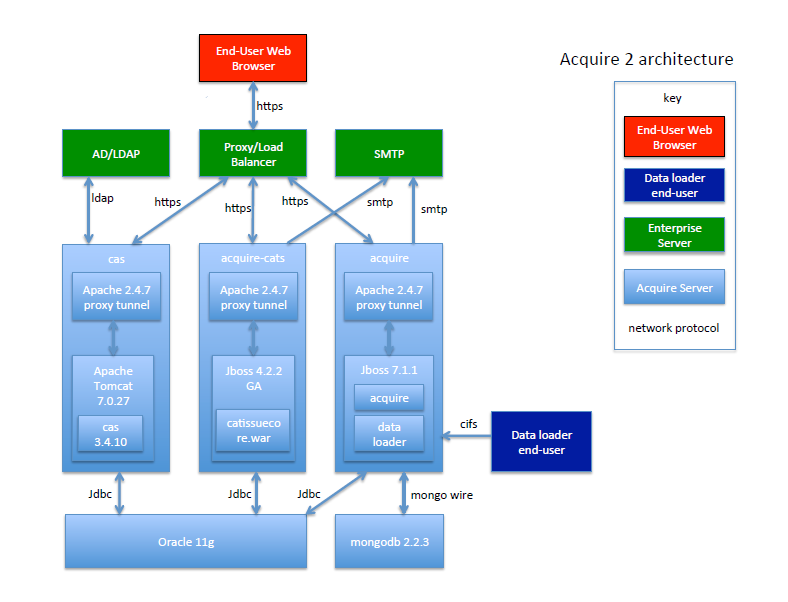
|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Last Modified by** |
|  |  |  |
|  |  |  |

**Acquire Deployment Guide**

****

1. **ARC Document for Acquire Architecture**



1. **Tech Stack Requirements:**

While Acquire was deployed on three separate RHEL Linux virtual machines, it is possible to deploy all modules on the same server.

Middle Ware:

Acquire 2.3

JBOSS 7.X

Catissue v1.2

JBOSS 4.2.2.GA

ANT 1.7

CAS 3.4.10

Tomcat 7.0.27

Acquire should be deployed in the following order:

1. Central CAS ---- 2. CaTissue ------ 3. Acquire

**3. SETTING UP A BASIC STANDALONE CAS INSTALLATION:**

The instructions contained within the external reference documentation links above will result in a basic CAS system.

Deployment Tip: Make sure that shell environment variables are all defined correctly if building on a Unix/Linux system, especially making sure to define the following:

**Required Environment Variables:**

TOMCAT\_HOME

JAVA\_HOME

CATALINA\_HOME

MAVEN\_HOME

M2\_HOME

M2

The system’s PATH environment variable must include $M2 and $JAVA\_HOME/bin.

To confirm that CAS is functional, follow the instructions within the "Maven2 WAR Overlay Method" link above.

**CAS CUSTOMIZATIONS FOR ACQUIRE:**

Information: The Maven WAR Overlay Method webpage linked above provides instructions for different types of authentication methods. Baylor College of Medicine utilizes LDAP and local DB lookup for those not in our domain and these methods were utilized by the authors for their Acquire implementation.

General LDAP instructions can be found at:

https://wiki.jasig.org/display/CASUM/LDAP

Acquire has been validated with the FastBindLdapAuthenticationHandler method.

DEPLOYMENT TIP: When using LDAP as an authentication method, the structure of your organization's LDAP schema must be known to filter on the "%u" field.

Information: To provide single sign-on authentication for users of Acquire and CaTissue, CAS must encrypt the password passed by the user in the same format that CaTissue and Acquire use to store the password in the database so that they can be compared without unencrypting the password. To achieve this, we duplicated the caTissue encryption algorithm "desPasswordEncoder" that hashes the password and placed it in CAS. The system can then compare the generated hash from CAS against the hash stored for that username in the Acquire and caTissue databases to check for a match. As a third source of authentication, the standalone CAS for Acquire refers to an institutional LDAP server for user authentication.

**To Configure CAS for authenticating against your local Acquire db schema and against your institutions LDAP, make the following configuration changes to the deployed CAS.**

**References –Acquire Code Repository Supplemental Materials Directory Files:**

* catissue-password-ecryptor-0.0.1-SNAPSHOT.jar
* (CAS) pom.xml
* deployerConfigContext.xml

**Configuring CAS**

**Step 1. deployerConfigContext.xml:**

Acquire comes with a customized password encoder that is used by CAS for authentication. This is configured as bean named desPasswordEncoder in the CAS deployment configuration file, i.e. in deployerConfigContext.xml. The PasswordEncoder entry must be made in the deployerConfigContext.xml since it is referred to in the catissue-password-ecryptor-0.0.1-SNAPSHOT.jar

The following entry should be in your deployerConfigContext.xml:

-<bean class="org.jasig.cas.adaptors.jdbc.SearchModeSearchDatabaseAuthenticationHandler" id="SearchModeSearchDatabaseAuthenticationHandler" autowire="default" lazy-init="default" scope="singleton" abstract="false">-<property name="tableUsers"><value>your\_user\_table</value></property>-<property name="fieldUser"><value>your\_user\_name</value></property>-<property name="fieldPassword"><value>your\_password</value></property><property name="dataSource" ref="dataSource"/><property name="passwordEncoder" ref="desPasswordEncoder"/></bean>

This reference **is already made** in the deployerConfigContext.xml file found in the Supplemental Materials directory.

In addition to the bean listed above, the deployerConfigContext.xml from the Supplemental Materials directory **must be edited to insert the database connection string for your DB** installation.

**Edit the following highlighted lines in the deployerConfigContext.xml to match your environment:**

<bean class="com.github.inspektr.audit.support.Slf4jLoggingAuditTrailManager" id="auditTrailManager"/>-<bean class="org.springframework.ldap.core.support.LdapContextSource" id="contextSource"><property name="pooled" value="false"/><property name="url" value="ldap://your.ldap.server.edu"/>-<property name="baseEnvironmentProperties">-<map>-<entry>-<key><value>java.naming.security.authentication</value></key><value>simple</value></entry></map></property></bean><bean class="edu.bcm.dldcc.common.cas.CatissuePasswordEncoder" id="desPasswordEncoder"/>-<bean class="org.apache.commons.dbcp.BasicDataSource" id="dataSource">-<property name="driverClassName"><value>oracle.jdbc.driver.OracleDriver</value></property>-<property name="url"><value>jdbc:oracle:thin:@your.oracle.server.edu:1521:dbsid</value></property>-<property name="username"><value>your\_catissue\_username</value></property>-<property name="password"><value>your\_catissue\_password</value></property></bean>-<bean class="org.apache.commons.dbcp.BasicDataSource" id="acquireDataSource">-<property name="driverClassName"><value>oracle.jdbc.driver.OracleDriver</value></property>-<property name="url"><value>jdbc:oracle:thin:@your.oracle.server.edu:1521:dbsid</value></property>-<property name="username"><value>your\_acquire\_username</value></property>-<property name="password"><value>your\_acquire\_password</value></property></bean>

**Step 2. Deploying the Encoder Library:**

The encoder class uses Triple DES (3DES) algorithm. The encoder library can be found in the supplementary materials and is named catissue-password-ecryptor-0.0.1-SNAPSHOT.jar. Two different methods may be used to make this password encoder available to CAS, depending on the source of the deployed CAS server.

1st Method:

If CAS is being deployed from a binary webarchive (.war file), unpack the CAS webarchive and add the catissue-password-ecryptor-0.0.1-SNAPSHOT.jar file to the WEB-INF/lib directory of the expanded CAS.war. Then rezip and deploy the CAS.war file.

2nd Method:

Alternatively, If CAS is being built from source, and if using Maven to build the web archive, add the encoder jar file as a dependency before building the source. For example, the instruction below installs the jar to a local repo with the group id of edu.bcm.dldcc and artifact id of tripledes:

Example: Applicable when using Maven

*mvn install:install-file -Dfile=/directory-where-the-jar-file-is-located/catissue-password-ecryptor-0.0.1-SNAPSHOT.jar -DgroupId=edu.bcm.dldcc -DartifactId=tripledes -Dversion=1.0.0 -Dpackaging=jar*

Next, add the encoder jar as a dependency section in the CAS source pom.xml file:

 <dependency>

           <groupId>edu.bcm.dldcc </groupId>

           <artifactId>tripledes </artifactId>

           <version>1.0.0</version>

           <type>jar</type>

           <scope>runtime</scope>

         </dependency>

Configuration Tip: Sections highlighted in grey can be changed to be organizational specific but must match the Maven command

1. **Deploying CaTissue Suite:**
   1. **Link to NCI deployment Guide**

CaTissue is a National Cancer Institute initiative. Source code and Installable can be found on the NCI website at, <https://wiki.nci.nih.gov/display/caTissuedoc/2+-+Deploying+the+Web+Application+v1.2>

* 1. **Modifications to CaTissueInstall.properties file**

Instructions for site specific installation properties are stored and read on deployment from the caTissueInstall.properties file in the caTissue installation directory. CaTissue’s default deployment installs CAS to the CaTissue server for authentication. For deployment of CaTissue with Acquire, the CaTissueInstall.properties file must be configured to refer to a central CAS authentication server. This is done by modifying the following section of the CaTissueInstall.properties file.

EX:

* 1. **Additions to CaTissue db – See Catissue Database Change Log**

To allow caTissue Suite 1.2 to function as a module of Acquire 2.3, a few additions were added to caTissue’s database. For deployment purposes, both the Acquire and CaTissue schemas should exist in the same relational database instance. These additions allow Acquire to be notified of changes within caTissue and therefore to keep its data synchronized with caTissue. Other tables that were not created for Acquire, but are utilized by the system are also noted in the *CaTissue Database Change Control File* found in the supplemental materials directory of the Acquire source code repository on GitHub. These new tables and triggers need to be created in the CaTissue database post installation of CaTissue v1.2.

1. **Deploying Acquire:**
   1. **Location of Acquire Source Code**

Acquire source code has been deposited in the GitHub repository. The source code can be found at: HERE

* 1. **Use of Maven on Github**

See Ben

* 1. **Instructions for building Acquire**
  2. Acquire ReadMe

Acquire contains a ReadMe file in the Acquire source code directory of the source code. All associated libraries and modules can be found in this file along with instructions for configuring JBOSS for use with Acquire.

* 1. Systems modifications to JBOSS

JBOSS 7 must be installed and running on the Acquire server prior to deploying the Acquire source code. In addition, the following customizations must be made to JBOSS prior to deployment.

1. Add Oracle libraries as a Module in JBoss.

2. Update Hibernate libraries, including Envers, in Jboss to 4.2.2

* 1. Modifications to JBOSS Standalone-full.xml File

Since Acquire makes use of messaging for notifications, it requires use of standalone-full.xml for configuration information. The following information documents the changes that need to be made to the Standalone-full.xml file in the JBOSS … directory

Add the following to the <datasources> element of standalone-full.xml:

<xa-datasource jta="true" jndi-name="java:/jdbc/caTissueBase" pool-name="CaTissueBaseDS" enabled="true" use-java-context="true">

<xa-datasource-property name="URL">

database URL

</xa-datasource-property>

<driver>oracle</driver>

<xa-pool>

<min-pool-size>20</min-pool-size>

<max-pool-size>200</max-pool-size>

</xa-pool>

<security>

<user-name>username</user-name>

<password>password</password>

</security>

</xa-datasource>

<xa-datasource jta="true" jndi-name="java:/jdbc/acquire" pool-name="AcquireDS" enabled="true" use-java-context="true">

<xa-datasource-property name="URL">

database URL

</xa-datasource-property>

<driver>oracle</driver>

<xa-pool>

<min-pool-size>20</min-pool-size>

<max-pool-size>200</max-pool-size>

</xa-pool>

<security>

<user-name>username</user-name>

<password>password</password>

</security>

</xa-datasource>

5. Add the following to the <drivers> element of standalone-full.xml:

<driver name="oracle" module="com.oracle.db">

<xa-datasource-class>oracle.jdbc.xa.client.OracleXADataSource</xa-datasource-class>

</driver>

6. Replace the <subsystem xmlns="urn:jboss:domain:messaging:1.1"> section of standalone-full.xml with the following:

<subsystem xmlns="urn:jboss:domain:messaging:1.1">

<hornetq-server>

<persistence-enabled>true</persistence-enabled>

<journal-file-size>102400</journal-file-size>

<journal-min-files>2</journal-min-files>

<connectors>

<netty-connector name="netty" socket-binding="messaging"/>

<netty-connector name="netty-throughput" socket-binding="messaging-throughput">

<param key="batch-delay" value="50"/>

</netty-connector>

<in-vm-connector name="in-vm" server-id="0"/>

</connectors>

<acceptors>

<netty-acceptor name="netty" socket-binding="messaging"/>

<netty-acceptor name="netty-throughput" socket-binding="messaging-throughput">

<param key="batch-delay" value="50"/>

<param key="direct-deliver" value="false"/>

</netty-acceptor>

<in-vm-acceptor name="in-vm" server-id="0"/>

</acceptors>

<security-settings>

<security-setting match="#">

<permission type="send" roles="guest"/>

<permission type="consume" roles="guest"/>

<permission type="createDurableQueue" roles="guest"/>

<permission type="deleteDurableQueue" roles="guest"/>

<permission type="createNonDurableQueue" roles="guest"/>

<permission type="deleteNonDurableQueue" roles="guest"/>

</security-setting>

</security-settings>

<address-settings>

<address-setting match="#">

<dead-letter-address>jms.queue.deadLetterQueue</dead-letter-address>

<expiry-address>jms.queue.ExpiryQueue</expiry-address>

<redelivery-delay>5000</redelivery-delay>

<max-delivery-attempts>3</max-delivery-attempts>

<max-size-bytes>10485760</max-size-bytes>

<page-size-bytes>1048576</page-size-bytes>

<address-full-policy>PAGE</address-full-policy>

<message-counter-history-day-limit>10</message-counter-history-day-limit>

</address-setting>

</address-settings>

<jms-connection-factories>

<connection-factory name="InVmConnectionFactory">

<connectors>

<connector-ref connector-name="in-vm"/>

</connectors>

<entries>

<entry name="java:/jms/JmsConnectionFactory"/>

</entries>

</connection-factory>

<connection-factory name="RemoteConnectionFactory">

<connectors>

<connector-ref connector-name="netty"/>

</connectors>

<entries>

<entry name="RemoteConnectionFactory"/>

<entry name="java:jboss/exported/jms/RemoteConnectionFactory"/>

</entries>

</connection-factory>

<pooled-connection-factory name="hornetq-ra">

<transaction mode="xa"/>

<connectors>

<connector-ref connector-name="in-vm"/>

</connectors>

<entries>

<entry name="java:/JmsXA"/>

</entries>

<client-id>acquire</client-id>

</pooled-connection-factory>

</jms-connection-factories>

<jms-destinations>

<jms-queue name="testQueue">

<entry name="queue/test"/>

</jms-queue>

<jms-queue name="deadLetterQueue">

<entry name="java:/deadLetterQueue"/>

<durable>true</durable>

</jms-queue>

<jms-queue name="scoreboardQueue">

<entry name="queue/scoreboard"/>

<durable>true</durable>

</jms-queue>

<jms-queue name="annotationUpdateQueue">

<entry name="queue/annotationUpdate"/>

<durable>true</durable>

</jms-queue>

<jms-queue name="naLabReportQueue">

<entry name="queue/naLabReport"/>

<durable>false</durable>

</jms-queue>

<jms-topic name="testTopic">

<entry name="topic/test"/>

</jms-topic>

<jms-topic name="newSpecimenTopic">

<entry name="topic/newSpecimen"/>

</jms-topic>

<jms-topic name="newParticipantTopic">

<entry name="topic/newParticipant"/>

</jms-topic>

<jms-topic name="updateParticipantTopic">

<entry name="topic/updateParticipant"/>

</jms-topic>

<jms-topic name="userChangeTopic">

<entry name="topic/userChange"/>

</jms-topic>

<jms-topic name="newUserTopic">

<entry name="topic/newUser"/>

</jms-topic>

<jms-topic name="newSiteTopic">

<entry name="topic/newSite"/>

</jms-topic>

<jms-topic name="updateConsentTopic">

<entry name="topic/updateConsent"/>

</jms-topic>

<jms-topic name="updateSpecimenTopic">

<entry name="topic/updateSpecimen"/>

</jms-topic>

<jms-topic name="dynamicExtensionTopic">

<entry name="topic/dynamicExtension"/>

</jms-topic>

</jms-destinations>

</hornetq-server>

</subsystem>

* 1. Source Code Changes Required Prior to Acquire Deployment

Some additional configurations are required to the source code to customize the installation for your organization. The following changes are recommended post deployment:

In acquire-utility/src/main/resources/META-INF/beans.xml, update the MailConfig to include the address of your mail server.

In admin-ejb/src/main/resources/templates/accountRequestEmailTemplate, update the email address information.

In inventory-ejb/src/main/resources/templates/applicationSubmittedTemplate, update the email address information.

In inventory-ejb/src/main/resources/templates/postSubmitSaveTemplate, update the email address information.

1. **Deploying Acquire:**
   1. **Checkout and Deploy…..**

* 1. **Installing Templates for Shipment Forms (see Shipment Form Template)**

Acquire’s Shipment Form module allows users to add aliquots of specimen to a shipping form which can be included in the specimen shipment if desired. Adding specimen to the shipment form also deducts material shipped from the inventory. The form is created by searching for specimen by the Medical Record Number or Specimen Label. The form is then generated on the screen and in an export Excel format. The Excel file format adheres to the TissueIntake.xls template located in the JBOSS deployment data directory post deployment.

The Shipment Form contains two sections: solids and fluids. The first section is for tissue specimens data. This section allows a maximum of 95 rows. The second section is for body fluid specimen (e.g. blood) data and has no row limitations. The exported Excel file is named HGSCtissue-intake\_v1.05\_yyyy-MM-dd.xls, where yyyy-MM-dd is replaced by the date the file was exported.

* 1. **Using the Nucleic Acid Loader**

The nucleic acid annotation data is loaded into acquire periodically by the the NA Loader component of Acquire. The NA Loader component processes the NA loader Template and inserts the data into the Acquire database. The loader is configured to look for a new file on the Acquire VM (server) to process every 20 minutes from a specific location available to the server (in the case of Jboss AS, the jboss data directory) in the folder named uploadfiles.

Ex: /usr/local/jboss/standalone/data/uploadfiles

The load frequency may be changed either in the source code in the *NaLabDataUploadTimerBean* or via the standard EJB timer service configuration.

The load file must follow the NA Loader Template format found in the supplemental materials directory of the Acquire source code and be named Lims-Acquire.xls (2010 Excel format or earlier). The loader will validate the data in the template based on uniqueness of new derivative labels, and the identification of existing parent specimen in the database. A report of the outcome of the processing is emailed to a specified email address. This email address can be configured in beans.xml file in the Acquire source code. Rows which fail to load can be edited in the template and reprocessed.

Once a file has been processed by the loader, the loader renames the file with a new file extension. EX: Lims-Acquire.xls.processed.dd-MM-yyyy-HH-mm