Using OpenRules with RESTful Web Services

By Alex Goldin, alexgoldin@openrules.com

I was tasked to quickly provide a simple example that demonstrates how OpenRules-based decisions can be invoked from RESTful Web Services. Below I will describe what I actually did.

Preparing a Basic RESTful Web Service

First, I selected the simplest RESTful Tutorial that allow you to setup an environment and to quickly build a simple RESTful Web Service called "UserManagement". A Java developer without preliminary knowledge of the REST architecture can follow this tutorial. So, I followed the tutorial that provides all sources, and quickly build an Eclipse project "UserManagement" that supports a list of users on a server and allows a client to add, modify, delete, or get users. The only problem I faced was that my Eclipse installation didn't have "Dynamic Web Project". I followed these instructions to add it.

After building the project, I deployed it as a RESTful service at Apache Tomcat using the Eclipse Export+WAR file. First I tested it from my Chrome browser. Then I followed the tutorial to create and run a Java test "WebServiceTester". Only when everything worked fine, I was ready to add an OpenRules-based Decision to some of user services.

Adding a Java-based condition to the Service

My intention was to modify one of already working services by adding a simple condition that later may be replaced to an OpenRules decision. I selected the service "getUser" implemented in the class "UserDao":

```
public User getUser(int id){
      List<User> users = getAllUsers();
      for(User user: users){
         if(user.getId() == id){
            return user;
         }
      }
      return null;
}
```

I decided to make some user "hidden" and for such users instead of showing their names, show only question marks. To do that, I added a new attribute "hidden" to the class User that already contained Id, Name, and Profession. This String attribute "hidden" may take value "Yes" or "No" (default). I also modified Use's method toString() to show the "hidden" attribute.

For simplicity, I decided to consider a user with profession "developer" as hidden. Here is how I modified the above implementation of the method "getUser":

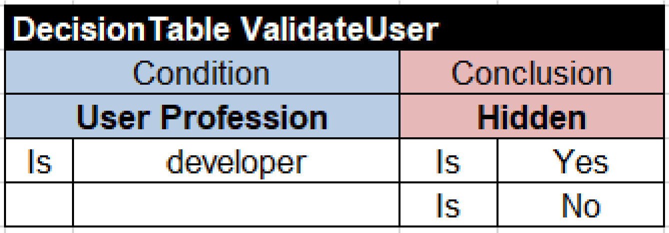
```
public User getUser(int id){
   List<User> users = getAllUsers();

   for(User user: users){
      if(user.getId() == id) {
           if (user.getProfession().equals("developer")) {
                 User hiddenUser = new User(id, "???", user.getProfession());
                 hiddenUser.setHidden("Yes");
                 return hiddenUser;
           }
           return user;
      }
   }
   return null;
}
```

When I redeployed the service again, and the same test produced:

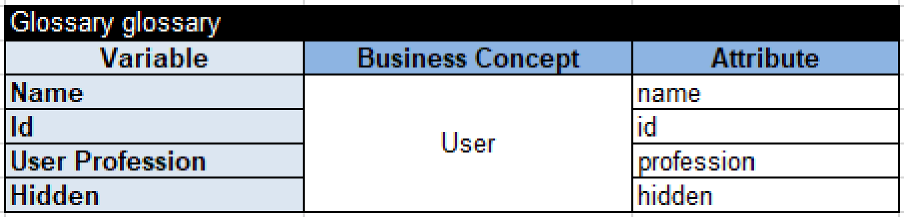
```
Test case name: testGetUser, Result: pass
       User [id=3, name=???, profession=developer, hidden=Yes]
```

Adding OpenRules-based Decision to the Service

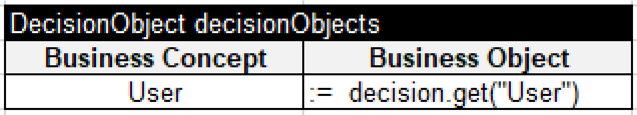
Now, I decided to replace the logic that specifies the hidden user with an OpenRules-based decision. So, I created an Excel file "Decision.xls" in the folder "UserManagement/WebContent/rules" with the following content:



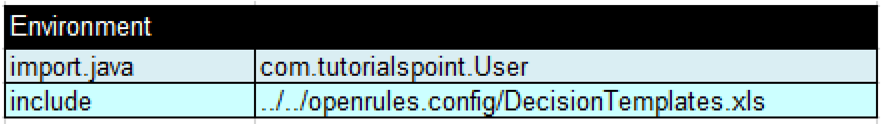
DecisionTable ValidateUser			
Condition		Conclusion	
User Profession		Hidden	
Is	developer	Is	Yes
		Is	No



Glossary glossary		
Variable	Business Concept	Attribute
Name	User	name
Id		id
User Profession		profession
Hidden		hidden



DecisionObject decisionObjects	
Business Concept	Business Object
User	:= decision.get("User")



Environment	
import.java	com.tutorialspoint.User
include	../../openrules.config/DecisionTemplates.xls

Of course, now the logic that defines a hidden user can be made much more sophisticated as it is now externalized from the code.

To invoke this decision, I made the following changes in the method “getUser”:

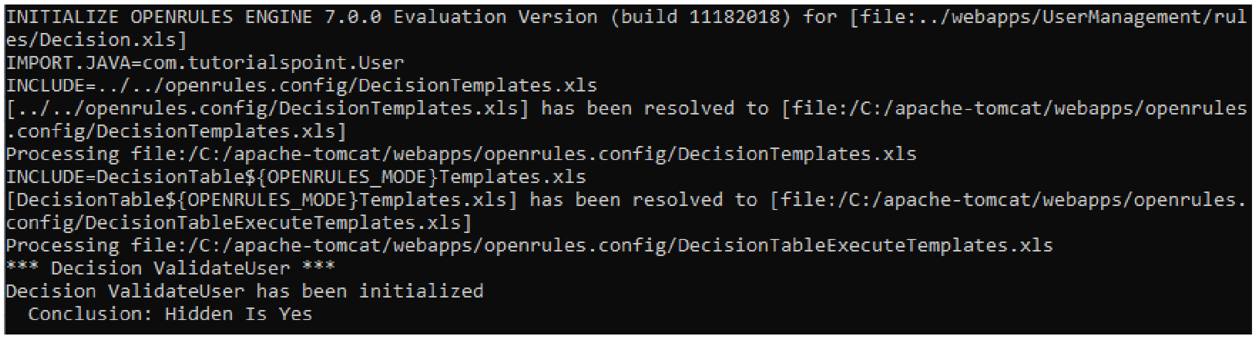
```java
public User getUser(int id){
   List<User> users = getAllUsers();

   for(User user: users){
      if(user.getId() == id) {
         // Use OpenRules to define is a user is hidden
         String excelFile = "file:../webapps/UserManagement/rules/Decision.xls";
         Decision decision = new Decision("ValidateUser",excelFile);
         decision.put("User", user);
         decision.execute();
         if (user.getHidden().equals("Yes")) {
               User hiddenUser = new User(id, "???", user.getProfession());
               hiddenUser.setHidden("Yes");
               return hiddenUser;
         }
         return user;
      }
   }
   return null;
}
```

As you can see, now this method creates an OpenRules decision based on the file `"file:../webapps/UserManagement/rules/Decision.xls"` and executes this decision to decide if a user is hidden or not.

I could not simply redeploy the service, because it still has no idea about OpenRules. So, what should I do to make my RESTful service to be aware of OpenRules? Only two things:

1) Copy all jars from the standard OpenRules configuration folder “openrules.config/lib” to the folder “UserManagement/WebContent/WEB-INF/lib”
2) Copy the entire “openrules.config” to your Tomcat’s webapps folder, so our decision will be able to include standard OpenRules templates from this folder (remember my include “../../openrules.config/DecisionTemplates.xls” in the above Environment table?)

After doing these two “things”, I redeployed my service again and ran the same Java test. While it produced the same results, the server protocol had shown that OpenRules-decision was created and executed:



```
INITIALIZE OPENRULES ENGINE 7.0.0 Evaluation Version (build 11182018) for [file:../webapps/UserManagement/rul
es/Decision.xls]
IMPORT.JAVA=com.tutorialspoint.User
INCLUDE=../../openrules.config/DecisionTemplates.xls
[../../openrules.config/DecisionTemplates.xls] has been resolved to [file:/C:/apache-tomcat/webapps/openrules
.config/DecisionTemplates.xls]
Processing file:/C:/apache-tomcat/webapps/openrules.config/DecisionTemplates.xls
INCLUDE=DecisionTable${OPENRULES_MODE}Templates.xls
[DecisionTable${OPENRULES_MODE}Templates.xls] has been resolved to [file:/C:/apache-tomcat/webapps/openrules.
config/DecisionTableExecuteTemplates.xls]
Processing file:/C:/apache-tomcat/webapps/openrules.config/DecisionTableExecuteTemplates.xls
*** Decision ValidateUser ***
Decision ValidateUser has been initialized
  Conclusion: Hidden Is Yes
```

Naturally, we can add the same or similar OpenRules-based decisions to other services.

Conclusion. OpenRules-based decisions can be invoked from any RESTful service like any other Java program by making its jar-files known to the services. The location of the standard OpenRules templates in Excel files can be easily configured inside the decision Environment table.