\ExceptionMgmt`
3. Copy the unencrypted file to the `geronimo\var\config` subfolder where AX Exception is installed.
4. Open the property file in a text editor, make the necessary changes, and save and close the file.
5. Start the ACL Analytics Exchange Service in Windows Services. The `axExceptionConnect.xml` file is encrypted when the AX Exception component starts.

Related tasks

Stopping the service
Starting the service

3.7. Modifying aclDatabase.xml

The settings for connecting to the ACL Analytics Exchange database are securely stored in an encrypted file named `aclDatabase.xml`. If you need to modify any of these settings, you need to enter the new information in an unencrypted copy of the file, and move it to the `TomCat\conf` subfolder where it will be encrypted automatically the next time the ACL Analytics Exchange Service starts. If you update your database settings, you need to update the `aclDatabase.xml` file on each Tomcat application server that connects to the ACL Analytics Exchange database. This includes AX Server, AX Engine Node, and the AX Exception server if AX Exception is part of your implementation and it is installed on a separate server from AX Server.

To modify the encrypted configuration file:

1. Stop the ACL Analytics Exchange Service in Windows Services.
2. Navigate to the folder where an unencrypted copy of `aclDatabase.xml` is located.
3. Copy the unencrypted file to the `TomCat\conf` subfolder where the Tomcat application server application server you are updating is installed.
4. Open the property file in a text editor, make the necessary changes, and save and close the file.
5. Start the ACL Analytics Exchange Service in Windows Services. The `aclDatabase.xml` file is encrypted when the service starts.

4. Workflow and escalation administration

Workflow and escalation are two closely related components of the AX Exception application. Workflow refers to the set of rules that are applied to exceptions from the time they enter the application until they are resolved. Escalation is the notification mechanism that is used to alert users when exceptions are not processed according to the defined rules.

The topics in this section cover the information you need to know to understand and modify the default AX Exception workflow and escalation settings.

Section contents

Understanding the escalation process
Configuring escalation email templates
Workflow configuration overview
Modifying the workflow XML file

Workflow configuration syntax
Workflow tutorial

4.1. Understanding the escalation process

The AX Exception escalation process assigns unresolved exceptions to users belonging to particular roles. The assignment of exceptions to particular roles is based on the rules defined in the workflow definition file, an XML file that describes the rules for the application. The escalation process is the part of the application that tracks the status of assigned exceptions, and sends email notifications if exceptions are not handled within a predetermined time period. The workflow definition file defines the rules used to determine when email notifications should be sent and which user groups they should be sent to. The escalation process excludes Saturday and Sunday from calculations that involve escalation duration.

The escalation process runs at a frequency defined by the **escalation.job.cron** setting in the `axException.xml` configuration file. By default, the escalation process is configured to run at 4 a.m. every day, but this can be set to another frequency. Each time the escalation process runs it completes the following tasks:

1. It identifies exceptions that need to be processed based on their due date and the workflow step (status) they are currently in.
2. It reads the workflow definition file and determines the valid escalation levels for each workflow step (status).
3. It identifies the current status of each unresolved exception and processes it in one of the following ways:
    - If an exception is past the due date for the current escalation level, it is moved to the next escalation level and assigned a new due date. The new due date is based on the duration set for that step and the escalation level in the workflow definition file.
    - If an exception is already at the highest escalation level for the step, and is past the due date, a new due date is assigned to the exception based on the value of the **escalation.aftermaxlevel.duration** setting in the `axException.xml` configuration file.
    - If an exception is identified that is older than the total duration of all of the higher escalation levels defined for the step, the exception is moved to the highest escalation level and the due date is calculated using the current system time plus the value of the **escalation.aftermaxlevel.duration** setting in the `axException.xml` configuration file.

      For example, consider a step with the following escalation levels:

```
<step id=”1” name=”In Review”>
 <meta name=”escalation_0_usergroup”>Secondary Reviewer</meta>
 <meta name=”escalation_0_duration”>0</meta>
 <meta name=”escalation_1_usergroup”>Secondary Reviewer/meta>
 <meta name=”escalation_1_duration”>5</meta>
 <meta name=”escalation_2_usergroup”>Primary Reviewer</meta>
 <meta name=”escalation_2_duration”>10</meta>
```

If the exception is at *escalation_0* the due date will be recalculated using the logic outlined above if it is more than 15 days old (*escalation_1_duration* + *escalation_2_duration*). However, if the exception is at *escalation_1*, the due date will be recalculated using the logic above if it is more than 10 days old. Only the *escalation_2_duration* value is used because this is the only escalation level that is higher than the escalation level currently assigned to the exception.

4. After the first three steps are completed, the escalation process sends escalation email notifications to user groups that have exceptions assigned to them that are past the due date.

4.2. Configuring escalation email templates

AX Exception uses the Apache Velocity engine to process email notifications. There are three email template files you can edit to customize the content of escalation emails. The email templates are preconfigured to send HTML email messages, but you can edit them to use plain text if you prefer. If you use plain text for your templates, you need to set the **escalation.mail.format.html** setting in the `axException.xml` file to false.

You can edit the following template files as necessary:

- **master-template-html.vm** – This template controls the formatting and content of the escalation emails. You can use the following variables to insert dynamic content into the emails:
  - ${template.firstName}
  - ${template.lastName}
  - ${template.Email}
  - ${template.UserID}
  - ${summaryReport}

  A sample plain text template named `master-template.vm` is available in the same folder.
- **summary-report-template-html.vm** – This template controls the formatting and content of the escalation report that is inserted into the email message. You can edit the layout of the report, and the information displayed, but it must maintain valid Velocity syntax.

  A sample plain text template named `summary-report-template.vm` is available in the same folder.
- **subject-template-html.vm** – This template specifies the subject line of the email messages. It should only contain a single line with plain text whether you are using HTML for you other templates or not.

For detailed information on the syntax used in the email templates, see the Velocity User Guide (http://velocity.apache.org/engine/releases/velocity-1.5/user-guide.html).

To change email template settings:

1. Navigate to the folder on the AX Exception server where the escalation email templates are located. The default location is the `geronimo\var\config` subfolder where AX Exception is installed.
2. If you want to modify the format of the email messages sent, or the template files used, you need to modify the appropriate **escalation.mail** properties in the `axException.xml` configuration file.
3. Open the email template file you want to modify in a text editor and make the necessary changes.
4. Save and close the file.
5. If you made changes to the `axException.xml` configuration file you need to restart the Tomcat application server service for your changes to take effect.

   The next time escalation emails are processed, the updated email templates will be used.

4.3. Workflow configuration overview

In AX Exception, workflow is used to track and manage the resolution of exceptions generated by AX Server analytics. A workflow is a series of activities performed by one or more users in predefined roles to complete a business process.

The workflow for AX Exception is defined in an XML file that contains the following parts:

- **Workflow** – This is the root element for the workflow. All of the other elements in the workflow must be specified within the `<workflow>` element.
- **Steps** – The positions in a workflow (e.g. Assigned to Primary Reviewer, In Review). As the workflow progresses, it moves from one step to another. Steps are displayed in the **State** drop-down list in AX Exception.
- **Actions** – Specify the transitions that can take place within a particular step. An action can often result in a change of step. A step may have many actions connected to it.
- **Initial actions** – A special type of action used to start the workflow. At the very beginning of the workflow, there's no current state, and the workflow is not in any current step. The user must take some action to start off the process, and this set of possible actions to start the workflow is specified in the `<initial-actions>` section of the workflow file. An initial action is required to create a valid OSWorkflow configuration file, but it is not used by AX Exception.
- **State** – Each state is represented by the combination of step ID and status. A transition from one state to another cannot happen without an action occurring first. There is always at least one active state during the lifetime of a workflow.

4.4. Modifying the workflow XML file

If you want to make minor adjustments to the workflow XML file, you can edit the default workflow file that is installed with AX Exception. The procedure below outlines the process for modifying the existing file. If you want to make more extensive changes you can create a new XML file and update the `exceptionmgmt.workflow.file` configuration setting in the `axException.xml` file to point to it. For more information on creating a new workflow XML file, see Create a workflow XML file in the workflow tutorial.

To modify the workflow XML file:

1. Navigate to the folder on the AX Exception server where the workflow XML file is located. The default location is the `geronimo\var\config` subfolder where AX Exception is installed.
2. Open the `eccmExceptionManagementWF-Audit.xml` file in an XML editor and make the necessary changes. For information on the syntax required, see Workflow configuration syntax.
3. Save and close the file.
4. Restart the AX Exception component (com.acl/exceptionmgmt/<version>/car) in the Geronimo Console, or restart the ACL Analytics Exchange Service, on the AX Exception server.

4.5. Workflow configuration syntax

The workflow must include all required elements defined in the DTD. The following example shows a sample configuration file that includes the required elements. If you create a new workflow XML file, you need to include these elements, and then add the required steps for your workflow.

```
<?xml version=”1.0” encoding=”UTF-8”?>
<!DOCTYPE workflow PUBLIC
  "-//OpenSymphony Group//DTD OSWorkflow 2.8//EN"
  "http://www.opensymphony.com/osworkflow/workflow_2_8.dtd">
<workflow>
  <initial-actions>
    <action id=”0” name=”New” auto=”true”>
      <results>
        <unconditional-result id=”15” old-status=”Finished”
         status=”Underway” step=”1” />
      </results>
    </action>
  </initial-actions>
  <steps>
   ...
  </steps>
</workflow>
```

The first line identifies the file as an XML file. The document type definition on lines 2-4 defines the document type as a workflow, and specifies that it should be validated against the OSWorkflow DTD. This validation ensures that all of the required elements are included, that they are included in the correct order, and that they match any additional rules defined in the DTD. When you create a new workflow, or edit an existing workflow, you should use an XML

editor that automatically validates the file against the DTD as you add or modify elements in the workflow XML file.

4.5.1. Workflow

The <workflow> element is the root element for the workflow. It must contain an <initial-actions> element and a <steps> element that contains all of the individual steps in the workflow.

The <initial-actions> element is not used by AX Exception, but it is required to validate the XML file against the OSWorkflow DTD. You should not remove this element from, or edit it in, the workflow configuration file delivered with AX Exception. You must add the following <initial-actions> section to any new workflow configuration files you create:

```
<initial-actions>
  <action id="0" name="New" auto="true">
    <results>
      <unconditional-result id="15" old-status="Finished"
       status="Underway" step="1"/>
    </results>
  </action>
</initial-actions>
```

In addition to the two required elements, you can add optional <meta> elements if you want to record metadata. The name attribute for <meta> should only contain alphabetical characters. Dashes, spaces, etc. are not supported. Any <meta> elements, you define inside the workflow element, but outside any child elements, are available to all steps in the workflow. The following example defines a <meta> element named Priority that includes three different priority levels, separated by semi-colons:

```
<meta name=”Priority”>High;*Normal;Low</meta>
```

The asterisk is used to indicate that the Normal entry is the default value.

Important

You should ensure that <meta> element additions or changes are necessary before you update the AX Exception application with them. There is a limited number of database columns assigned to store application metadata. If you change the name attribute for a <meta> element, two database columns will be required, one to store the history of the original metadata element and one to store values for the new metadata element.

4.5.2. Steps

The <steps> element contains all of the individual steps in the workflow. You must include at least one <step> element to describe the steps in your workflow.

There are three categories of steps:

- **Initial step** – The step that starts the workflow. It includes actions that point to other steps that continue workflow, but other steps do not have actions that point to the initial step.
- **Open steps** – The steps the workflow progresses through while it is active. They include actions that point to other steps that continue the workflow, and other steps, including initial steps in the workflow point to them to continue the workflow.
- **Closed step** – The step that ends the workflow. Closed steps must be identified using the "ClosedStatus" metadata element. The following example shows the syntax required for closed steps:

```
<step id="99" name="Closed">
  <meta name="ClosedStatus"></meta>
</step>
```

When you are editing a workflow being used by the AX Exception application, you need to understand the different categories of steps because the application does not allow you to change a step from one category to another. For example, changing an open step to either an initial step, by removing actions that point to it from other steps, or to a closed step by adding the "ClosedStatus" <meta> element, is not allowed.

The reason for this restriction is that, to optimize performance of the web application, New, Open, and Closed exceptions are stored in different partitions in the database. When an exception is edited the workflow status is retrieved from correct partition in the database based on the step ID. This means that you cannot change a step from one type to another because exceptions at that step in the workflow will be stored in a different database partition. When you restart the application server and the changes take effect, any existing exceptions in the old state will not be found because they will be in a different partition.

4.5.3. Step

The <step> element is used to define a step in the workflow sequence. The <step> element has two required attributes: Name and ID. The ID value must be unique for steps within the workflow, and must be a value between 1 and 99. A step element must also include an <actions> element if it is an initial or open step. Closed steps can, but do not have to, include an <actions> element, but they must include the "ClosedStatus" <meta> element.

Each step can also include <meta> elements that apply to the specific step. The name attribute for <meta> should only contain alphabetical characters. Dashes, spaces, etc. are not supported.

AX Exception uses a special syntax for <meta> elements to record escalation information from the workflow to the exception record in the database. You need to add two <meta> elements to a step for each escalation level you want to add. The first <meta> element specifies the escalation level and user group to escalate the exception to in the following format:

```
escalation_<escalation level>_usergroup
```

The second <meta> element specifies the escalation level and the number of days before the exception should be escalated in the following format:

```
escalation_<escalation level>_duration
```

The following example uses both meta elements to specify that the first level escalation for the Primary Reviewer user group should happen after the exception has been in the state for 7 days:

```
<meta name=”escalation_1_usergroup”>Primary Reviewer</meta>
<meta name=”escalation_1_duration”>7</meta>
```

If you include more than one escalation level, the duration values for the escalation entries are cumulative. For example, if level 1 is 7 days and level 2 is 7 days, an exception at level 2 is escalated after 14 days.

You can specify more than one group at each level by separating the group names with a semicolon. Do not include any spaces before or after the semicolon.The following example specifies two user groups at the same escalation level:

```
<meta name=”escalation_1_usergroup”>Primary Reviewer;Secondary Reviewer;</meta>
```

If you want to escalate an exception as soon as it enters a particular state, you can create an escalation level of 0 with a duration of 0, as shown in the following example:

```
<meta name=”escalation_0_usergroup”>Secondary Reviewer</meta>
<meta name=”escalation_0_duration”>0</meta>
```

When the system processes escalation notifications and encounters this escalation rule, it will send out an email notification indicating the number of exceptions that have been moved into the state at Level 0, and the number of exceptions that have been moved to the next escalation level. Whenever Level 0 is listed in an escalation email it indicates that the exceptions listed have been immediately processed and moved to the next escalation level.

4.5.4. Action

The <actions> element contains one or more <action> elements that can take place to transition from the current step to the next step. The <action> element has two required attributes: Name and ID. The ID value must be unique within the workflow. Each action must have a <results> element that includes one unconditional result, but an also include one or more conditional results. Unconditional results must be assigned an ID that is unique for results in the workflow, and they must use the step attribute to specify the next step to move to.

The following is an example of an unconditional result used to move to step 99, the final step in the workflow:

```
<unconditional-result id=”9” step=”99” old-status=”Finished” status=”Underway”>
```

The “old-status” and “status” attributes are not used by the AX Exception application, but are required by the OSWorkflow DTD. You can also use the <conditional-result> element to implement more complex workflow logic.

You can also use the <restrict-to> element to limit the action to a particular user role, or set of user roles. The following syntax is used to restrict an action to a particular user role:

```
<restrict-to>
 <conditions>
  <condition type=”class”>
   <arg name=”class.name”>
    com.acl.exceptionmgmt.util.wfUserRoleEntityCondition
   </arg>
   <arg name=”role”>Secondary Reviewer</arg>
  </condition>
 </conditions>
</restrict-to>
```

The <arg name=”class.name”> value above provides a secure way to query for the role authorizations of the user currently logged in to the web application when the workflow is being used. You can change the condition by substituting another role used to manage access to the AX Exception web application.

If you want to restrict an action to multiple user roles you can use the following syntax:

```
<restrict-to>
  <conditions type="OR">
    <condition type="class" >
     <arg name=”class.name”>
      com.acl.exceptionmgmt.util.wfUserRoleEntityCondition
     </arg>
     <arg name="role">Primary Reviewer</arg>
    </condition>
  <condition type="class" >
    <arg name=”class.name”>
     com.acl.exceptionmgmt.util.wfUserRoleEntityCondition
    </arg>
    <arg name="role">Secondary Reviewer</arg>
    </condition>
  </conditions>
</restrict-to>
```

You can also modify the above example to specify that a user must belong to both roles by changing the second line to an AND condition: <conditions type=”AND”>

Each action can also include <meta> elements that apply to that specific action. The name attribute for <meta> should only contain alphabetical characters. Dashes, spaces, etc. are not supported. In the AX Exception web application, metadata for an action is displayed in a drop-down list when the step the action belongs to is selected.

4.6. Workflow tutorial

The following tutorial describes the steps you need to complete to create a simple AX Exception workflow. In order to complete this tutorial, you should have access to an XML editor that is capable of validating an XML file against a DTD as you type.

4.6.1. Create a workflow XML file

The workflow rules and logic for the AX Exception application are configured using a third-party workflow management component called OSWorkflow. The first step you need to complete to create a workflow is to define the basic structure of the XML file used to specify your workflow.

Figure 1. Basic workflow structure

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow PUBLIC "-//OpenSymphony Group//DTD OSWorkflow 2.8//EN"
"http://www.opensymphony.com/osworkflow/workflow_2_8.dtd">
<workflow>
  <initial-actions>
    <action id="0" name="New" auto="true">
      <results>
        <unconditional-result id="1" old-status="Finished" status="Underway" step="1"/>
      </results>
    </action>
  </initial-actions>
  <steps>

  </steps>
</workflow>
```

To create the basic workflow syntax:

1. Open you XML editor and create a new XML file.
2. Type in the required structure listed in Figure 1, or copy the structure from an existing workflow.

   This creates a valid OSWorkflow structure that you can use as a starting point for creating your workflow. It is not a valid configuration file until you add the individual steps to the workflow.

> **Tip**
>
> In the screenshot above the closing steps tag is underlined, indicating that more information must be specified inside the tag. You should be familiar with the way your XML editor indicates problems with the structure of the XML file.

Related concepts

Workflow configuration syntax

4.6.2. Add global metadata

Global metadata is metadata that is available to all steps in the workflow. Adding the following <meta> tag to the top of your workflow enables users to select the priority of an exception when they change the workflow state in AX Exception:

```
<meta name="Priority">High;*Normal;Low</meta>
```

These priority options are displayed as a drop-down list in the web application. You can specify the default value to appear in the drop-down list by adding an asterisk before the entry. For example, Normal is the default priority value in the example above.

To add global metadata to your workflow:

1. Locate the <workflow> tag in your workflow XML file.
2. Enter the following <meta> tag immediately below the <workflow> tag, and before the <initial-actions> tag:

```
<meta name="Priority">High;*Normal;Low</meta>
```

4.6.3. Add workflow steps

The next step in defining your workflow is adding steps that define the positions or statuses in your workflow. We will define three steps to create a simple workflow: Assigned to Primary Reviewer, In Review, and Complete.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow PUBLIC "-//OpenSymphony Group//DTD OSWorkflow 2.8//EN"
"http://www.opensymphony.com/osworkflow/workflow_2_8.dtd">
<workflow>
  <meta name="Priority">High;*Normal;Low</meta>
  <initial-actions>
    <action id="0" name="New" auto="true">
      <results>
        <unconditional-result id="1" old-status="Finished" status="Underway" step="1"/>
      </results>
    </action>
  </initial-actions>
  <steps>
      <step id="1" name="Assigned to Primary Reviewer">

      </step>
      <step id="2" name="In Review">

      </step>
      <step id="99" name="Complete">
        <meta name="ClosedStatus"></meta>
      </step>
  </steps>
</workflow>
```

To add workflow steps:

1. Locate the <steps> tag in your workflow XML file.
2. Enter the following between the opening and closing <steps> tags:

```
<step id="1" name="Assigned to Primary Reviewer">
</step>
```

```
<step id="2" name="In Review">
</step>
<step id="99" name="Complete">
<meta name="ClosedStatus"></meta>
</step>
```

4.6.4. Add actions to your steps

After you define the steps in your workflow, you need to add actions. Actions are used to define the rules that control how the workflow transitions from one step to the next. In the following example, an action called "Assign alert to Secondary Reviewer" has been added to the "Assigned to Primary Reviewer" step that moves the workflow to step 2 (In Review).

```
<step id="1" name="Assigned to Primary Reviewer">
  <actions>
    <action id="1" name="Assign alert to Secondary Reviewer">
      <results>
        <unconditional-result id="1" old-status="Finished"
         status="Underway" step="2" />
      </results>
    </action>
  </actions>
</step>
```

Note

The "old-status" and "status" attributes are not used by the AX Exception application, but they are required by the OSWorkflow DTD. The key piece of information in the `<unconditional-result>` tag is the step attribute, which indicates the ID of the step to transition to.

To add actions to a step:

1. Locate the Assigned step in your workflow XML file.
2. Enter the following between the opening and closing `<step>` tags:

```
<actions>
  <action id="1" name="Assign alert to Secondary Reviewer">
    <results>
      <unconditional-result id="1" old-status="Finished
       status="Underway" step="2" />
    </results>
  </action>
</actions>
```

4.6.5. Add escalation metadata

Escalation metadata is defined within steps, and it is used to specify the levels of escalation for the step. Escalations are used to alert particular users if an exception remains in the same state longer than the specified duration period. The following escalation metadata defines a first escalation level that notifies the Primary Reviewer if any exceptions remain in the Assigned to Primary Reviewer step after 7 days.

```
<step id=”1” name=”Assigned to Primary Reviewer”>
  <meta name=”escalation_1_usergroup”>Primary Reviewer</meta>
  <meta name=”escalation_1_duration”>7</meta>
  <actions>
   ...
  </actions>
</step>
```

To add escalation metadata:

1. Locate the Assigned step in your workflow XML file.
2. Enter the following immediately after the opening <step> tag:

```
<meta name=”escalation_1_usergroup”>Primary Reviewer</meta>
<meta name=”escalation_1_duration”>7</meta>
```

4.6.6. Add restrictions

Restrictions are used to limit actions to a particular user role, or set of user roles. The effect of limiting access to an action is to limit the users that can move the workflow from the current step to the next step specified by the action. The following restriction specifies that only Secondary Reviewers can change the workflow status from the In Review to Complete.

```
<step id="2" name="In Review">
  <meta name="escalation_2_usergroup">Primary Reviewer</meta>
  <meta name="escalation_2_duration">14</meta>
  <actions>
   <action id="3" name="Close">
      <restrict-to>
        <conditions>
          <condition type="class">
            <arg name="class.name">com.acl.exceptionmgmt.util.WFUserRoleEntityCondition</arg>
            <arg name="role">Secondary Reviewer</arg>
          </condition>
        </conditions>
      </restrict-to>
      <results>
        <unconditional-result id="3" old-status="Finished" status="Underway" step="99"/>
      </results>
    </action>
   </actions>
</step>
<step id="99" name="Complete">
  <meta name="ClosedStatus"></meta>
</step>
```

To add a restriction:

1. Locate the In Review step in the workflow XML file.

2. Insert the following tags immediately after the opening <action> tag:

```
<restrict-to>
  <conditions>
    <condition type=”class”>
      <arg name=”class.name”>com.acl.exceptionmgmt.util.
          WFUserRoleEntityCondition</arg>
      <arg name=”role”>Secondary Reviewer</arg>
    </condition>
  </conditions>
</restrict-to>
```

4.6.7. View the completed workflow

Once you have completed the previous steps in this tutorial, you should have a workflow file with identical content to the workflow in Figure 1.

Figure 1. Completed workflow

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow PUBLIC "-//OpenSymphony Group//DTD OSWorkflow 2.8//EN"
"http://www.opensymphony.com/osworkflow/workflow_2_8.dtd">
<workflow>
  <meta name="Priority">High;*Normal;Low</meta>
  <initial-actions>
    <action id="0" name="New" auto="true">
      <results>
        <unconditional-result id="1" old-status="Finished" status="Underway" step="1"/>
      </results>
    </action>
  </initial-actions>
  <steps>
    <step id="1" name="Assigned to Primary Reviewer">
      <meta name="escalation_1_usergroup">Primary Reviewer</meta>
      <meta name="escalation_1_duration">7</meta>
      <actions>
        <action id="2" name="Assign alert to Secondary Reviewer">
          <restrict-to>
            <conditions>
              <condition type="class" >
                <arg name="class.name">com.acl.exceptionmgmt.util.WFUserRoleEntityCondition</arg>
                <arg name="role">Primary Reviewer</arg>
              </condition>
            </conditions>
          </restrict-to>
          <results>
            <unconditional-result id="2" old-status="Finished" status="Underway" step="2"/>
          </results>
        </action>
      </actions>
    </step>
    <step id="2" name="In Review">
      <meta name="escalation_2_usergroup">Primary Reviewer</meta>
      <meta name="escalation_2_duration">14</meta>
      <actions>
       <action id="3" name="Close">
          <restrict-to>
            <conditions>
              <condition type="class">
                <arg name="class.name">com.acl.exceptionmgmt.util.WFUserRoleEntityCondition</arg>
                <arg name="role">Secondary Reviewer</arg>
              </condition>
            </conditions>
          </restrict-to>
          <results>
            <unconditional-result id="3" old-status="Finished" status="Underway" step="99"/>
          </results>
        </action>
       </actions>
     </step>
     <step id="99" name="Complete">
      <meta name="ClosedStatus"></meta>
    </step>
  </steps>
</workflow>
```

5. AX Exception configuration files

The topics in this section provide information about the files that contain application configuration settings.

Filename	Location	Description
aclAuditExchange.xml	geronimo\var\config	Stores configuration settings for AX Server. If AX Exception is installed on a separate server, a limited subset of the properties are specified.

Filename	Location	Description
aclCasClient.xml	geronimo\var\config	Stores the client configuration settings for ACL Analytics Exchange authentication.
aclDatabase.xml	geronimo\var\config	Stores database connection information for the ACL Analytics Exchange database.
axException.xml	geronimo\var\config	Stores most configuration settings for AX Exception including database connection settings, UI configuration settings, and escalation email settings.
axExceptionAdmin.xml	geronimo\var\config	Stores configuration settings for the AX Exception Administration web application.
axExceptionConnect.xml	geronimo\var\config	Stored encrypted username and password information for the AX Exception database account and mail server account used for escalation emails.
dataloaderCommon.properties	Dataloader\sessions\template\conf folder on AX Server or AX Engine Node.	Stores configuration settings for the AX Exception Dataloader, which is used to publish exception information from AX Server or AX Engine Node to AX Exception.
exActivation.properties	geronimo\var\config	Stores the AX Exception version number.
wrapper.conf	geronimo\var\config	Stores configuration settings for the ACL Analytics Exchange Service.

Section contents

5.1. aclAuditExchange.xml

If AX Exception is installed on a separate server from AX Server, this configuration file includes the subset of configuration settings required for ACL Analytics Exchange authentication. For a complete list of settings if AX Exception and AX Server are installed on the same server, see “aclAuditExchange.xml” in the *ACL Analytics Exchange Server Administrator Guide*.

Property	Description
DefaultDomain	The Active Directory domain to use by default if a user does not specify a domain when they log in. This property corresponds to the **Default Active Directory domain** setting in the AX Server Configuration web application.
aclseProfileName	The name of the server profile used to connect to server tables. This fields should be left at the default value (AuditExchange) unless there is a specific reason to change it.
AXHostname	Specifies the hostname for AX Server. The value must be entered with uppercase characters. For example: AXCORE.ACL.COM
AXHttpsPort	Specifies the port used for encrypted communications with AX Server. The default value is 443. The default value in versions prior to 5.0 was 8443.

5.2. aclCasClient.xml

This configuration file stores the settings required to connect to the AX Server authentication service.

Property	Description
cas.securityContext.casServerHost	The hostname of the AX Server. This property value must match the Common Name (CN) value specified in your security certificate.
cas.securityContext.casServerPort	The port to use to connect to the ACL Analytics Exchange authentication component (CAS). The default port is 8443.

5.3. aclDatabase.xml

This configuration file stores configuration information for the ACL Analytics Exchange database. The file is encrypted when the ACL Analytics Exchange Service starts. For information on changing settings in this file, see Modifying aclDatabase.xml.

Because the configuration file is an XML file, you must use special syntax if the passwords include characters that cannot be included inside an XML tag (<,>,&,", or '). In order for the AX Server application to read passwords that include these characters, you must wrap the password in a CDATA tag. For example, the following standard syntax for specifying "pa&&word" would not be read correctly by the application:

```
<entry key="password">pa&&word</entry>
```

In order to specify the password in a way the application can read it, you need to enter the following:

```
<entry key="password"><![CDATA[pa&&word]]></entry>
```

Property	Description
username	The username required to connect to the ACL Analytics Exchange database.
password	The password required to connect to the ACL Analytics Exchange database.
host	The hostname or IP address of the server where the ACL Analytics Exchange database is located.
port	The port used to connect to the ACL Analytics Exchange database server. For PostgreSQL the default value is 5432. For Oracle the default value is 1521 for non-SSL or 2482 for SSL.
database	The name of the ACL Analytics Exchange database.
driver	The database driver used to connect to the database. For PostgreSQL the value required is: *jdbc:postgresql*. For Oracle the value required is: *jdbc:oracle*
ssl	Specifies whether or not SSL should be used to secure communications with the ACL Analytics Exchange database. The permitted values are “true” (use SSL) or “false” (do not use SSL). **Note:** If you update the SSL setting in this property file, you also need to update the AclAuditExchangeDbPool database pool in Tomcat application server to match your SSL configuration.

5.4. axException.xml

This configuration file stores most of the configurable settings for AX Exception including database configuration settings, UI configuration settings, and escalation email settings.

Property	Description
exceptionmgmt.url	The URL for the root of the AX Exception web server. For example: https://axexception.acl.com

Property	Description
exceptionmgmt.jdbc.url	Specifies the connection string used to access the AX Exception database. For SQL Server databases, the connection string is specified in the following format: jdbc:sqlserver://<*server*>:<*port*>;database=<*database_name*> The <*server*> value can be specified using either an IP address or the server host name. For example: jdbc:sqlserver://127.0.0.1:1433;database=AXExceptionDB For Oracle databases, the connection string is specified in the following format: jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=<protocol>) (HOST=<server>)(PORT=<port>)) (CONNECT_DATA=(SERVICE_NAME=<service>))) <protocol> is either “tcp” for an unencrypted connection or “tcps” for SSL. <server> is the server name or IP address for the Oracle server. <port> is the port the Oracle listener is configured on. <service> is the SID to connect to. For example: jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps) (HOST=axdb.acl.com)(PORT=2484)) (CONNECT_DATA=(SERVICE_NAME=oracle)))
exceptionmgmt.jdbc.driverClassName	Specifies the database driver used to access the AX Exception database. If SQL Server is the database platform it must be set to the following: com.microsoft.sqlserver.jdbc.SQLServerDriver If Oracle is the database platform it must be set to the following: oracle.jdbc.driver.OracleDriver
exceptionmgmt.jdbc.pool.initialSize	Specifies the initial number of connections in the database connection pool.
exceptionmgmt.jdbc.pool.maxActive	Specifies the maximum number of connections in the database connection pool.
exceptionmgmt.jdbc.pool.maxWait	Specifies the maximum time to wait for a connection before failing. The time is specified in milliseconds.
exceptionmgmt.jdbc.pool.maxIdle	Specifies the maximum number of active connections that can remain idle in the pool, without extra ones being released, or zero for no limit.
exceptionmgmt.jdbc.pool.minIdle	Specifies the minimum number of active connections that can remain idle in the pool, without extra ones being created, or zero to create none.
exceptionmgmt.jdbc.pool.validationQuery	Specifies the SQL query to use to validate connections from the database pool. The query must be a SELECT statement that returns at least one row.
exceptionmgmt.workflow.file	Specifies the name and file path where the XML workflow definition file is stored.
exception.management.date.format	Specifies the formatting of dates in the application.
exception.management.currency.numberFormat	This property is not used in this release.
exception.management.date.interval.max	Specifies the maximum number of days users can select as the interval between the Date From and Date To values when they select a closed state from the **State** drop-down list.
exception.management.date.interval.min	Specifies the minimum number of days users can select as the interval between the Date From and Date To values when they select a closed state from the **State** drop-down list.

Property	Description
exception.management.page.size.default	Specifies the default number of exceptions displayed on each page of the table in the **Exception Details** page.
exception.management.page.size.max	Specifies the maximum value the user can input in the **exceptions per page** text box. If the user specifies a number higher than the maximum value the text box is set to the maximum value. The value entered controls the number of exceptions displayed on each page of the table in the **Exception Details** page.
exception.management.lock.duration	Specifies the number of seconds that the exceptions users have selected are locked for their user accounts after they enter the **Edit Exception** page. This property is not supported in this release.
velocityEngine.resourceLoaderPath	Specifies the path the application used to send escalation emails loads required resources from. The default location is the **geronimo/var/config** subfolder where AX Exception is installed.
escalation.velocity.master-template	Specifies the location of the master template that controls the formatting and content of the escalation emails.
escalation.velocity.summary-report-template	Specifies the location of the template that controls the formatting and content of the escalation report that is inserted into the master template.
escalation.velocity.subject-template	Specifies the location of the template that sets the subject line of escalation email messages. The template should only contain a single line with plain text.
exceptionmgmt.fileupload.maxfilesize	Specifies the maximum size of a file that can be uploaded as an attachment. The value is specified in bytes.
exception.management.job.cron	Specifies the frequency of the job that runs to clean up any attachments that have been uploaded to the server, but are not associated with an exception because the end-user did not save their updates to the exception details. The **cron.txt** file in the **geronimo\var\config** subfolder provides the syntax and examples of frequency setting.
exception.jobs.timezone	Specifies the timezone used to calculate the time to run the cleanup job to delete unused attachments. The **timezones.txt** file in the **geronimo\var\config** subfolder provides a list of valid timezones.
escalation.mail.from	Specifies the sender email address to use for notification emails.
escalation.mail.format.html	Specifies the format of the notification emails. The value should be “true” if your email templates include HTML code, or “false” if they are plain text.
escalation.mail.host	Specifies the IP address or hostname of the mail server used to send email notifications. The mail server you specify must support the Simple Mail Transport Protocol (SMTP), which is a widely supported standard internet protocol. The SMTPs extension to the protocol is not supported.
escalation.mail.port	Specifies the port to connect to the mail server in on.
escalation.aftermaxlevel.duration	Specifies the duration in days used by the escalation process to calculate the new due date when the exception is already at its maximum escalation level.

Property	Description
escalation.job.cron	Specifies the frequency of the escalation job that searches the database for exceptions that should be escalated and sends email notifications. The **cron.txt** file in the **geronimo\var\config** subfolder provides the syntax and examples of frequency setting.
escalation.jobs.timezone	Specifies the timezone that applies to the escalation job. The **timezones.txt** file in the **geronimo\var\config** subfolder provides a list of valid timezones.
exceptionmgmt.dataloading.allowIPs	Specifies a list of IP addresses that are permitted to upload information generated by the Dataloader. This should be configured in one of the following ways: • Leave the value blank, or comment out the setting, if you do not want to restrict data publishing. • Enter the IP address of AX Server if it is being used to process analytics, and you want to restrict data publishing to this server. • Enter the IP address of each AX Engine Node that is being used to process analytics if you want to restrict data publishing to those individual servers. Multiple IP addresses must be separated by semicolons.

5.5. axExceptionAdmin.xml

This configuration file stores settings for the AX Exception Administration web application.

Property	Description
useradministration.exceptionmgmt.url	Specifies the URL for the AX Exception web application. The hostname specified in the URL must match the hostname specified in your security certificate.
useradministration.entities.xml.path	Specifies the folder where the **entities.xml** file will be temporarily saved. The **entities.xml** file lists the entity IDs available in AX Exception.

5.6. axExceptionConnect.xml

This configuration file stores username and password information for the AX Exception database account and the escalation email account. For information on modifying this file, see Modifying axExceptionConnect.xml.

Because the configuration file is an XML file, you must use special syntax if the passwords include characters that cannot be included inside an XML tag (<,>,&,", or '). In order for the AX Exception application to read passwords that include these characters, you must wrap the

password in a CDATA tag. For example, the following standard syntax for specifying "pa&&word" would not be read correctly by the application:

```
<entry key="escalation.mail.password">pa&&word</entry>
```

In order to specify the password in a way the application can read it, you need to enter the following:

```
<entry key="escalation.mail.password"><![CDATA[pa&&word]]></entry>
```

Property	Description
exceptionmgmt.jdbc.username	Specifies the username to access the AX Exception database.
exceptionmgmt.jdbc.password	Specifies the password to access the AX Exception database.
escalation.mail.username	Specifies the username of the account used to send email notifications.
escalation.mail.password	Specifies the password of the account used to send email notifications.

5.7. dataloaderCommon.properties

This configuration file stores configuration settings for the AX Exception Dataloader, which is used to publish exception information from AX Server or AX Engine Node to AX Exception. The file is located on the server where AX Server is installed in the `Dataloader\sessions\template\conf` subfolder.

Property	Description
baseServerUrl	This property is not used in this release. The location of AX Exception is set by the **Data Upload URL** setting in the AX Server Configuration web application.
trustStoreFile	The path and filename of the keystore used to encrypt communications between the AX Exception and AX Server.
trustStorePass	The password for the keystore.
keyStoreFile	This property is not used in this release.
keyStorePass	This property is not used in this release.
username	This property is not used in this release.
password	This property is not used in this release.

Property	Description
timeZone	The timezone that the ACL results for the current data set were produced in. This is required for correct timekeeping where the files from other systems may be collected/analyzed in different timezones. UTC dates are used in the database. A complete list of supported timezones is available in timezones.txt in the **TomCat\conf** subfolder. If this property is not set, the default local system timezone will be used. **Note:** This property is set to America/Vancouver by default. You need to set it to the timezone you want to use, or remove it to use the local system timezone.
dataLoadChunkSize	The maximum number of records to send per connection (for data loading only). Setting this value to zero (0), or lower, causes the application to send all data for each analytic at once. This setting is only used if you are using certificate-based authentication.
skipMissingResultFiles	Set to “true” to specify that the Dataloader should continue processing when a corresponding result file is not found for an analytic. If this value is set to “false” a File Not Found error is generated whenever a result file cannot be found for an analytic. The default value is “true”.
debugMode	Specifies whether debug mode is enabled (true) or disabled (false). The default value is “false”.
testMode	Specifies whether test mode is enabled (true) or disabled (false). If test mode is enabled, the data loader process runs, but does not connect to the server. The default value is “false”.
MailLogger.*value*	The MailLogger properties are not supported in this release.

5.8. exActivation.properties

This configuration file stores license activation settings for AX Exception including the serial number and user limit. You should not modify this file unless your licensing terms change.

Property	Description
serialNumber	This property is not used in this release.
softwareVersion	Specifies the installed release number.
usersLimit	This property is not used in this release.
activationKey	This property is not used in this release.

5.9. system.properties

The `system.properties` file is the main configuration file for Tomcat application server. The ACL Analytics Exchange Service must be restarted for changes made to this file to take effect.

Property	Description
-com.acl.ax.project.encoding	For Unicode installations of AX Server this parameter must be present, and set to UTF-16LE. This property is not present for non-Unicode installations of AX Server, or in Unicode installations if AX Exception is the only application installed on the server.
-cas.authentication.mode	Specify “form” for form-based authentication, or “silent” for Integrated Windows Authentication. The value entered must be lowercase. This property is not present on AX Engine Node, or if AX Exception is the only application installed on the server.
-javax.net.ssl.trustStore	The path to the keystore file used by Tomcat. The default location is the App/keystores/MyKeyStore subfolder.
-javax.net.ssl.trustStorePassword	The password for keystore being used.
-oracle.net.ssl_cipher_suites	The cipher suites that will be used for SSL connections. The supported values are entered as a comma separated list.
-com.acl.ax.server.deleteJobDirectories	This startup parameter isn’t included by default, but it can be added and set to “false” if you want to prevent the system from deleting analytic job directories after each job completes. This can be useful if you need to troubleshoot unexpected analytic results.
-sun.security.krb5.msinterop.kstring	When set to “true” the Kerberos string used for authentication is encoded as UTF-8. This property is not present on AX Engine Node, or if AX Exception is the only application installed on the server.
-oracle.jdbc.defaultNChar	This property must be set to “true” in order to support Oracle database instances with non-UTF8 system encodings. This property is ignored if Oracle is not the system database platform. This property is not present if AX Exception is the only application installed on the server.
-oracle.jdbc.ConvertNcharLiterals	This property must be set to “true” to preserve Unicode analytic and entity names being published from AX Server to AX Exception. This property is ignored if Oracle is not the system database platform, an only needs to be set on the server where AX Exception is installed.
-exceptionmgmt.jdbc.database.protocol	Valid values are “tcp” for non-SSL and “tcps” for SSL (Oracle only). The value entered must be lowercase. If “tcps” is set for Oracle, the required configuration must be completed to allow SSL connections to the Oracle database instance. This property is only present if AX Exception is installed on the server.
-exceptionmgmt.jdbc.database.type	Valid values are “oracle” and “mssql”. The value entered must be lowercase. This property is only present if AX Exception is installed on the server.