# WCT Core Dependency Upgrade

## Introduction

This document outlines the changes required to migrate the wct-core Maven dependencies from their current state to the latest versions.

This document covers only the wct-core module and not the harvest agent or das/store modules.

The table below outlines the wct-core Maven dependencies that were removed:

|  |  |  |
| --- | --- | --- |
| Dependency Artifact ID | Version | Replaced With |
| acegi-security | 1.0.1 | Spring Security 5.0.6 |
| axis | 1.4 | Spring WS 3.0.1 |
| commons-discovery | 0.2 | Not Required |
| commons-httpclient | 3.1 | Apache HTTP Client 4.5.6 |
| commons-lang | 2.3 | Commons lang3 3.7 |
| droolsjbpm-knowledge | 5.4.0.Final | Superseded by 7.8 |
| hibernate | 3.1.3 | Hibernate 5.3.2 |
| jaxrpc-api | 1.1 | Spring WS 3.0.1 |
| jta | 1.1 | Not Required |
| log4j | 1.2.15 | Log4j2 2.11.0 |
| mail | 1.4 | Not Required |
| org.eclipse.jdt.core | 3.7.3 | Not Required |
| saaj-api | 1.2 | Not Required |
| slf4j-log4j12 | 1.7.5 | Not Required |
| spring-dao | 1.2.7 | No longer exists |
| spring-hibernate | 1.2.7 | No longer exists |
| spring-mock | 1.2.7 | Spring Mock MVC 5.0.7 |
| spring-remoting | 1.2.7 | No longer exists |
| spring-support | 1.2.7 | No longer exists |
| struts | 1.2.8 | Not Required |

The table below outlines the new wct-core Maven dependencies that were added:

|  |  |
| --- | --- |
| Dependency Artifact ID | Version |
| httpclient | 4.5.6 |
| commons-lang3 | 3.7 |
| log4j-core | 2.11.0 |
| log4j-api | 2.11.0 |
| log4j-slf4j-impl | 2.11.0 |
| log4j-web | 2.11.0 |
| log4j-jcl | 2.11.0 |
| tiles-core | 3.0.8 |
| tiles-api | 3.0.8 |
| tiles-jsp | 3.0.8 |
| tiles-servlet | 3.0.8 |
| tiles-el | 3.0.8 |
| tiles-extras | 3.0.8 |
| tiles-template | 3.0.8 |
| tiles-request-api | 1.0.7 |
| tiles- request-servlet | 1.0.7 |
| tiles- request-jsp | 1.0.7 |
| hibernate-core | 5.3.2.Final |
| hibernate-c3p0 | 5.3.2.Final |
| c3p0 | 0.9.5.2 |
| spring-tx | 5.0.7.RELEASE |
| spring-security-core | 5.0.6.RELEASE |
| spring-security-web | 5.0.6.RELEASE |
| spring-security-config | 5.0.6.RELEASE |
| spring-security-ldap | 5.0.6.RELEASE |
| spring-security-taglibs | 5.0.6.RELEASE |
| spring-ldap-core | 2.3.2.RELEASE |
| spring-ws-core | 3.0.1.RELEASE |
| spring-ws-security | 3.0.1.RELEASE |
| spring-ws-support | 3.0.1.RELEASE |
| spring-ws-test | 3.0.1.RELEASE |
| spring-xml | 3.0.1.RELEASE |
| spring-oxm | 3.0.1.RELEASE |

The table below outlines the wct-core Maven dependencies that were directly upgraded:

|  |  |  |
| --- | --- | --- |
| Dependency Artifact ID | Current Version | Upgraded Version |
| commons-cli | 1.0 | 1.4 |
| commons-codec | 1.3 | 1.11 |
| commons-collections | 3.1 | 3.2.2 |
| commons-fileupload | 1.1.1 | 1.3.3 |
| commons-io | 1.3.1 | 2.6 |
| commons-logging | 1.1.1 | 1.2 |
| commons-net | 1.4.1 | 3.6 |
| drools-compiler | 5.3.0.Final | 7.8.0.Final |
| drools-core | 5.3.0.Final | 7.8.0.Final |
| ehcache | 1.1 | 3.5.2 |
| javassist | 3.6.0.GA | 3.22.0-GA |
| jsp-api | 2.0 | 2.3.1 |
| jstl | 1.1.2 | 1.2 |
| jzlib | 1.0.7 | 1.1.3 |
| libidn | 0.6.5 | 1.15 |
| mail-api | 1.4.2 | 1.6.1 |
| mvel2 | 2.1.0.drools4 | 2.4.0.Final |
| mysql-connector-java | 5.1.6 | 8.0.11 |
| poi | 2.5.1-final-20040804 | 3.17 |
| poi- scratchpad | 2.5.1-final-20040804 | 3.17 |
| quartz | 1.5.2 | 2.3.0 |
| servlet-api | 2.4 | 3.1.0 |
| spring-beans | 1.2.7 | 5.0.7.RELEASE |
| spring-context | 1.2.7 | 5.0.7.RELEASE |
| spring-core | 1.2.7 | 5.0.7.RELEASE |
| spring-jdbc | 1.2.7 | 5.0.7.RELEASE |
| spring-orm | 1.2.7 | 5.0.7.RELEASE |
| spring-web | 1.2.7 | 5.0.7.RELEASE |
| spring-webmvc | 1.2.7 | 5.0.7.RELEASE |

The following Maven plugins where either added, upgraded or removed:

|  |  |  |
| --- | --- | --- |
| Plugin Artifact ID | Current Version | Upgraded Version |
| maven-compiler-plugin | 2.5.1 | 3.7.0 |
| xdoclet-maven-plugin | 1.0 | - |
| maven-resources-plugin | 2.6 | 3.1.0 |
| maven-antrun-plugin | 1.7 | - |
| maven-war-plugin | 2.3 | 3.2.2 |
| jetty-maven-plugin | 7.5.4.v20111024 | 9.4.11.v20180605 |
| maven-jaxb2-plugin | - | 0.14.0 |
| jaxb2-maven-plugin | 1.6 | 2.1 |

Note that there are other Maven dependencies which could also be upgraded.

## Methodology

### Original Code

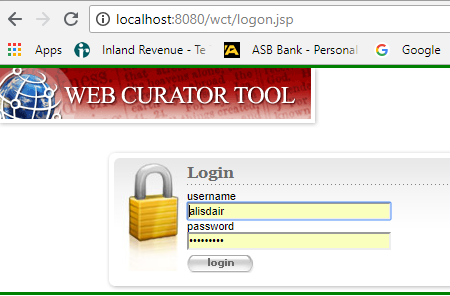
A version of the wct-core module was copied and all extraneous code (including all tests) was removed except for the login page, a cut-down home page with a link to a test page which loads a test table using Hibernate.

This code can be found in the master branch of the webcurator-upgrade-poc git hub repository: <https://github.com/ahamblyn/webcurator-upgrade-poc>

### Upgraded Code

The code using the upgraded dependencies can be found in the Upgrade\_Spring\_Hibernate branch of the webcurator-upgrade-poc git hub repository: <https://github.com/ahamblyn/webcurator-upgrade-poc>

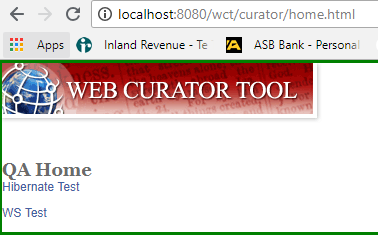
#### Login



#### Home Page

The application consists of a home page with two links:

* Hibernate Test
* WS Test

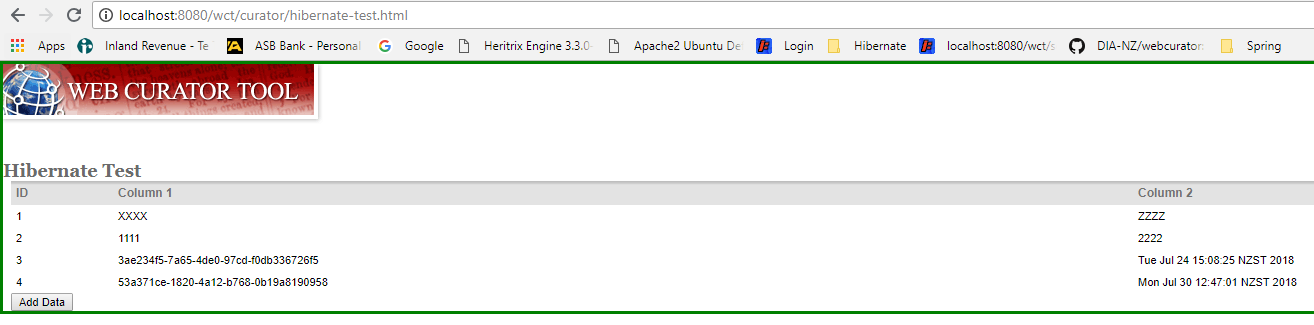


#### Hibernate Test Page

The following SQL needs to be run in the WCT database before deploying the cut-down application:

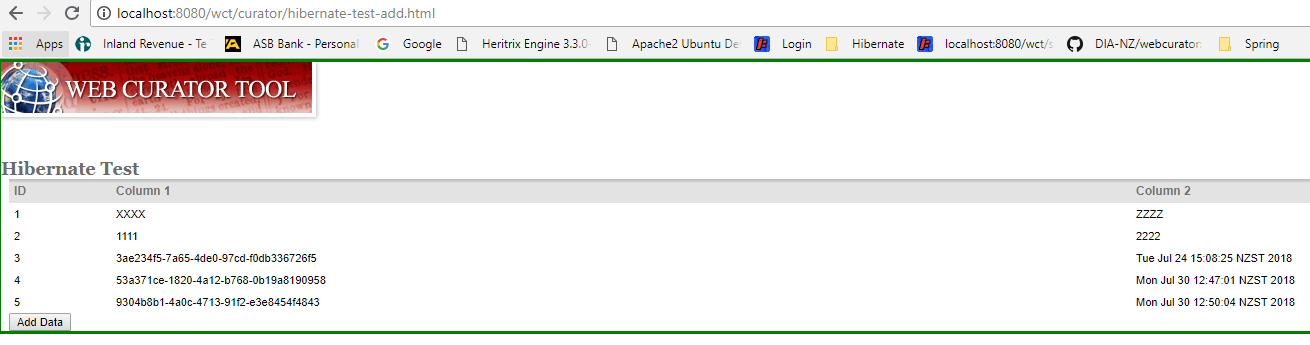
create table HIBERNATE\_TEST(ID int NOT NULL, COLUMN1 varchar(100), COLUMN2 varchar(100), primary key(ID));

insert into HIBERNATE\_TEST values(1, 'XXXX', 'XXXX');



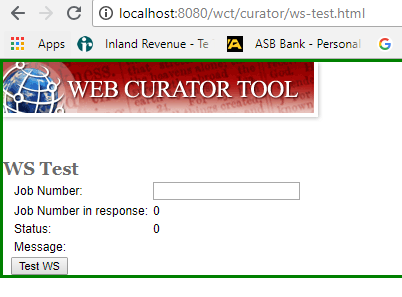
This page displays the contents of the HIBERNATE\_TEST table.

Clicking the “Add Data” button will add a test row to this table.

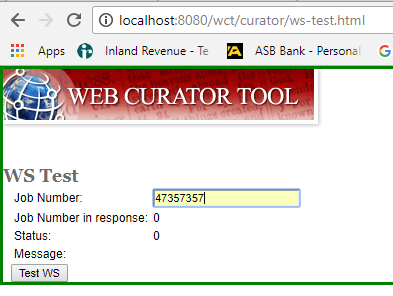


#### WS Test Page

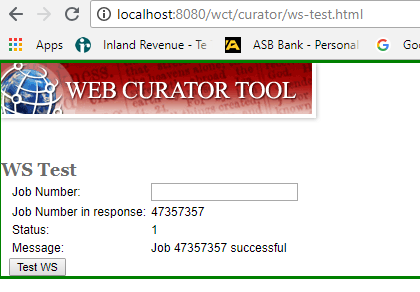
A SOAP web service and corresponding client have been created using Spring WS.



This page displays a form which sends a SOAP request with a job number via the client to the web service which echoes job number back along with a status and a message.



Clicking the “Test WS” button.



## Hibernate Configuration

### Database Change

As part of the upgrade from Hibernate 3.1.3 to 5.3.2, the following SQL needs to be run to increase the size of the IG\_VALUE column of the ID\_GENERATOR table.

rename table id\_generator to id\_generator\_tmp;

create table ID\_GENERATOR ( IG\_TYPE varchar(255), IG\_VALUE bigint(20) ) ;

insert into id\_generator

select \* from id\_generator\_tmp;

drop table id\_generator\_tmp;

### H2Dialect

The H2Dialect class needs modifying to account for the changes in Hibernate from versions 3.1.3 to 5.3.2.

Mostly the type instances required to register the functions have changed and the name of the method implemented in the static EXTRACTOR private class has changed. E.g.:

registerFunction(**"acos"**, **new** StandardSQLFunction(**"acos"**, DoubleType.***INSTANCE***));

**protected** String doExtractConstraintName(SQLException sqle)

### Hibernate Properties

A new hibernate.properties file was created to hold the Hibernate properties:

**datasource.jndi.name**=**java:comp/env/jdbc/wctDatasource***# Hibernate properties***hibernate.show\_sql**=**false  
hibernate.hbm2ddl.auto**=**validate  
hibernate.id.new\_generator\_mappings**=**true***# hibernate.dialect has been defined in the wct-core.properties file so use a different name***hibernate-core.dialect**=**${hibernate.dialect}***#C3P0 properties***hibernate.c3p0.min\_size**=**5  
hibernate.c3p0.max\_size**=**20  
hibernate.c3p0.acquire\_increment**=**1  
hibernate.c3p0.timeout**=**1800  
hibernate.c3p0.max\_statements**=**150**

Only the dialect is templated from other property files but it could be extended to others.

The hibernate.properties file is used in the Spring configuration to set up the datasource and the hibernate session factory.

### Hibernate Spring Configuration

The Hibernate Spring configuration was moved to a separate data-config.xml file from the wct-core.xml file:

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd"**>  
  
 <**context:property-placeholder location="classpath:hibernate.properties"** />  
  
 <**bean id="dataSource" class="org.springframework.jndi.JndiObjectFactoryBean"**>  
 <**property name="jndiName"**>  
 <**value**>${datasource.jndi.name}</**value**>  
 </**property**>  
 </**bean**>  
  
 <**bean class="org.springframework.orm.hibernate5.LocalSessionFactoryBean" id="sessionFactory"**>  
 <**property name="dataSource" ref="dataSource"**></**property**>  
 <**property name="annotatedClasses"**>  
 <**list**>  
 <**value**>org.webcurator.domain.model.auth.Agency</**value**>  
 <**value**>org.webcurator.domain.model.auth.Role</**value**>  
 <**value**>org.webcurator.domain.model.auth.RolePrivilege</**value**>  
 <**value**>org.webcurator.domain.model.auth.User</**value**>  
 <**value**>org.webcurator.domain.model.core.HibernateTest</**value**>  
 <**value**>org.webcurator.domain.model.core.AuthorisingAgent</**value**>  
 <**value**>org.webcurator.domain.model.core.PermissionExclusion</**value**>  
 <**value**>org.webcurator.domain.model.core.Permission</**value**>  
 <**value**>org.webcurator.domain.model.core.UrlPattern</**value**>  
 <**value**>org.webcurator.domain.model.core.Site</**value**>  
 <**value**>org.webcurator.core.permissionmapping.Mapping</**value**>  
 <**value**>org.webcurator.core.permissionmapping.MappingView</**value**>  
 </**list**>  
 </**property**>  
 <**property name="hibernateProperties"**>  
 <**props**>  
 <**prop key="hibernate.dialect"**>${hibernate-core.dialect}</**prop**>  
 <**prop key="hibernate.show\_sql"**>${hibernate.show\_sql}</**prop**>  
 <**prop key="hibernate.hbm2ddl.auto"**>${hibernate.hbm2ddl.auto}</**prop**>  
 <**prop key="hibernate.id.new\_generator\_mappings"**>${hibernate.id.new\_generator\_mappings}</**prop**>  
 <**prop key="hibernate.c3p0.min\_size"**>${hibernate.c3p0.min\_size}</**prop**>  
 <**prop key="hibernate.c3p0.max\_size"**>${hibernate.c3p0.max\_size}</**prop**>  
 <**prop key="hibernate.c3p0.acquire\_increment"**>${hibernate.c3p0.acquire\_increment}</**prop**>  
 <**prop key="hibernate.c3p0.timeout"**>${hibernate.c3p0.timeout}</**prop**>  
 <**prop key="hibernate.c3p0.max\_statements"**>${hibernate.c3p0.max\_statements}</**prop**>  
 </**props**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="transactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager"**>  
 <**property name="sessionFactory" ref="sessionFactory"**/>  
 </**bean**>  
  
 <**tx:annotation-driven transaction-manager="transactionManager"** />  
  
 <**bean id="transactionTemplate" class="org.springframework.transaction.support.TransactionTemplate"**>  
 <**constructor-arg**>  
 <**ref bean="transactionManager"**/>  
 </**constructor-arg**>  
 </**bean**>  
</**beans**>

The xml file loads the hibernate.properties file access its properties.

### web.xml

The data-config.xml file is added to the context config location:

<**context-param**>  
 <**param-name**>contextConfigLocation</**param-name**>  
 <**param-value**>/WEB-INF/classes/data-config.xml, /WEB-INF/classes/wct-core.xml, /WEB-INF/classes/wct-core-security.xml</**param-value**>  
</**context-param**>

The Hibernate filter class is updated to Hibernate 5:

<**filter**>  
 <**filter-name**>ws-osivFilter</**filter-name**>  
 <**filter-class**>org.springframework.orm.hibernate5.support.OpenSessionInViewFilter</**filter-class**>  
</**filter**>

### Jetty Datasource

For developer testing using Jetty, the Jetty datasource configured in the jetty-env.xml file was modified to use a dbcp2 basic datasource:

*<?***xml version="1.0" encoding="UTF-8"***?>***<!DOCTYPE Configure PUBLIC "-//Mort Bay Consulting//DTD Configure//EN" "http://jetty.mortbay.org/configure.dtd"*>***<**Configure class="org.eclipse.jetty.webapp.WebAppContext"**>  
 <**New id="wctDatasource" class="org.eclipse.jetty.plus.jndi.Resource"**>  
 <**Arg**>java:comp/env/jdbc/wctDatasource</**Arg**>  
 <**Arg**>  
 <**New class="org.apache.commons.dbcp2.BasicDataSource"**>  
 <**Set name="driverClassName"**>${schema.driver}</**Set**>  
 <**Set name="username"**>${schema.user}</**Set**>  
 <**Set name="password"**>${schema.password}</**Set**>  
 <**Set name="url"**>${schema.url}</**Set**>  
 <**Set name="testOnBorrow"**>true</**Set**>  
 <**Set name="validationQuery"**>${schema.query}</**Set**>  
 </**New**>  
 </**Arg**>  
 </**New**>  
</**Configure**>

The dbcp2 dependency was added to the jetty-maven-plugin.

### Entity Configuration

As Hibernate 5 no longer uses generated hbm.xml files containing the class, id, and property information, these have to be added to the entity classes using annotations.

The named queries in each entity class also have to be converted to annotations.

#### Mapping.java

@Entity  
@Table(name = **"URL\_PERMISSION\_MAPPING"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.core.permissionmapping.Mapping.LIST"**, query = **"from Mapping where domain=?1"**),  
 @NamedQuery(name = **"org.webcurator.core.permissionmapping.Mapping.FETCH"**, query = **"from Mapping where oid=?1"**),  
 @NamedQuery(name = **"org.webcurator.core.permissionmapping.Mapping.DELETE"**, query = **"delete from Mapping where urlPattern.oid = :urlPatternOid and permission.oid = :permissionOid"**),  
 @NamedQuery(name = **"org.webcurator.core.permissionmapping.Mapping.DELETE\_BY\_SITE"**, query = **"delete from Mapping where permission.site.oid = :siteOid"**)})  
**public class** Mapping {

…

*/\*\* The UrlPattern \*/*@ManyToOne(targetEntity = UrlPattern.**class**)  
@JoinColumn(name = **"UPM\_URL\_PATTERN\_ID"**, foreignKey = @ForeignKey(name = **"FK\_UPM\_URL\_PATTERN\_ID"**))  
**private** UrlPattern **urlPattern**;  
*/\*\* The Permission \*/*@ManyToOne(targetEntity = Permission.**class**)  
@JoinColumn(name = **"UPM\_PERMISSION\_ID"**, foreignKey = @ForeignKey(name = **"FK\_UPM\_PERMISSION\_ID"**))  
**private** Permission **permission**;  
*/\*\* The calculate base domain \*/*@Column(name = **"UPM\_DOMAIN"**, length = 1024, columnDefinition = **"text"**)  
**private** String **domain**;  
*/\*\* The Oid \*/*@Id  
@Column(name = **"UPM\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"mappingGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"mappingGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"General"**)  
**private** Long **oid**;

…

#### MappingView.java

@Entity  
@Table(name = **"URL\_PERMISSION\_MAPPING\_VIEW"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.core.permissionmapping.MappingView.LIST"**, query = **"from MappingView where domain=?1"**)})  
**public class** MappingView {

…

*/\*\* The oid of the mapping view record. \*/*@Id  
@Column(name = **"UPM\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"mappingViewGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"mappingViewGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"General"**)  
**private** Long **oid**;  
  
*/\*\* The UrlPattern \*/*@Column(name = **"UP\_PATTERN"**, length = 2048, columnDefinition = **"text"**)  
**private** String **urlPattern**;  
*/\*\* The Permission end date\*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_END\_DATE"**)  
**private** Date **endDate**;  
*/\*\* The calculated base domain \*/*@Column(name = **"UPM\_DOMAIN"**, length = 1024, columnDefinition = **"text"**)  
**private** String **domain**;  
*/\*\* The database id of the owning agency. \*/*@Column(name = **"PE\_OWNING\_AGENCY\_ID"**)  
**private** Long **owningAgencyId**;  
*/\*\* The permission id of the permission. \*/*@Column(name = **"PE\_OID"**)  
**private** Long **permissionOId**;  
 */\*\* Site is active flag. \*/*@Column(name = **"ST\_ACTIVE"**, nullable = **false**)  
 **private boolean siteActive** = **true**;

…

#### Agency.java

@Entity  
@Table(name = **"AGENCY"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.domain.model.auth.Agency.getAllAgencies"**, query = **"from Agency agc order by agc.name"**)})  
**public class** Agency **implements** Serializable {

…

*/\*\* The database OID of the Agency \*/* @Id  
 @Column(name = **"AGC\_OID"**, nullable = **false**)  
 @GeneratedValue(generator = **"agencyGen"**, strategy = GenerationType.***TABLE***)  
 @TableGenerator(name = **"agencyGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"Agency"**)  
 **private** Long **oid**;  
 */\*\* The name of the agency \*/* @Column(name = **"AGC\_NAME"**, length = 80, unique = **true**, nullable = **false**)  
 **private** String **name**;  
 */\*\* The address of the agency \*/* @Column(name = **"AGC\_ADDRESS"**, length = 255, nullable = **false**)  
 **private** String **address**;  
 */\*\* The phone number for the agency \*/* @Column(name = **"AGC\_PHONE"**, length = 20, nullable = **true**)  
 **private** String **phone**;  
 */\*\* The URL for the agency - such as their home page \*/* @Column(name = **"AGC\_URL"**, length = 255, nullable = **true**)  
 **private** String **agencyURL**;  
 */\*\* The URL to a logo that can be used in the Permission Request templates \*/* @Column(name = **"AGC\_LOGO\_URL"**, length = 255, nullable = **true**)  
 **private** String **agencyLogoURL**;  
 */\*\* The email to use to contact the agency \*/* @Column(name = **"AGC\_EMAIL"**, length = 80, nullable = **true**)  
 **private** String **email**;  
 */\*\* The fax number to use to contact the agency \*/* @Column(name = **"AGC\_FAX"**, length = 20, nullable = **true**)  
 **private** String **fax**;  
 */\*\* The set of users in this agency \*/* @OneToMany(cascade = CascadeType.***ALL***, orphanRemoval = **true**, mappedBy = **"agency"**, targetEntity = User.**class**)  
 **private** Set<User> **users**;  
 */\*\* The set of roles in this agency \*/* @OneToMany(cascade = CascadeType.***ALL***, mappedBy = **"agency"**, targetEntity = Role.**class**)  
 **private** Set<Role> **roles**;  
 */\*\* Flag for displaying tasks on the intray screen \*/* @Column(name = **"AGC\_SHOW\_TASKS"**)  
 **private boolean showTasks** = **false**;  
 */\*\* Default description used for creating targets \*/* @Column(name = **"AGC\_DEFAULT\_DESC\_TYPE"**)  
**private** String **defaultDescriptionType**;

…

#### Role.java

@Entity  
@Table(name = **"WCTROLE"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.domain.model.auth.Role.getRoles"**, query = **"FROM Role rol order by rol.agency.name, rol.name"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.Role.getAssociatedRolesByUser"**, query = **"SELECT rol FROM Role rol, User usr JOIN usr.roles usr\_roles WHERE usr\_roles.oid = rol.oid AND usr.oid=?1 order by rol.name"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.Role.getRolesByAgency"**, query = **"SELECT rol FROM Role rol WHERE rol.agency.oid = ?1"**)})  
**public class** Role **implements** AgencyOwnable, Serializable{

…

*/\*\* The database OID of the Role \*/*@Id  
@Column(name = **"ROL\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"roleGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"roleGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"Role"**)  
 **private** Long **oid**;  
*/\*\* The name of the role \*/*@Column(name = **"ROL\_NAME"**, length = 80, nullable = **false**)  
**private** String **name**;  
*/\*\* A descrption for the role \*/*@Column(name = **"ROL\_DESCRIPTION"**, length = 255, nullable = **true**)  
**private** String **description**;  
*/\*\* The set of Users that hold this role \*/*@ManyToMany(targetEntity = User.**class**)  
@Cascade({org.hibernate.annotations.CascadeType.***SAVE\_UPDATE***})  
@JoinTable(name = **"USER\_ROLE"**, joinColumns = {@JoinColumn(name = **"URO\_ROL\_OID"**)}, foreignKey = @ForeignKey(name = **"FK\_USERROLE\_TO\_ROLE"**),  
 inverseJoinColumns = {@JoinColumn(name = **"URO\_USR\_OID"**)}, inverseForeignKey = @ForeignKey(name = **"FK\_USERROLE\_TO\_USER"**))  
**private** Set<User> **users**;  
*/\*\* The set of privileges that this role is made up from. \*/*@OneToMany(cascade = CascadeType.***ALL***, orphanRemoval = **true**, mappedBy = **"role"**, targetEntity = RolePrivilege.**class**)  
**private** Set<RolePrivilege> **rolePrivileges**;  
*/\*\* The agency that this role belongs to \*/*@ManyToOne(targetEntity = Agency.**class**)  
@JoinColumn(name = **"ROL\_AGENCY\_OID"**, nullable = **false**, foreignKey = @ForeignKey(name = **"FK\_ROLE\_AGENCY\_OID"**))  
**private** Agency **agency**;

…

#### RolePrivilege.java

@Entity  
@Table(name = **"ROLE\_PRIVILEGE"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.domain.model.auth.RolePrivilege.getUserPrivileges"**, query = **"SELECT distinct rolpriv FROM RolePrivilege rolpriv JOIN rolpriv.role.users rp\_r\_users WHERE rp\_r\_users.username=?1 "**)})  
**public class** RolePrivilege **implements** Serializable {

…

*/\*\* The database OID for the Role Privilege. \*/*@Id  
@Column(name = **"PRV\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"rolePrivGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"rolePrivGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"RolePriv"**)  
**private** Long **oid**;  
*/\*\* The identifier of the privilege \*/*@Column(name = **"PRV\_CODE"**, length = 40, nullable = **false**)  
**private** String **privilege**;  
*/\*\* The scope of the privilege - i.e. how widely the privilege applies \*/*@Column(name = **"PRV\_SCOPE"**, nullable = **false**)  
**private int privilegeScope**;  
*/\*\* The role that this privilege belongs to. \*/*@ManyToOne(targetEntity = Role.**class**)  
@JoinColumn(name = **"PRV\_ROLE\_OID"**, foreignKey = @ForeignKey(name = **"FK\_PRIV\_ROLE\_OID"**))  
**private** Role **role**;

…

#### User.java

@Entity  
@Table(name = **"WCTUSER"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.domain.model.auth.User.getUserByName"**, query = **"SELECT usr FROM User usr WHERE usr.username=?1 "**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.User.getUsersByAgency"**, query = **"SELECT usr FROM User usr WHERE usr.agency.oid=?1 ORDER BY usr.username"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.User.getAllUserDTOs"**, query = **"SELECT new org.webcurator.domain.model.dto.UserDTO(usr.oid, usr.username, usr.email, usr.notificationsByEmail, usr.tasksByEmail, usr.title, usr.firstname, usr.lastname, usr.phone, usr.address, usr.active, usr.agency.name, usr.notifyOnHarvestWarnings, usr.notifyOnGeneral) FROM User usr ORDER BY usr.agency.name, usr.username"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.User.getAllUserDTOsByAgency"**, query = **"SELECT new org.webcurator.domain.model.dto.UserDTO(usr.oid, usr.username, usr.email, usr.notificationsByEmail, usr.tasksByEmail, usr.title, usr.firstname, usr.lastname, usr.phone, usr.address, usr.active, usr.agency.name, usr.notifyOnHarvestWarnings, usr.notifyOnGeneral) FROM User usr WHERE usr.agency.oid=?1 ORDER BY usr.username"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.User.getUserDTOsByPrivilege"**, query = **"SELECT DISTINCT new org.webcurator.domain.model.dto.UserDTO(usr.oid, usr.username, usr.email, usr.notificationsByEmail, usr.tasksByEmail, usr.title, usr.firstname, usr.lastname, usr.phone, usr.address, usr.active, usr.agency.name, usr.notifyOnHarvestWarnings, usr.notifyOnGeneral) FROM Role rol, User usr JOIN usr.roles usr\_roles JOIN rol.rolePrivileges rol\_priv WHERE usr\_roles.oid = rol.oid AND rol\_priv.privilege=?1"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.User.getUserDTOByOid"**, query = **"SELECT new org.webcurator.domain.model.dto.UserDTO(usr.oid, usr.username, usr.email, usr.notificationsByEmail, usr.tasksByEmail, usr.title, usr.firstname, usr.lastname, usr.phone, usr.address, usr.active, usr.agency.name, usr.notifyOnHarvestWarnings, usr.notifyOnGeneral) FROM User usr WHERE usr.oid=?1"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.auth.User.getUserDTOsByPrivAgency"**, query = **"SELECT DISTINCT new org.webcurator.domain.model.dto.UserDTO(usr.oid, usr.username, usr.email, usr.notificationsByEmail, usr.tasksByEmail, usr.title, usr.firstname, usr.lastname, usr.phone, usr.address, usr.active, usr.agency.name, usr.notifyOnHarvestWarnings, usr.notifyOnGeneral) FROM Role rol, User usr JOIN usr.roles usr\_roles JOIN rol.rolePrivileges rol\_priv WHERE usr\_roles.oid = rol.oid AND rol\_priv.privilege=?1 AND usr.agency.oid=?2"**)})  
**public class** User **implements** Serializable {

…

*/\*\* The database OID of the User object \*/*@Id  
@Column(name = **"USR\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"userGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"userGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"User"**)  
**private** Long **oid**;  
*/\*\* The login username \*/*@Column(name = **"USR\_USERNAME"**, unique = **true**, length = 80, nullable = **false**)  
**private** String **username**;  
*/\*\* The contact email address \*/*@Column(name = **"USR\_EMAIL"**, length = 100, nullable = **false**)  
**private** String **email**;  
*/\*\* True to enable notifications to be sent by e-mail as well as to the intray \*/*@Column(name = **"USR\_NOTIFICATIONS\_BY\_EMAIL"**, nullable = **false**)  
**private boolean notificationsByEmail**;  
*/\*\* True to enable tasks to be sent by e-mail as well as to the intray \*/*@Column(name = **"USR\_TASKS\_BY\_EMAIL"**, nullable = **false**)  
**private boolean tasksByEmail**;  
*/\*\* The title of the user: Mr., Mrs., etc \*/*@Column(name = **"USR\_TITLE"**, length = 10, nullable = **true**)  
**private** String **title**;  
*/\*\* The first name of the user \*/*@Column(name = **"USR\_FIRSTNAME"**, length = 50, nullable = **false**)  
**private** String **firstname**;  
*/\*\* The last name of the user \*/*@Column(name = **"USR\_LASTNAME"**, length = 50, nullable = **false**)  
**private** String **lastname**;  
*/\*\* True if the user account is active \*/*@Column(name = **"USR\_ACTIVE"**, nullable = **false**)  
**private boolean active**;  
*/\*\* True if the user must change their password on the next login \*/*@Column(name = **"USR\_FORCE\_PWD\_CHANGE"**, nullable = **false**)  
**private boolean forcePasswordChange**;  
*/\*\* True if the user authentication should use an external authentication source such as LDAP \*/*@Column(name = **"USR\_EXTERNAL\_AUTH"**, nullable = **false**)  
**private boolean externalAuth**;  
*/\*\* The user's password \*/*@Column(name = **"USR\_PASSWORD"**, length = 255, nullable = **true**)  
**private** String **password**;  
*/\*\* The user's phone number \*/*@Column(name = **"USR\_PHONE"**, length = 16, nullable = **true**)  
**private** String **phone**;  
*/\*\* The user's address \*/*@Column(name = **"USR\_ADDRESS"**, length = 200, nullable = **true**)  
**private** String **address**;  
*/\*\* The set of roles the user belongs to \*/*@ManyToMany(targetEntity = Role.**class**)  
@Cascade({CascadeType.***SAVE\_UPDATE***})  
@JoinTable(name = **"USER\_ROLE"**, joinColumns = {@JoinColumn(name = **"URO\_USR\_OID"**)}, foreignKey = @ForeignKey(name = **"FK\_USERROLE\_TO\_USER"**),  
 inverseJoinColumns = {@JoinColumn(name = **"URO\_ROL\_OID"**)}, inverseForeignKey = @ForeignKey(name = **"FK\_USERROLE\_TO\_ROLE"**))  
**private** Set<Role> **roles**;  
*/\*\* The agency the user belongs to \*/*@ManyToOne(targetEntity = Agency.**class**)  
@JoinColumn(name = **"USR\_AGC\_OID"**, nullable = **false**, foreignKey = @ForeignKey(name = **"FK\_USER\_AGENCY\_OID"**))  
**private** Agency **agency**;  
*/\*\* For inactive users, the date the user was deactivated \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"USR\_DEACTIVATE\_DATE"**, nullable = **true**)  
**private** Date **deactivateDate**;  
*/\*\* Enable notifications for changes to objects the user owns \*/*@Column(name = **"USR\_NOTIFY\_ON\_GENERAL"**, nullable = **false**)  
**private boolean notifyOnGeneral** = **true**;  
*/\*\* Enable notifications for harvester warnings. \*/*@Column(name = **"USR\_NOTIFY\_ON\_WARNINGS"**, nullable = **false**)  
**private boolean notifyOnHarvestWarnings** = **false**;

…

#### AuthorisingAgent.java

@Entity  
@Table(name = **"AUTHORISING\_AGENT"**)  
**public class** AuthorisingAgent **extends** AbstractIdentityObject {

…

*/\*\* The database oid \*/*@Id  
@Column(name = **"AA\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"authorisingAgentGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"authorisingAgentGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"General"**)  
**private** Long **oid**;  
*/\*\* The name of the agent. \*/*@Column(name = **"AA\_NAME"**, unique = **true**, length = 255)  
**private** String **name**;  
*/\*\* A description of the agent. \*/*@Column(name = **"AA\_DESC"**, length = 2048, columnDefinition = **"text"**)  
**private** String **description**;  
*/\*\* The name of the contact within the agency. \*/*@Column(name = **"AA\_CONTACT"**, length = 255)  
**private** String **contact**;  
*/\*\* The phone number for the contact. \*/*@Column(name = **"AA\_PHONE\_NUMBER"**, length = 32)  
**private** String **phoneNumber**;  
*/\*\* The e-mail address for the contact. \*/*@Column(name = **"AA\_EMAIL"**, length = 255)  
**private** String **email**;  
*/\*\* The mailing address for the contact. \*/*@Column(name = **"AA\_ADRESS"**, length = 2048, columnDefinition = **"text"**)  
**private** String **address**;

…

#### HibernateTest.java

@Entity  
@Table(name = **"HIBERNATE\_TEST"**)  
@NamedQueries({@NamedQuery(name = **"org.webcurator.domain.model.core.HibernateTest.getAll"**, query = **"SELECT ht FROM HibernateTest ht"**),  
 @NamedQuery(name = **"org.webcurator.domain.model.core.HibernateTest.maxID"**, query = **"SELECT max(ht.id) FROM HibernateTest ht"**)})  
**public class** HibernateTest {

…

@Id  
@Column(name = **"ID"**)  
**private** Integer **id**;  
@Column(name = **"COLUMN1"**, length = 100)  
**private** String **column1**;  
@Column(name = **"COLUMN2"**, length = 100)  
**private** String **column2**;

…

#### Permission.java

@Entity  
@Table(name = **"PERMISSION"**)  
**public class** Permission **extends** AbstractIdentityObject **implements** AgencyOwnable, InTrayResource, AgencyInTrayResource {

…

*/\*\* The database id of the permission. \*/*@Id  
@Column(name = **"PE\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"permissionGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"permissionGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"General"**)  
**private** Long **oid**;  
*/\*\* The agent granting the permission. \*/*@ManyToOne(targetEntity = AuthorisingAgent.**class**)  
@JoinColumn(name = **"PE\_AUTH\_AGENT\_ID"**)  
**private** AuthorisingAgent **authorisingAgent**;  
*/\*\* The set of UrlPatterns covered by this permission. \*/*@ManyToMany(targetEntity = Permission.**class**)  
@Cascade({org.hibernate.annotations.CascadeType.***SAVE\_UPDATE***})  
@JoinTable(name = **"PERMISSION\_URLPATTERN"**, joinColumns = {@JoinColumn(name = **"PU\_PERMISSION\_ID"**)}, foreignKey = @ForeignKey(name = **"PU\_FK\_2"**),  
 inverseJoinColumns = {@JoinColumn(name = **"PU\_URLPATTERN\_ID"**)}, inverseForeignKey = @ForeignKey(name = **"PU\_FK\_1"**))  
**private** Set<UrlPattern> **urls**;  
*/\*\* The date this permission starts. \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_START\_DATE"**)  
**private** Date **startDate**;  
*/\*\* The date this permission ends. \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_END\_DATE"**)  
**private** Date **endDate**;  
*/\*\* Whether this permission is approved. \*/  
//****TODO Check what this means.***@Column(name = **"PE\_APPROVED\_YN"**)  
**private boolean approved**;  
*/\*\* The status of the permission. One of the STATUS\_ constants \*/*@Column(name = **"PE\_STATUS"**)  
**private int status**;  
*/\*\* Any authorising agency acceptance notes attached to the permission. \*/*@Column(name = **"PE\_NOTES"**, length = 4000, columnDefinition = **"text"**)  
**private** String **authResponse**;  
*/\*\* The access status \*/*@Column(name = **"PE\_ACCESS\_STATUS"**, length = 255)  
**private** String **accessStatus**;  
*/\*\* The date at which this permission will be open access \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_OPEN\_ACCESS\_DATE"**)  
**private** Date **openAccessDate**;  
*/\*\* Whether this permission is publicly available \*/*@Column(name = **"PE\_AVAILABLE\_YN"**)  
**private boolean availableFlag**;  
*/\*\* Any special requirements attached to this permission. \*/*@Column(name = **"PE\_SPECIAL\_REQUIREMENTS"**, length = 2048, columnDefinition = **"text"**)  
**private** String **specialRequirements**;  
*/\*\* The creation date of this permission. \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_CREATION\_DATE"**)  
**private** Date **creationDate**;  
*/\*\* The copyright URL to use during the access component. \*/*@Column(name = **"PE\_COPYRIGHT\_URL"**, length = 2048, columnDefinition = **"text"**)  
**private** String **copyrightUrl**;  
*/\*\* The copyright statement to display in the access system. \*/*@Column(name = **"PE\_COPYRIGHT\_STATEMENT"**, length = 2048, columnDefinition = **"text"**)  
**private** String **copyrightStatement**;  
*/\*\* The date that a permission requested was sent to the authorising agent. \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_PERMISSION\_REQUESTED\_DATE"**)  
**private** Date **permissionSentDate**;  
*/\*\* The date that permission was granted/refused. \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"PE\_PERMISSION\_GRANTED\_DATE"**)  
**private** Date **permissionGrantedDate**;  
*/\*\* The site that this permission belongs to. \*/*@ManyToOne(targetEntity = Site.**class**)  
@JoinColumn(name = **"PE\_SITE\_ID"**, nullable = **false**, foreignKey = @ForeignKey(name = **"FK\_PE\_SITE\_ID"**))  
**private** Site **site**;  
*/\*\* Whether the permission is marked as a quick pick. \*/*@Column(name = **"PE\_QUICK\_PICK"**)  
**private boolean quickPick**;  
*/\*\* Quick Pick Display Name \*/*@Column(name = **"PE\_DISPLAY\_NAME"**, length = 32)  
**private** String **displayName**;  
*/\*\* The agency that owns this permission. \*/*@ManyToOne(targetEntity = Agency.**class**)  
@JoinColumn(name = **"PE\_OWNING\_AGENCY\_ID"**)  
**private** Agency **owningAgency**;  
*/\*\* A file reference \*/*@Column(name = **"PE\_FILE\_REFERENCE"**, length = 255)  
**private** String **fileReference**;  
*/\*\* A dirty flag to track if this permission has been updated. This is   
 \* managed by the SiteController, not self-managed. \*/*@Transient  
**private boolean dirty** = **false**;  
*/\*\*   
 \* The initial state of the permission. This is initialised when setStatus   
 \* is called for the first time, which will either be by Hibernate or the  
 \* BusinessObjectFactory.   
 \*/*@Transient  
**private int originalStatus** = -1;  
*/\*\* Flag to determine if a "Seek Permission" task should be created. \*/*@Transient  
**private boolean createSeekPermissionTask** = **false**;  
*/\*\* List of excluded URLs \*/*@OneToMany(cascade = CascadeType.***ALL***, orphanRemoval = **true**, mappedBy = **"permission"**, targetEntity = PermissionExclusion.**class**)  
**private** List<PermissionExclusion> **exclusions** = **new** LinkedList<PermissionExclusion>();

…

#### PermissionExclusion.java

@Entity  
@Table(name = **"PERMISSION\_EXCLUSION"**)  
**public class** PermissionExclusion {

…

*/\*\* The URL excluded \*/*@Column(name = **"PEX\_URL"**, length = 1024, columnDefinition = **"text"**)  
**private** String **url**;  
  
*/\*\* The reason for exclusion \*/*@Column(name = **"PEX\_REASON"**, length = 255)  
**private** String **reason**;  
  
*/\*\* The OID of the object \*/*@Id  
@Column(name = **"PEX\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"permExclusionGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"permExclusionGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"PermExclusion"**)  
**private** Long **oid** = **null**;  
  
@ManyToOne(targetEntity = Permission.**class**)  
@JoinColumn(name = **"PEX\_PERMISSION\_OID"**)  
**private** Permission **permission**;

…

#### Site.java

@Entity  
@Table(name = **"SITE"**)  
**public class** Site **extends** AbstractIdentityObject **implements** AgencyOwnable {

…

*/\*\* The database oid of the site \*/*@Id  
@Column(name = **"ST\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"siteGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"siteGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"General"**)  
**private** Long **oid**;  
*/\*\* The title of the site \*/*@Column(name = **"ST\_TITLE"**, unique = **true**, length = 255, nullable = **false**)  
**private** String **title**;  
*/\*\* A description of the site \*/*@Column(name = **"ST\_DESC"**, length = 4000, columnDefinition = **"text"**)  
**private** String **description**;  
*/\*\* A set of notes about the site. \*/*@Column(name = **"ST\_NOTES"**, length = 4000, columnDefinition = **"text"**)  
**private** String **notes**;  
*/\*\* A library order no. \*/*@Column(name = **"ST\_LIBRARY\_ORDER\_NO"**, length = 32)  
**private** String **libraryOrderNo**;  
*/\*\* Whether the site has been published or not. \*/*@Column(name = **"ST\_PUBLISHED"**, nullable = **false**)  
**private boolean published**;  
*/\*\* Site is active flag. \*/*@Column(name = **"ST\_ACTIVE"**, nullable = **false**)  
**private boolean active** = **true**;  
*/\*\* The date the Site was created \*/*@Temporal(TemporalType.***TIMESTAMP***)  
@Column(name = **"ST\_CREATION\_DATE"**)  
**private** Date **creationDate**;  
*/\*\* The set of authorising agents (those who must provide permission to  
 \* harvest the site.  
 \*/*@ManyToMany(targetEntity = AuthorisingAgent.**class**)  
@JoinTable(name = **"SITE\_AUTH\_AGENCY"**, joinColumns = {@JoinColumn(name = **"SA\_SITE\_ID"**)},  
 inverseJoinColumns = {@JoinColumn(name = **"SA\_AGENT\_ID"**)}, inverseForeignKey = @ForeignKey(name = **"FK\_SA\_AGENT\_ID"**))  
**private** Set<AuthorisingAgent> **authorisingAgents** = **new** HashSet<AuthorisingAgent>();  
*/\*\* A set of URL patterns that are encompassed by this site. \*/*@OneToMany(cascade = CascadeType.***ALL***, orphanRemoval = **true**, mappedBy = **"site"**, targetEntity = UrlPattern.**class**)  
**private** Set<UrlPattern> **urlPatterns** = **new** HashSet<UrlPattern>();  
*/\*\* A set of permissions that have been requested, granted, or refused by  
 \* the authorising agents.  
 \*/*@OneToMany(cascade = CascadeType.***ALL***, orphanRemoval = **true**, mappedBy = **"site"**, targetEntity = Permission.**class**)  
**private** Set<Permission> **permissions** = **new** HashSet<Permission>();  
*/\*\*  
 \* A set of permissions that have been removed.  
 \*/*@Transient  
**private** Set<Permission> **removedPermissions** = **new** HashSet<Permission>();  
  
*/\*\* The owning agency \*/*@ManyToOne(targetEntity = Agency.**class**)  
@JoinColumn(name = **"ST\_OWNING\_AGENCY\_ID"**)  
**private** Agency **owningAgency** = **null**;

…

#### UrlPattern.java

@Entity  
@Table(name = **"URL\_PATTERN"**)  
**public class** UrlPattern **extends** AbstractIdentityObject {

…

*/\*\* The database oid of the UrlPattern. \*/*@Id  
@Column(name = **"UP\_OID"**, nullable = **false**)  
@GeneratedValue(generator = **"urlPatternGen"**, strategy = GenerationType.***TABLE***)  
@TableGenerator(name = **"urlPatternGen"**, table = **"ID\_GENERATOR"**, pkColumnName = **"IG\_TYPE"**,  
 valueColumnName = **"IG\_VALUE"**, pkColumnValue = **"General"**)  
**private** Long **oid**;  
*/\*\* The url pattern. \*/*@Column(name = **"UP\_PATTERN"**, length = 2048, columnDefinition = **"text"**)  
**private** String **pattern**;  
*/\*\* A reference to the owning site. \*/*@ManyToOne(targetEntity = Site.**class**)  
@JoinColumn(name = **"UP\_SITE\_ID"**, foreignKey = @ForeignKey(name = **"FK\_UP\_SITE\_ID"**))  
**private** Site **site**;  
*/\*\* A set of permissions. \*/*@ManyToMany(targetEntity = Permission.**class**)  
@Cascade({org.hibernate.annotations.CascadeType.***SAVE\_UPDATE***})  
@JoinTable(name = **"PERMISSION\_URLPATTERN"**, joinColumns = {@JoinColumn(name = **"PU\_URLPATTERN\_ID"**)}, foreignKey = @ForeignKey(name = **"PU\_FK\_1"**),  
 inverseJoinColumns = {@JoinColumn(name = **"PU\_PERMISSION\_ID"**)}, inverseForeignKey = @ForeignKey(name = **"PU\_FK\_2"**))  
**private** Set<Permission> **permissions** = **new** HashSet<Permission>();

…

### Dao Configuration

The Hibernate daos were modified as follows.

The following Spring annotations were added to the top of each dao class:

@Repository  
@Transactional

The getSession() method has been removed as was replaced with the currentSession() method.

The getHibernateTemplate().findByNamedQuery() method has been deprecated and the currentSession() method should be used instead.

So

getHibernateTemplate().findByNamedQuery(Mapping.***QUERY\_BY\_OID***, mappingOid);

becomes

Query query = currentSession().getNamedQuery(Mapping.***QUERY\_BY\_OID***);  
query.setParameter(1, mappingOid);  
List results = query.list();

## Spring Core Configuration

The wct-core.xml file contains the core Spring configuration and did not change except for the Hibernate Spring configuration moving to the data-config.xml file and the addition of an XSD schema definition.

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans default-autowire="no" default-lazy-init="false" xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:tx="http://www.springframework.org/schema/tx" xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd"**>  
  
 <**bean id="WCTCoreConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"**>  
 <**property name="locations"**>  
 <**list**>  
 <**value**>classpath:wct-core.properties</**value**>  
 </**list**>  
 </**property**>  
 <**property name="ignoreResourceNotFound" value="true"**/>  
 <**property name="ignoreUnresolvablePlaceholders" value="true"**/>  
 <**property name="order" value="150"**/>  
 <**property name="properties"**>   
 <**props**>   
 *<!-- Defaults for values not specified in properties file -->* <**prop key="qualityReviewToolController.archiveName"**/>   
 <**prop key="qualityReviewToolController.archive.alternative"**/>  
 <**prop key="qualityReviewToolController.archive.alternative.name"**/>  
 <**prop key="harvestCoordinator.harvestOptimizationEnabled"**>false</**prop**>  
 <**prop key="harvestCoordinator.harvestOptimizationLookaheadHours"**>24</**prop**>  
 <**prop key="harvestCoordinator.numHarvestersExcludedFromOptimisation"**>0</**prop**>  
 </**props**>   
 </**property**>   
 </**bean**>  
  
 <**bean id="messageSource" class="org.springframework.context.support.ResourceBundleMessageSource"**>  
 <**property name="basename" value="messages"**/>  
 </**bean**>  
   
 <**bean id="userRoleDAO" class="org.webcurator.domain.UserRoleDAOImpl"**>  
 <**property name="sessionFactory" ref="sessionFactory"**/>  
 <**property name="txTemplate" ref="transactionTemplate"**/>  
 </**bean**>  
  
 <**bean id="hibernateTestDao" class="org.webcurator.domain.HibernateTestDAOImpl"**>  
 <**property name="sessionFactory" ref="sessionFactory"**/>  
 <**property name="txTemplate" ref="transactionTemplate"**/>  
 </**bean**>  
  
 <**bean id="lockManager" class="org.webcurator.core.util.LockManager" abstract="false" scope="singleton" lazy-init="default" autowire="default"**/>  
  
 <**bean id="permissionMappingStrategy" class="org.webcurator.core.permissionmapping.HierarchicalPermissionMappingStrategy" scope="singleton"**>  
 <**property name="dao" ref="permMappingDao"**/>  
 </**bean**>  
  
 <**bean id="permMappingDao" class="org.webcurator.core.permissionmapping.HierPermMappingDAOImpl"**>  
 <**property name="sessionFactory" ref="sessionFactory"**/>  
 <**property name="txTemplate" ref="transactionTemplate"**/>  
 </**bean**>  
   
 <**bean id="authorityManager" class="org.webcurator.auth.AuthorityManagerImpl" abstract="false" scope="singleton" lazy-init="default" autowire="default"**></**bean**>  
  
</**beans**>

Note that singleton=”true” attribute on the bean element is not valid in Spring 5 and it has been replaced with scope=”singleton”.

## Spring Security Configuration

The wct-core-security.xml file contains the security Spring configuration and was changed from using Acegi security to Spring security.

Most of the changes were package names and the use of constructors to pass references instead of properties.

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:sec="http://www.springframework.org/schema/security"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd  
 http://www.springframework.org/schema/security  
 http://www.springframework.org/schema/security/spring-security.xsd"**>  
  
 <**import resource="security-userdn-list.xml"**/>  
<**bean id="springSecurityFilterChain" class="org.springframework.security.web.FilterChainProxy"**>  
 <**constructor-arg**>  
 <**list**>  
 <**sec:filter-chain pattern="/curator/credentials/reset-password.html"  
 filters="httpSessionContextIntegrationFilter,authenticationProcessingFilter,exceptionTranslationFilter,filterInvocationInterceptor"** />  
 <**sec:filter-chain pattern="/curator/\*\*"  
 filters="httpSessionContextIntegrationFilter,authenticationProcessingFilter,exceptionTranslationFilter,filterInvocationInterceptor, WCTPasswordExpiredFilter"** />  
 <**sec:filter-chain pattern="/jsp/\*\*"  
 filters="httpSessionContextIntegrationFilter,authenticationProcessingFilter,exceptionTranslationFilter,filterInvocationInterceptor"** />  
 <**sec:filter-chain pattern="/j\_spring\_security\_check"  
 filters="httpSessionContextIntegrationFilter, authenticationProcessingFilter, exceptionTranslationFilter,filterInvocationInterceptor"** />  
 </**list**>  
 </**constructor-arg**>  
 </**bean**>  
   
 <**bean id="WCTPasswordExpiredFilter" class="org.webcurator.auth.dbms.WCTForcePasswordChange" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 </**bean**>  
  
 *<!-- ======================== AUTHENTICATION ======================= -->* <**bean id="authenticationManager" class="org.springframework.security.authentication.ProviderManager" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**constructor-arg**>  
 <**list**>  
 *<!-- Remove the comments if you want LDAP authentication to occur -->  
 <!-- <ref bean="ldapAuthenticator" /> -->* <**ref bean="daoAuthenticationProvider"** />  
 </**list**>  
 </**constructor-arg**>  
 </**bean**>  
   
 <**bean id="initialDirContextFactory" class="org.springframework.security.ldap.DefaultSpringSecurityContextSource"**>  
 <**constructor-arg value="${ldap.url}"**/>  
*<!-- <property name="managerDn"><value>OU=WCT Users,DC=webcurator,DC=org</value></property> -->  
<!-- <property name="managerPassword"><value>itsAsecretWord</value></property> -->* </**bean**>  
   
  
   
 *<!-- Note that the distinguished name patterns are in the file "security-userdn-list.xml" -->* <**bean id="ldapAuthenticator" class="org.springframework.security.ldap.authentication.LdapAuthenticationProvider" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**constructor-arg**>  
 <**bean class="org.springframework.security.ldap.authentication.BindAuthenticator"**>  
 <**constructor-arg**><**ref bean="initialDirContextFactory"**/></**constructor-arg**>  
*<!-- Use this if your users have the same ldap dn pattern, and the only difference -->  
<!-- is the user name -->* <**property name="userDnPatterns"**><**list**><**value**>${ldap.dn}</**value**></**list**></**property**>  
  
*<!-- Use this and edit security-userdn-list.xml if you need to specify -->  
<!-- multiple dn patterns -->  
<!-- <property name="userDnPatterns" ref="userDnList"/> -->  
  
<!-- Use this and uncomment the associated bean below if you need to specify -->  
<!-- a search (e.g. active directory), or the dn does not use the WCT -->  
<!-- user name -->  
<!-- <property name="userSearch" ref="userSearch"/> -->* </**bean**>  
 </**constructor-arg**>  
 <**constructor-arg**>  
 <**bean class="org.webcurator.auth.ldap.WCTAuthoritiesPopulator"**>  
 <**property name="authDAO"**><**ref bean="userRoleDAO"**/></**property**>  
 </**bean**>  
 </**constructor-arg**>  
 </**bean**>  
   
 <**bean id="daoAuthenticationProvider" class="org.springframework.security.authentication.dao.DaoAuthenticationProvider"**>  
 <**property name="userDetailsService"**><**ref bean="jdbcDaoImpl"**/></**property**>  
 <**property name="passwordEncoder"**><**ref bean="passwordEncoder"**/></**property**>  
 </**bean**>  
   
 <**bean id="passwordEncoder" class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"**>  
 </**bean**>  
  
 <**bean id="jdbcDaoImpl" class="org.webcurator.auth.dbms.WCTDAOAuthenticationProvider"**>  
 <**property name="dataSource"**><**ref bean="dataSource"**/></**property**>  
 <**property name="usersByUsernameQuery"**><**value**>select usr\_username, usr\_password, usr\_active, usr\_force\_pwd\_change from ${hibernate.default\_schema}.WCTUSER WHERE usr\_username = ?</**value**></**property**>  
 <**property name="authoritiesByUsernameQuery"**><**value**>SELECT   
 distinct PRV\_CODE  
 FROM   
 ${hibernate.default\_schema}.WCTUSER,  
 ${hibernate.default\_schema}.WCTROLE,  
 ${hibernate.default\_schema}.USER\_ROLE,  
 ${hibernate.default\_schema}.ROLE\_PRIVILEGE  
 WHERE   
 PRV\_ROLE\_OID = ROL\_OID and  
 URO\_USR\_OID = USR\_OID and  
 URO\_ROL\_OID = ROL\_OID and  
 usr\_username = ?  
 </**value**></**property**>  
 <**property name="rolePrefix"**><**value**>ROLE\_</**value**></**property**>  
 </**bean**>  
   
  
 *<!-- Automatically receives AuthenticationEvent messages -->* <**bean id="loggerListener" class="org.springframework.security.authentication.event.LoggerListener"**/>  
  
 <**bean id="httpSessionContextIntegrationFilter" class="org.springframework.security.web.context.SecurityContextPersistenceFilter"**>  
 </**bean**>  
  
 *<!-- ===================== HTTP CHANNEL REQUIREMENTS ==================== -->  
  
 <!-- You will need to uncomment the "Acegi Channel Processing Filter"  
 <filter-mapping> in web.xml for the following beans to be used -->* <**bean id="channelProcessingFilter" class="org.springframework.security.web.access.channel.ChannelProcessingFilter"**>  
 <**property name="channelDecisionManager"**><**ref bean="channelDecisionManager"**/></**property**>  
 <**property name="securityMetadataSource"**>  
 <**sec:filter-security-metadata-source request-matcher="regex" use-expressions="false"**>  
 <**sec:intercept-url pattern="\A/acegilogin.jsp.\*\Z" access="REQUIRES\_SECURE\_CHANNEL"**/>  
 <**sec:intercept-url pattern="\A.\*\Z" access="REQUIRES\_INSECURE\_CHANNEL"**/>  
 </**sec:filter-security-metadata-source**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="channelDecisionManager" class="org.springframework.security.web.access.channel.ChannelDecisionManagerImpl"**>  
 <**property name="channelProcessors"**>  
 <**list**>  
 <**ref bean="secureChannelProcessor"**/>  
 <**ref bean="insecureChannelProcessor"**/>  
 </**list**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="secureChannelProcessor" class="org.springframework.security.web.access.channel.SecureChannelProcessor"**/>  
 <**bean id="insecureChannelProcessor" class="org.springframework.security.web.access.channel.InsecureChannelProcessor"**/>  
  
 *<!-- ===================== HTTP REQUEST SECURITY ==================== -->* <**bean id="exceptionTranslationFilter" class="org.springframework.security.web.access.ExceptionTranslationFilter"**>  
 <**constructor-arg**>  
 <**ref bean="authenticationProcessingFilterEntryPoint"**/>  
 </**constructor-arg**>  
 </**bean**>  
  
 <**bean id="authenticationProcessingFilter" class="org.webcurator.auth.WCTAuthenticationProcessingFilter"**>  
 <**property name="authenticationManager"**><**ref bean="authenticationManager"**/></**property**>  
 <**property name="authenticationFailureHandler" ref="failureHandler"** />  
 <**property name="authenticationSuccessHandler" ref="successHandler"** />  
 <**property name="filterProcessesUrl"**><**value**>/j\_spring\_security\_check</**value**></**property**>  
 <**property name="authDAO"**><**ref bean="userRoleDAO"**/></**property**>  
 </**bean**>  
  
 <**bean id="successHandler" class="org.springframework.security.web.authentication.SimpleUrlAuthenticationSuccessHandler"** >  
 <**constructor-arg**>  
 <**value**>/curator/home.html</**value**>  
 </**constructor-arg**>  
 </**bean**>  
  
 <**bean id="failureHandler" class="org.springframework.security.web.authentication.SimpleUrlAuthenticationFailureHandler"** >  
 <**property name="defaultFailureUrl" value="/logon.jsp?failed=true"** />  
 </**bean**>  
   
 <**bean id="authenticationProcessingFilterEntryPoint" class="org.springframework.security.web.authentication.LoginUrlAuthenticationEntryPoint" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**constructor-arg**>  
 <**value**>/logon.jsp</**value**>  
 </**constructor-arg**>  
 <**property name="forceHttps"**>  
 <**value**>false</**value**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="httpRequestAccessDecisionManager" class="org.springframework.security.access.vote.AffirmativeBased"**>  
 <**property name="allowIfAllAbstainDecisions"**><**value**>false</**value**></**property**>  
 <**constructor-arg**>  
 <**list**>  
 <**bean class="org.springframework.security.web.access.expression.WebExpressionVoter"** />  
 <**ref bean="roleVoter"**/>  
 </**list**>  
 </**constructor-arg**>  
 </**bean**>  
   
 <**bean id="roleVoter" class="org.springframework.security.access.vote.RoleVoter"**/>  
  
 *<!-- Note the order that entries are placed against the securityMetadataSource is critical.  
 The FilterSecurityInterceptor will work from the top of the list down to the FIRST pattern that matches the request URL.  
 Accordingly, you should place MOST SPECIFIC (ie a/b/c/d.\*) expressions first, with LEAST SPECIFIC (ie a/.\*) expressions last -->* <**bean id="filterInvocationInterceptor" class="org.springframework.security.web.access.intercept.FilterSecurityInterceptor"**>  
 <**property name="authenticationManager"**><**ref bean="authenticationManager"**/></**property**>  
 <**property name="accessDecisionManager"**><**ref bean="httpRequestAccessDecisionManager"**/></**property**>  
 <**property name="securityMetadataSource"**>  
 <**sec:filter-security-metadata-source**>  
 <**sec:intercept-url pattern="/index.jsp" access="hasAnyRole('ROLE\_ANONYMOUS','ROLE\_USER')"**/>  
 <**sec:intercept-url pattern="/logon.jsp\*" access="hasAnyRole('ROLE\_ANONYMOUS','ROLE\_USER')"**/>  
 <**sec:intercept-url pattern="/\*\*" access="hasAnyRole('ROLE\_ADM','ROLE\_LOGIN')"**/>  
 </**sec:filter-security-metadata-source**>  
 </**property**>  
 </**bean**>  
  
</**beans**>

### Password Encoder Change

The main change between Acegi and Spring security is the removal of the SHA password encoder and its replacement with a BCrypt password encoder.

<**bean id="passwordEncoder" class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"**>  
</**bean**>

This means that all of the passwords stored in the WCTUSER table need to be rehashed to use the BCrypt password encoder.

The BCryptPasswordEncoderGenerator utility class was written to hash a password with the BCrypt password encoder.

**package** org.webcurator.core.util;  
  
**import** org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
  
**public class** BCryptPasswordEncoderGenerator {  
  
 **public static void** main(String[] args) {  
 **if** (args == **null** || args.**length** == 0) {  
 System.***out***.println(**"Please supply a password to encode"**);  
 } **else** {  
 String passwordToEncode = args[0];  
 BCryptPasswordEncoder passwordEncoder = **new** BCryptPasswordEncoder();  
 String hashedPassword = passwordEncoder.encode(passwordToEncode);  
 System.***out***.println(**"Hashed Password: "** + hashedPassword);  
 }  
 }  
}

Supplying “abc123” as a password on the command line results in:

Hashed Password: $2a$10$GkMe.8rFudMmEkLSL1FlTOSAh/GB6OMdm1Z5fkg5UDzL1ZHtwbGGa

This hashed password can then be used to update the USR\_PASSWORD column for a user in the WCTUSER table. The user will then be able to log on using “abc123” as a password.

### Spring Security Classes

#### WCTAuthenticationProcessingFilter.java

Extends a different class:

**public class** WCTAuthenticationProcessingFilter **extends** UsernamePasswordAuthenticationFilter {

The onSuccessfulAuthentication() method becomes the successfulAuthentication() method:

**protected void** successfulAuthentication(HttpServletRequest request, HttpServletResponse response, FilterChain chain, Authentication authResult)

The authentication token no longer has to be retrieved from the Security Context or put back.

*log*.debug(**"loaded WCT User object "**+wctUser.getUsername()+**" from database to update authentication token details"**);  
 UsernamePasswordAuthenticationToken auth = (UsernamePasswordAuthenticationToken)authResult;  
 auth.setDetails(wctUser);

The successfulAuthentication() method of the parent class has to be called so WCT will go to the home page.

*// goto the home page* **try** {  
 **super**.successfulAuthentication(request, response, chain, authResult);  
 } **catch** (ServletException e) {  
 *log*.error(e);  
 }

The onUnsuccessfulAuthentication() method becomes the unsuccessfulAuthentication() method:

**protected void** unsuccessfulAuthentication(HttpServletRequest aReq, HttpServletResponse aRes, AuthenticationException e) **throws** IOException, ServletException {

#### WCTDaoAuthenticationProvider.java

The main difference is that the GrantedAuthorityImpl class has been replaced by the SimpleGrantedAuthority class:

SimpleGrantedAuthority authority = **new** SimpleGrantedAuthority(roleName);

Also the initMappingSqlQueries() method has been replaced with the initDao() method.

**protected void** initDao() {  
 **this**.**usersByUsernameMapping** = **new** UsersByUsernameMapping(getDataSource());  
 **this**.**authoritiesByUsernameMapping** = **new** AuthoritiesByUsernameMapping(getDataSource());  
}

#### WCTForcePasswordChange.java

The Acegi import statements have been replaced with Spring Security ones:

**import** org.springframework.security.core.Authentication;  
**import** org.springframework.security.core.context.SecurityContextHolder;

#### AuthUtil.java

The Acegi import statements have been replaced with Spring Security ones:

**import** org.springframework.security.core.Authentication;  
**import** org.springframework.security.core.context.SecurityContextHolder;

#### SpringSecurityLogoutListener.java

This was originally called AcegiLogoutListener.

In the sessionDestroyed() method the SecurityContext is retrieved from the HTTP Session using the “SPRING\_SECURITY\_CONTEXT” key:

String remoteUser = **null**;  
Authentication auth = **null**;   
SecurityContext springCtx = (SecurityContext) event.getSession().getAttribute(**"SPRING\_SECURITY\_CONTEXT"**);  
**if**( springCtx != **null**) {  
 auth = springCtx.getAuthentication();  
 **if** (auth != **null**) {  
 remoteUser = auth.getName();  
 }  
}

### Logon Page

The logon.jsp was modified for Spring Security.

The form action was modified:

<form name=**"login"** action=**"j\_spring\_security\_check"** method=**"POST"**>

The user name and password text input field names were modified from j\_username and j\_password respectively:

<div id=**"homeBoxText"**>  
 username<br />  
 <input type=**"text"** name=**"username"** width=**"20"** style=**"width:200px"**><br />  
 password<br />  
 <input type=**"password"** name=**"password"** width=**"20"** style=**"width:200px"**><br />  
 <input type=**"image"** src=**"images/home-btn-login.gif"** alt=**"login"** width=**"67"** height=**"18"** border=**"0"** vspace=**"5"** />  
</div>

### web.xml

The Acegi Filter Chain Proxy filter is replaced by a Spring Security filter:

<**filter**>  
 <**filter-name**>springSecurityFilterChain</**filter-name**>  
 <**filter-class**>org.springframework.web.filter.DelegatingFilterProxy</**filter-class**>  
 </**filter**>

…

<**filter-mapping**>  
 <**filter-name**>springSecurityFilterChain</**filter-name**>  
 <**url-pattern**>/\*</**url-pattern**>  
</**filter-mapping**>

The Acegi logout listener is replaced with a Spring Security one:

<**listener**>  
 <**listener-class**>org.webcurator.ui.listener.SpringSecurityLogoutListener</**listener-class**>  
</**listener**>

## Spring UI Configuration

The wct-core-servlet.xml file is largely unchanged except for the addition of the Hibernate Test and WS Test controllers and the updated Hibernate 5 and Tiles 3 classes.

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans default-autowire="no" default-lazy-init="false" xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd"**>  
  
 <**bean id="WCTCoreServletConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"**>  
 <**property name="locations"**>  
 <**list**>  
 <**value**>classpath:wct-core.properties</**value**>  
 </**list**>  
 </**property**>  
 <**property name="ignoreResourceNotFound" value="true"**/>  
 <**property name="ignoreUnresolvablePlaceholders" value="true"**/>  
 <**property name="order" value="150"**/>  
 </**bean**>  
   
 <**bean id="simpleUrlMapping" class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="interceptors"**>  
 <**list**>  
 <**ref bean="openSessionInViewInterceptor"**/>  
 </**list**>  
 </**property**>  
   
 <**property name="mappings"**>  
 <**map**>  
 <**entry key="/curator/logout.html" value="logoutController"**/>  
 <**entry key="/curator/home.html" value="homeController"**/>  
 <**entry key="/curator/hibernate-test.html" value="hibernateTestController"**/>  
 <**entry key="/curator/hibernate-test-add.html" value="hibernateTestAddController"**/>  
 <**entry key="/curator/ws-test.html" value="harvestAgentSOAPClientController"**/>  
 </**map**>  
 </**property**>  
 </**bean**>  
   
 <**bean id="localeResolver" class="org.springframework.web.servlet.i18n.FixedLocaleResolver"**/>  
   
 <**bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver"**>  
 <**property name="maxUploadSize" value="1500000"**/>  
 </**bean**>   
  
 <**bean id="viewResolver" class="org.springframework.web.servlet.view.tiles3.TilesViewResolver"**>  
 <**property name="requestContextAttribute" value="requestContext"**/>  
 <**property name="viewClass" value="org.springframework.web.servlet.view.tiles3.TilesView"**/>  
 </**bean**>  
   
 <**bean id="tilesConfigurer" class="org.springframework.web.servlet.view.tiles3.TilesConfigurer"**>  
 *<!--<property name="factoryClass" value="org.apache.struts.tiles.xmlDefinition.I18nFactorySet"/>-->* <**property name="definitions"**>  
 <**list**>  
 <**value**>/WEB-INF/tiles-defs.xml</**value**>  
 </**list**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="exceptionResolver"  
 class="org.springframework.web.servlet.handler.SimpleMappingExceptionResolver"  
 abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="defaultErrorView"**>  
 <**value type="java.lang.String"**>Error</**value**>  
 </**property**>  
 <**property name="exceptionMappings"**>  
 <**map**>  
 <**entry  
 key="org.springframework.orm.hibernate5.HibernateObjectRetrievalFailureException"  
 value="NoObjectFound"**>  
 </**entry**>  
 <**entry  
 key="org.springframework.web.multipart.MaxUploadSizeExceededException"  
 value="max-file-size-exceeeded"**/>  
 </**map**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="homeController" class="org.webcurator.ui.home.controller.HomeController" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="supportedMethods" value="GET"**/>  
 <**property name="enableQaModule"**>  
 <**value type="java.lang.Boolean"**>${queueController.enableQaModule}</**value**>  
 </**property**>   
 </**bean**>  
  
 <**bean id="openSessionInViewInterceptor" class="org.springframework.orm.hibernate5.support.OpenSessionInViewInterceptor" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="sessionFactory"**><**ref bean="sessionFactory"**/></**property**>  
 </**bean**>  
   
 <**bean id="logoutController" class="org.webcurator.ui.base.LogoutController" abstract="false" scope="singleton" lazy-init="default" autowire="default"**/>  
  
 <**bean id="hibernateTestController" class="org.webcurator.ui.hibernate.HibernateTestController" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="supportedMethods" value="GET,POST"**/>  
 <**property name="hibernateTestDAO" ref="hibernateTestDao"**/>  
 </**bean**>  
  
 <**bean id="hibernateTestAddController" class="org.webcurator.ui.hibernate.HibernateTestAddController" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="supportedMethods" value="POST"**/>  
 <**property name="hibernateTestDAO" ref="hibernateTestDao"**/>  
 </**bean**>  
  
 <**bean id="harvestAgentSOAPClientController" class="org.webcurator.ui.hibernate.HarvestAgentSOAPClientController" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="supportedMethods" value="GET,POST"**/>  
 <**property name="harvestAgentSOAPClient" ref="harvestAgentSOAPClient"**/>  
 </**bean**>  
  
 <**bean id="harvestAgentJAXBMarshaller" class="org.springframework.oxm.jaxb.Jaxb2Marshaller"**>  
 *<!-- this is the package name specified in the <generatePackage> in the pom.xml -->* <**property name="contextPath" value="org.webcurator.wsclient"**/>  
 </**bean**>  
  
 <**bean id="harvestAgentSOAPConnector" class="org.webcurator.soap.client.SOAPConnector"**>  
 <**property name="defaultUri" value="http://localhost:8080/wct/service/harvest-agent"**/>  
 <**property name="marshaller" ref="harvestAgentJAXBMarshaller"**/>  
 <**property name="unmarshaller" ref="harvestAgentJAXBMarshaller"**/>  
 </**bean**>  
  
 <**bean id="harvestAgentSOAPClient" class="org.webcurator.soap.client.HarvestAgentSOAPClient"**>  
 <**property name="soapConnector" ref="harvestAgentSOAPConnector"**/>  
 </**bean**>  
  
</**beans**>

### Tiles Configuration

The tiles-defs.xml file has an entirely new schema.

The path attribute of the definition element is now called template and the put element is now called put-attribute. Also a type attribute in the put-attribute element needs to be defined when the value attribute is not a reference to another JSP page.

*<?***xml version="1.0" encoding="ISO-8859-1"** *?>***<!DOCTYPE tiles-definitions PUBLIC "-//Apache Software Foundation//DTD Tiles Configuration 3.0//EN" "http://tiles.apache.org/dtds/tiles-config\_3\_0.dtd"*>***<**tiles-definitions**>  
 <**definition name=".header" template="/jsp/layouts/header.jsp"**>  
 <**put-attribute name="page-help" value="Header Help" type="string" cascade="true"** />  
 </**definition**>  
  
 <**definition name=".generic-object-identifier" template="/jsp/layouts/object-identifier-generic.jsp"**>  
 <**put-attribute name="object-name" value="name"** />  
 </**definition**>  
  
 <**definition name=".subTabbedNamed" template="/jsp/layouts/sub-tabbed-named.jsp"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="validation" value="/jsp/layouts/validation-messages.jsp"** />  
 <**put-attribute name="footer-nav" value="/jsp/layouts/footer-nav.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Subtabbed Layout Help" type="string"** />  
 <**put-attribute name="object-identifier" value=".generic-object-identifier"** />  
 <**put-attribute name="object-name" value="Name"** />  
 </**definition**>  
  
 <**definition name=".reportLayout" template="/jsp/layouts/basic-report.jsp"**>  
 <**put-attribute name="title" value="Web Curator Tool"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="validation" value="/jsp/layouts/validation-messages.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Report Layout Help" type="string"** />  
 </**definition**>  
  
 <**definition name=".mainLayout" template="/jsp/layouts/basic.jsp"**>  
 <**put-attribute name="title" value="Web Curator Tool"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="validation" value="/jsp/layouts/validation-messages.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Main Layout Help" type="string"** />  
 </**definition**>  
  
 <**definition name=".popup" template="/jsp/layouts/popup.jsp"**>  
 <**put-attribute name="title" value="Web Curator Tool"** />  
 </**definition**>  
  
  
 <**definition name=".ajax" template="/jsp/layouts/blank-AJAX.jsp"**>  
 </**definition**>  
  
 <**definition name=".mainLayoutNew" template="/jsp/layouts/basic-new.jsp"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="tab-nav" value="/jsp/layouts/tab-nav.jsp"** />  
 <**put-attribute name="validation" value="/jsp/layouts/validation-messages.jsp"** />  
 <**put-attribute name="footer-nav" value="/jsp/layouts/footer-nav.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Main Layout New Help" type="string"** />  
 </**definition**>  
  
 <**definition name=".errorLayout" template="/jsp/layouts/error-template.jsp"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="page-icon" value="/jsp/layouts/page-icon-error.jsp"** />  
 <**put-attribute name="footer-nav" value="/jsp/layouts/footer-nav.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Error Help" type="string"** />  
 </**definition**>  
  
 <**definition name=".tabbedNew" template="/jsp/layouts/tabbed-new.jsp"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="tab-nav" value="/jsp/layouts/tab-nav.jsp"** />  
 <**put-attribute name="validation" value="/jsp/layouts/validation-messages.jsp"** />  
 <**put-attribute name="footer-nav" value="/jsp/layouts/footer-nav.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Tabbed Layout Help" type="string"** />  
 </**definition**>  
  
 <**definition name=".subTabbed" template="/jsp/layouts/sub-tabbed.jsp"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="header" value=".header"** />  
 <**put-attribute name="validation" value="/jsp/layouts/validation-messages.jsp"** />  
 <**put-attribute name="footer-nav" value="/jsp/layouts/footer-nav.jsp"** />  
 <**put-attribute name="footer" value="/jsp/layouts/footer.jsp"** />  
 <**put-attribute name="page-help" value="Subtabbed Layout Help" type="string"** />  
 </**definition**>  
  
 <**definition name="Error" extends=".errorLayout"**>  
 <**put-attribute name="title" value="Error"** />  
 <**put-attribute name="body" value="/jsp/layouts/Error.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
  
 <**definition name="max-file-size-exceeeded" extends=".errorLayout"**>  
 <**put-attribute name="title" value="Maximum File Upload Size Exceeeded"** />  
 <**put-attribute name="body" value="/jsp/max-file-size-exceeeded.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="NoObjectFound" extends=".errorLayout"**>  
 <**put-attribute name="title" value="Object Not Found"** />  
 <**put-attribute name="body" value="/jsp/no-object-found.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="HomeView" extends=".mainLayout"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="body" value="/jsp/home.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="HibernateTestView" extends=".mainLayout"**>  
 <**put-attribute name="title" value="Hibernate Test"** />  
 <**put-attribute name="body" value="/jsp/hibernate-test.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="WSTestView" extends=".mainLayout"**>  
 <**put-attribute name="title" value="WS Test"** />  
 <**put-attribute name="body" value="/jsp/ws-test.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="QaHomeView" extends=".mainLayout"**>  
 <**put-attribute name="title" value="WEB CURATOR TOOL"** />  
 <**put-attribute name="body" value="/jsp/qa-home.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="authorisation-failure" extends=".mainLayout"**>  
 <**put-attribute name="title" value="Authorisation Failure"** />  
 <**put-attribute name="body" value="/jsp/authorisation-failure.jsp"** />  
 <**put-attribute name="page-help" value="http://dia-nz.github.io/webcurator/" type="string"** />  
 </**definition**>  
  
 <**definition name="Logout" template="/jsp/logout.jsp"** />  
  
</**tiles-definitions**>

### JSP Pages

The tiles taglib uri on all JSP pages was changed from a struts tld to:

<%@ taglib uri=**"http://tiles.apache.org/tags-tiles"** prefix=**"tiles"** %>

The tiles taglib has changed as follows:

|  |  |
| --- | --- |
| Old Tag | New Tag |
| <tiles:insert attribute=””> | <tiles:insertAttribute name=””> |
| <tiles:put name=””> | <tiles:putAttribute name=””> |

The below example demonstrates the new tiles taglib.

<c:set var=**"pageHelpValue"** scope=**"request"**><tiles:getAsString name=**"page-help"**/></c:set>  
<tiles:insertAttribute name=**"header"**>  
 <tiles:putAttribute name=**"page-help"** type=**"string"**><c:out value=**"${pageHelpValue}"**/></tiles:putAttribute>  
</tiles:insertAttribute>  
<br class=**"clear"** />  
<tiles:insertAttribute name=**"validation"** />  
  
<tiles:insertAttribute name=**"body"** />

Note that the <tiles:getAsString> tag is not nested inside the <tiles:putAttribute> tag but is set to a local variable (pageHelpValue) which is then used inside the <tiles:putAttribute> tag.

This is a workaround to avoid the following exception that occurs when the <tiles:getAsString> tag is nested inside the <tiles:putAttribute> tag:

HTTP ERROR 500

Problem accessing /wct/curator/home.html. Reason:

Server Error

Caused by:

javax.servlet.ServletException: org.springframework.web.util.NestedServletException: Request processing failed; nested exception is org.apache.tiles.request.render.CannotRenderException: An exception occurred processing JSP page [/jsp/layouts/basic.jsp] at line [49]

46:

47: <body>

48: <a name="top"></a>

49: <tiles:insertAttribute name="header">

50: <tiles:putAttribute name="page-help" type="string">

51: <tiles:getAsString name="page-help"/>

52: </tiles:putAttribute>

…

Caused by: javax.servlet.jsp.JspException: org.apache.tiles.template.NoSuchAttributeException: Attribute 'page-help' not found.

at org.apache.jsp.jsp.layouts.basic\_jsp$Helper.invoke(basic\_jsp.java:371)

at org.apache.tiles.request.jsp.autotag.JspModelBody.evaluate(JspModelBody.java:62)

... 112 more

Caused by: org.apache.tiles.template.NoSuchAttributeException: Attribute 'page-help' not found.

at org.apache.tiles.template.DefaultAttributeResolver.computeAttribute(DefaultAttributeResolver.java:50)

at org.apache.tiles.template.GetAsStringModel.resolveAttribute(GetAsStringModel.java:152)

at org.apache.tiles.template.GetAsStringModel.execute(GetAsStringModel.java:110)

at org.apache.tiles.jsp.taglib.GetAsStringTag.doTag(GetAsStringTag.java:261)

at org.apache.jsp.jsp.layouts.basic\_jsp.\_jspx\_meth\_tiles\_005fgetAsString\_005f1(basic\_jsp.java:270)

at org.apache.jsp.jsp.layouts.basic\_jsp.access$1(basic\_jsp.java:258)

at org.apache.jsp.jsp.layouts.basic\_jsp$Helper.invoke1(basic\_jsp.java:340)

at org.apache.jsp.jsp.layouts.basic\_jsp$Helper.invoke(basic\_jsp.java:363)

... 113 more

…

org.apache.tiles.template.NoSuchAttributeException: Attribute 'page-help' not found.

at org.apache.tiles.template.DefaultAttributeResolver.computeAttribute(DefaultAttributeResolver.java:50)

Further investigation will be required to resolve this.

## Spring WS

A test SOAP web service endpoint and a SOAP client were created to demonstrate the replacement of Axis with Spring WS.

A Harvest Agent web service with one method (getHarvestAgentStatus) was created and a client to consume this web service was also created.

### web.xml

The following changes were made to the web.xml file to enable Spring WS.

The AxisHTTPSessionListener was removed and replaced with:

<**servlet**>  
 <**servlet-name**>ws</**servlet-name**>  
 <**servlet-class**>org.springframework.ws.transport.http.MessageDispatcherServlet</**servlet-class**>  
 <**init-param**>  
 <**param-name**>transformWsdlLocations</**param-name**>  
 <**param-value**>true</**param-value**>  
 </**init-param**>  
</**servlet**>  
<**servlet-mapping**>  
 <**servlet-name**>ws</**servlet-name**>  
 <**url-pattern**>/service/\*</**url-pattern**>  
</**servlet-mapping**>

The AxisServlet was also removed.

### Harvest Agent Web Service

#### XSD Schema

The schema file for the Harvest Agent web service was created in the main resources directory at schemas/harvestAgent.xsd:

<**xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://webcurator.org/xml/harvestagent"  
 targetNamespace="http://webcurator.org/xml/harvestagent" elementFormDefault="qualified"**>  
  
 <**xs:element name="HarvestAgentStatusRequest"**>  
 <**xs:complexType**>  
 <**xs:sequence**>  
 <**xs:element name="jobNumber" type="xs:string"**/>  
 </**xs:sequence**>  
 </**xs:complexType**>  
 </**xs:element**>  
  
 <**xs:element name="HarvestAgentStatusResponse"**>  
 <**xs:complexType**>  
 <**xs:sequence**>  
 <**xs:element name="HarvestAgentStatus" type="tns:HarvestAgentStatus"**/>  
 </**xs:sequence**>  
 </**xs:complexType**>  
 </**xs:element**>  
  
 <**xs:complexType name="HarvestAgentStatus"**>  
 <**xs:sequence**>  
 <**xs:element name="jobNumber" type="xs:string"**/>  
 <**xs:element name="status" type="xs:int"**/>  
 <**xs:element name="message" type="xs:string"**/>  
 </**xs:sequence**>  
 </**xs:complexType**>  
  
</**xs:schema**>

#### Maven Plugin

<**configuration**>  
<**outputDirectory**>src/main/java</**outputDirectory**>  
 <**clearOutputDir**>false</**clearOutputDir**>  
 <**sources**>  
 <**source**>src/main/resources/schemas</**source**>  
 </**sources**>  
 <**noGeneratedHeaderComments**>true</**noGeneratedHeaderComments**>  
 <**generateEpisode**>false</**generateEpisode**>  
</**configuration**>

The jaxb2-maven-plugin reads the schema file and generates classes in the org.webcurator.xml.harvestagent package in the main src directory. Note that this package is derived for the targetNamespace in the schema.

These classes are used by the Web Service endpoint.

#### Web Service Endpoint

The web service endpoint class was created to implement the web service behaviour.

**package** org.webcurator.soap.service;  
  
**import** org.apache.commons.logging.Log;  
**import** org.apache.commons.logging.LogFactory;  
**import** org.springframework.ws.server.endpoint.annotation.Endpoint;  
**import** org.springframework.ws.server.endpoint.annotation.PayloadRoot;  
**import** org.springframework.ws.server.endpoint.annotation.RequestPayload;  
**import** org.springframework.ws.server.endpoint.annotation.ResponsePayload;  
**import** org.webcurator.xml.harvestagent.HarvestAgentStatus;  
**import** org.webcurator.xml.harvestagent.HarvestAgentStatusRequest;  
**import** org.webcurator.xml.harvestagent.HarvestAgentStatusResponse;  
  
@Endpoint  
**public class** HarvestAgentEndpoint {  
 **private static** Log *log* = LogFactory.*getLog*(HarvestAgentEndpoint.**class**);  
 **public static final** String ***NAMESPACE\_URI*** = **"http://webcurator.org/xml/harvestagent"**;  
  
 @PayloadRoot(namespace = ***NAMESPACE\_URI***, localPart = **"HarvestAgentStatusRequest"**)  
 @ResponsePayload  
 **public** HarvestAgentStatusResponse getHarvestAgentStatus(@RequestPayload HarvestAgentStatusRequest request) {  
 *log*.debug(**"Calling Harvest Agent SOAP Web Service - getHarvestAgentStatus: jobNumber="** + request.getJobNumber());  
 HarvestAgentStatusResponse harvestAgentStatusResponse = **new** HarvestAgentStatusResponse();  
 HarvestAgentStatus harvestAgentStatus = **new** HarvestAgentStatus();  
 harvestAgentStatus.setJobNumber(request.getJobNumber());  
 harvestAgentStatus.setStatus(1);  
 harvestAgentStatus.setMessage(**"Job "** + request.getJobNumber() + **" successful"**);  
 harvestAgentStatusResponse.setHarvestAgentStatus(harvestAgentStatus);  
 **return** harvestAgentStatusResponse;  
 }  
}

#### Spring Configuration

A ws-servlet.xml file was created to define the WSDL and the endpoint.

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans default-autowire="no" default-lazy-init="false" xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:context="http://www.springframework.org/schema/context"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd"**>  
  
 *<!-- PayloadRootAnnotationMethodEndpointMapping is the Mapping that detects and handles the @PayloadRoot Annotation -->* <**bean class="org.springframework.ws.server.endpoint.mapping.PayloadRootAnnotationMethodEndpointMapping"**>  
 <**property name="interceptors"**>  
 <**list**>  
 <**bean class="org.springframework.ws.server.endpoint.interceptor.PayloadLoggingInterceptor"**/>  
 </**list**>  
 </**property**>  
 </**bean**>  
  
 <**bean id="harvestAgentEndpoint" class="org.webcurator.soap.service.HarvestAgentEndpoint"**>  
 </**bean**>  
  
 <**bean id="harvestAgentWsdl" class="org.springframework.ws.wsdl.wsdl11.DefaultWsdl11Definition"**>  
 <**property name="schema" ref="harvestAgentSchema"**/>  
 <**property name="portTypeName" value="HarvestAgentPort"**/>  
 <**property name="locationUri" value="/service/harvest-agent"** />  
 <**property name="targetNamespace" value="http://webcurator.org/xml/harvestagent"**/>  
 </**bean**>  
  
 <**bean id="harvestAgentSchema" class="org.springframework.xml.xsd.SimpleXsdSchema"**>  
 <**property name="xsd" value="/WEB-INF/classes/schemas/harvestAgent.xsd"**/>  
 </**bean**>  
  
</**beans**>

Note that the locationUri starts with the url pattern defined for the Spring WS servlet in the web.xml file.

The WSDL url is <http://localhost:8080/wct/service/harvestAgentWsdl.wsdl>

The harvestAgentWsdl part of the WSDL url is the bean id of the WSDL definition defined above in the ws-servlet.xml file.

### Harvest Agent Web Service Client

#### WSDL

The WSDL above was saved to the main resources directory at wsdl/HarvestAgent.wsdl:

<**wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:sch="http://webcurator.org/xml/harvestagent" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://webcurator.org/xml/harvestagent" targetNamespace="http://webcurator.org/xml/harvestagent"**>  
 <**wsdl:types xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"**>  
 <**xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://webcurator.org/xml/harvestagent" elementFormDefault="qualified" targetNamespace="http://webcurator.org/xml/harvestagent"**>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="HarvestAgentStatusRequest"**>  
 <**xs:complexType xmlns:xs="http://www.w3.org/2001/XMLSchema"**>  
 <**xs:sequence xmlns:xs="http://www.w3.org/2001/XMLSchema"**>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="jobNumber" type="xs:string"**/>  
 </**xs:sequence**>  
 </**xs:complexType**>  
 </**xs:element**>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="HarvestAgentStatusResponse"**>  
 <**xs:complexType xmlns:xs="http://www.w3.org/2001/XMLSchema"**>  
 <**xs:sequence xmlns:xs="http://www.w3.org/2001/XMLSchema"**>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="HarvestAgentStatus" type="tns:HarvestAgentStatus"**/>  
 </**xs:sequence**>  
 </**xs:complexType**>  
 </**xs:element**>  
 <**xs:complexType xmlns:xs="http://www.w3.org/2001/XMLSchema" name="HarvestAgentStatus"**>  
 <**xs:sequence xmlns:xs="http://www.w3.org/2001/XMLSchema"**>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="jobNumber" type="xs:string"**/>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="status" type="xs:int"**/>  
 <**xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="message" type="xs:string"**/>  
 </**xs:sequence**>  
 </**xs:complexType**>  
 </**xs:schema**>  
 </**wsdl:types**>  
 <**wsdl:message xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentStatusRequest"**>  
 <**wsdl:part xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" element="tns:HarvestAgentStatusRequest" name="HarvestAgentStatusRequest"**/>  
 </**wsdl:message**>  
 <**wsdl:message xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentStatusResponse"**>  
 <**wsdl:part xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" element="tns:HarvestAgentStatusResponse" name="HarvestAgentStatusResponse"**/>  
 </**wsdl:message**>  
 <**wsdl:portType xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentPort"**>  
 <**wsdl:operation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentStatus"**>  
 <**wsdl:input xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" message="tns:HarvestAgentStatusRequest" name="HarvestAgentStatusRequest"**/>  
 <**wsdl:output xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" message="tns:HarvestAgentStatusResponse" name="HarvestAgentStatusResponse"**/>  
 </**wsdl:operation**>  
 </**wsdl:portType**>  
 <**wsdl:binding xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentPortSoap11" type="tns:HarvestAgentPort"**>  
 <**soap:binding xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" style="document" transport="http://schemas.xmlsoap.org/soap/http"**/>  
 <**wsdl:operation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentStatus"**>  
 <**soap:operation xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" soapAction=""**/>  
 <**wsdl:input xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentStatusRequest"**>  
 <**soap:body xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" use="literal"**/>  
 </**wsdl:input**>  
 <**wsdl:output xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentStatusResponse"**>  
 <**soap:body xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" use="literal"**/>  
 </**wsdl:output**>  
 </**wsdl:operation**>  
 </**wsdl:binding**>  
 <**wsdl:service xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="HarvestAgentPortService"**>  
 <**wsdl:port xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" binding="tns:HarvestAgentPortSoap11" name="HarvestAgentPortSoap11"**>  
 <**soap:address xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" location="http://localhost:8080/wct/service/harvest-agent"**/>  
 </**wsdl:port**>  
 </**wsdl:service**>  
</**wsdl:definitions**>

#### Maven Plugin

<**configuration**>  
 <**strict**>false</**strict**>  
 <**schemaLanguage**>WSDL</**schemaLanguage**>  
 <**generatePackage**>org.webcurator.wsclient</**generatePackage**>  
 <**schemaDirectory**>src/main/resources/wsdl</**schemaDirectory**>  
 <**schemaIncludes**>  
 <**include**>\*.wsdl</**include**>  
 </**schemaIncludes**>  
</**configuration**>

The maven-jaxb2-plugin reads the WSDL file and generates classes in the target/generated-sources/xjc directory using the defined org.webcurator.wsclient package.

These classes are used by the Web Service client.

#### Web Service Client

A SOAPConnector class was created to marshal any web service request. This class is used by the Web Service client.

**package** org.webcurator.soap.client;  
  
**import** org.springframework.ws.client.core.support.WebServiceGatewaySupport;  
  
**public class** SOAPConnector **extends** WebServiceGatewaySupport {  
  
 **public** Object callWebService(String url, Object request){  
 **return** getWebServiceTemplate().marshalSendAndReceive(url, request);  
 }  
}

The Web Service client was implemented as follows:

**package** org.webcurator.soap.client;  
  
**import** org.apache.commons.logging.Log;  
**import** org.apache.commons.logging.LogFactory;  
**import** org.webcurator.core.harvester.agent.HarvestAgent;  
**import** org.webcurator.core.harvester.agent.HarvestAgentStatusDTO;  
**import** org.webcurator.wsclient.HarvestAgentStatus;  
**import** org.webcurator.wsclient.HarvestAgentStatusRequest;  
**import** org.webcurator.wsclient.HarvestAgentStatusResponse;  
  
**public class** HarvestAgentSOAPClient **implements** HarvestAgent {  
 **private static** Log *log* = LogFactory.*getLog*(HarvestAgentSOAPClient.**class**);  
 **private** SOAPConnector **soapConnector**;  
  
 @Override  
 **public** HarvestAgentStatusDTO getHarvestAgentStatus(String jobNumber) {  
 *log*.debug(**"Calling Harvest Agent SOAP Web Client - getHarvestAgentStatus: jobNumber="** + jobNumber);  
 HarvestAgentStatusRequest harvestAgentStatusRequest = **new** HarvestAgentStatusRequest();  
 harvestAgentStatusRequest.setJobNumber(jobNumber);  
 HarvestAgentStatusResponse harvestAgentStatusResponse = (HarvestAgentStatusResponse) **soapConnector**.callWebService(**"http://localhost:8080/wct/service/harvest-agent"**, harvestAgentStatusRequest);  
 *// copy response to dto* HarvestAgentStatus harvestAgentStatus = harvestAgentStatusResponse.getHarvestAgentStatus();  
 HarvestAgentStatusDTO harvestAgentStatusDTO = **new** HarvestAgentStatusDTO(harvestAgentStatus.getJobNumber(),  
 harvestAgentStatus.getStatus(), harvestAgentStatus.getMessage());  
 **return** harvestAgentStatusDTO;  
 }  
  
 **public void** setSoapConnector(SOAPConnector soapConnector) {  
 **this**.**soapConnector** = soapConnector;  
 }  
}

Note that the Web Service client uses the locationUri defined in the WSDL definition.

#### Spring Configuration

The spring configuration below in the wct-core-servlet.xml file defines beans for the client classes and injects a web service client into the UI controller.

<**bean id="harvestAgentSOAPClientController" class="org.webcurator.ui.hibernate.HarvestAgentSOAPClientController" abstract="false" scope="singleton" lazy-init="default" autowire="default"**>  
 <**property name="supportedMethods" value="GET,POST"**/>  
 <**property name="harvestAgentSOAPClient" ref="harvestAgentSOAPClient"**/>  
</**bean**>  
  
<**bean id="harvestAgentJAXBMarshaller" class="org.springframework.oxm.jaxb.Jaxb2Marshaller"**>  
 *<!-- this is the package name specified in the <generatePackage> in the pom.xml -->* <**property name="contextPath" value="org.webcurator.wsclient"**/>  
</**bean**>  
  
<**bean id="harvestAgentSOAPConnector" class="org.webcurator.soap.client.SOAPConnector"**>  
 <**property name="defaultUri" value="http://localhost:8080/wct/service/harvest-agent"**/>  
 <**property name="marshaller" ref="harvestAgentJAXBMarshaller"**/>  
 <**property name="unmarshaller" ref="harvestAgentJAXBMarshaller"**/>  
</**bean**>  
  
<**bean id="harvestAgentSOAPClient" class="org.webcurator.soap.client.HarvestAgentSOAPClient"**>  
 <**property name="soapConnector" ref="harvestAgentSOAPConnector"**/>  
</**bean**>

## Log4j2

The log4j.xml and log4j.dtd files were deleted and a new log4j2.xml file was created:

*<?***xml version="1.0" encoding="UTF-8"***?>*<**Configuration status="WARN"**>  
 <**Appenders**>  
 *<!-- Console Appender -->* <**Console name="Console" target="SYSTEM\_OUT"**>  
 <**PatternLayout pattern="%d{DEFAULT} %-5level [%t] %C{2} (%F:%L) - %msg%n"**/>  
 </**Console**>  
 *<!-- Rolling File Appender -->* <**RollingFile name="GeneralLog"**>  
 <**FileName**>${log4j.dir}/wct-core.log</**FileName**>  
 <**FilePattern**>${log4j.dir}/wct-core.log.%d{yyyy-MM-dd}</**FilePattern**>  
 <**PatternLayout**>  
 <**Pattern**>%d{DEFAULT} %-5level [%t] %C{2} (%F:%L) - %msg%n</**Pattern**>  
 </**PatternLayout**>  
 <**Policies**>  
 <**TimeBasedTriggeringPolicy interval="2" modulate="true"** />  
 </**Policies**>  
 <**DefaultRolloverStrategy max="5"** />  
 </**RollingFile**>  
 </**Appenders**>  
 <**Loggers**>  
 <**Logger name="org.webcurator" level="${log4j.level}"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.hibernate" level="warn"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.springframework" level="warn"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.springframework.security" level="warn"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.springframework.web.servlet" level="fatal"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.springframework.ws" level="warn"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.apache.tiles" level="warn"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Logger name="org.apache.commons" level="warn"**>  
 <**AppenderRef ref="GeneralLog"**/>  
 </**Logger**>  
 <**Root level="warn"**>  
 <**AppenderRef ref="Console"**/>  
 </**Root**>  
 </**Loggers**>  
</**Configuration**>

This log4j2 configuration file defines the same behaviour as the log4j configuration. Note that loggers for org.apache.tiles and org.apache.commons were added to suppress a lot of extraneous logging.