| | |
|---|---|
| **Customer name** | Scottish Government (Agriculture, Food & Rural Communities) |
| **Account Number** | 1157386 |
| **Customer Type** | Government |
| **Project Name** | AFRC Futures |

## Customer business problem *( background / context )*

AFRC are responsible for distributing approximately £650m of EU Common Agricultural Payments (CAP) funds every year to Scottish producers / farmers. Faced with a complete overhaul of the CAP framework AFRC decided to build a new fund management system; one that would promote customer self-service and streamline back office processing and provide a fresh new on-line application.

### Requested Feature Enhancement

To provide Oracle UCP support in JBoss EAP 6.1.x

### *Customer* Business Case

AFRC's new on line application ('Futures') replaces an Oracle forms based solution with a self-serve portal that allows farmers and advisory firms to manage their business accounts, apply for grants and manage payments.

Futures a data intensive web application uses Oracle 11g (RAC) as its primary data store. This architectural solution relies on a number of features of the RAC technology including:

Fast Connection Fail over (FCF);
Run-time connection load balancing and
Connection affinity.

The UCP technology provided by Oracle is essential to the delivery of these requirements. In addition UCP will also give a performance boost to the Futures solution.

More specifically supporting Oracle UCP, would allow Futures to leverage the Oracle Notification Service which in turn allows you to enable Fast Connection Fail over and Run-time Load Balancing. When a RAC node goes down, the UCP driver is immediately aware of this by receiving a Fast Application Notification (FAN) event for the node down . The driver is then capable of propagating this event by raising a Java exception Closed Connection (ORA-17008) to all the connections held by the driver that are persistent to this node. For connections actively in use, the application typically handles this situation by re-acquiring new connections from the connection cache. The cache itself will handle unused connections that were affected by this event by rebuilding new connections on the remaining RAC nodes when necessary.

The client have heavily invested in the well established Oracle RAC technology and would like to gain maximum benefits from it.

The Futures solution went live in December 2014 with out any UCP support but is now becoming critical to the success of the program for the following reasons :

- HA implementation
- DR implementation
- System tuning
- System Scalability

Not being able to provide UCP support is now seriously affecting Red Hat's ability to help the client deliver this program.

### Customer perceived benefits

A fault tolerant resilient Futures application

- Lowers cost of administration through the promotion of on-line self-service, better back office staff management and a more powerful rules based approaches to validation and compliance checking.

- Provide a solution architecture that is able to react to CAP rules changes without having to re-engineer the solution through the encapsulation and abstraction of rules and processes.
- Reduce reliance on paper based processing and improve EU audit compliance.

**Red Hat benefits**

Competitor products like Oracle Application Server and IBM Web Sphere already have built in support for Oracle UCP. Providing this support, would give parity in the data intensive enterprise solutions space.

## Red Hat products used

Red Hat technology used on the engagement:

RHEL 6.4
JBoss Portal 6.1.1
JBoss EAP 6.0.1
JBoss Fuse 6.0
JBoss BRMS 5.3.1
JON 3.1.2

**Red Hat contract value**

| | |
|---|---|
| **Total Contract Value** | $7,389M (conversion from £4,667M GBP) T&M |

**Alternative Technical Solutions Explored**

- Oracle UCP on EAP 6.0.1 and EAP 6.1.1

This was explored and tested to see if Oracle UCP could be a viable solution on either of the platforms. This approach allowed us to use jdbc drivers to connect to the oracle database but experienced class loading problems with the Oracle UCP Drivers. Thus this was not a viable solution. The following were the changes that were implemented in standalone.xml

```
<datasources>

<xa-datasource jndi-name="java:/jdbcjcr_portal" pool-name="JCRPortalDS" spy="true">
<xa-datasource-property name="URL"> jdbc:oracle:thin:@//xxxxxxxxx:1521/xxxxxxxxxxxxxx </xa-datasource-property>
<xa-datasource-property name="MinPoolSize"> 10 </xa-datasource-property>
<xa-datasource-property name="MaxPoolSize"> 10 </xa-datasource-property>
<xa-datasource-property name="InactiveConnectionTimeout"> 0</xa-datasource-property>
<xa-datasource-property name="TimeToLiveConnectionTimeout">0</xa-datasource-property>
<xa-datasource-property name="AbandonedConnectionTimeout"> 0</xa-datasource-property>
<xa-datasource-property name="ConnectionWaitTimeout"> 0 </xa-datasource-property>
<xa-datasource-property name="PropertyCycle"> 900</xa-datasource-property>
<xa-datasource-property name="ValidateConnectionOnBorrow"> true</xa-datasource-property>
<xa-datasource-property name="ConnectionPoolName"> JCRPortalDS </xa-datasource-property>
<xa-datasource-property name="FastConnectionFailoverEnabled"> true </xa-datasource-property>
<xa-datasource-property name="ONSConfiguration"> nodes=xxxxxxxx:xxxxxxxxx </xa-datasource-property>
<xa-datasource-property name="ConnectionFactoryClassName"> oracle.jdbc.xa.client.OracleXADataSource </xa-datasource-property>
<driver>oracle-ucp</driver>
<xa-pool>
        <min-pool-size>10</min-pool-size>
        <max-pool-size>10</max-pool-size>
        <is-same-rm-override>false</is-same-rm-override>
        <no-tx-separate-pools>true</no-tx-separate-pools>
</xa-pool>
<security>
        <user-name>xxxxxxxxxxxx</user-name>
        <password>xxxxxxxxxxx</password>
</security>
</xa-datasource>
<drivers>
        <driver name="oracle-ojdbc6" module="com.oracle.ojdbc6">
        <xa-datasource-class>oracle.jdbc.xa.client.OracleXADataSource</xa-datasource-class>
```

```
</driver>
<driver  name="oracle-ucp"  module="com.oracle.ucp">  <xa-datasource-class>oracle.ucp.jdbc.PoolXADataSourceImpl</xa-datasource-class>
</driver>
</drivers>
</datasources>
```

- Programmatic use of Oracle UCP on EAP 6.0.1

In this scenario, there was success but this was limited to areas of the solution where we have programmatic control on the data source and connection pools as elaborated in the example below.

```
import java.sql.ResultSet;
import oracle.ucp.jdbc.PoolDataSourceFactory;
import oracle.ucp.jdbc.PoolDataSource;
import javax.naming.*;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Enumeration;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

Context ctx = new InitialContext();
Context envContext = (Context) ctx.lookup("java:/comp/env");
javax.sql.DataSource ds = (javax.sql.DataSource) envContext.lookup ("jdbc/BRMSPool");
writer.println("Got the datasource");

Connection conn = ds.getConnection();
writer.println("<p>Connected to an Oracle intance</P> ");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("select '********* Instance name: '||sys_context('userenv','instance_name') || CHR(10) || '********* Service
Name: ' || sys_context('userenv','service_name') from dual");

while (rs.next()) {
        writer.println(rs.getString(1));
}
rs.close();
stmt.close();
conn.close();
```

Thus this was not a viable solution for JBoss Portal where programmatic intervention is not an option when connecting to the portal data sources.

**Conclusion**

Oracle UCP is an important technology component in the AFRC Futures application to provide optimal performance and fault tolerant capabilities. Its support in EAP 6.1.1 would be of considerable critical value to the client and Red Hats ongoing partnership with the SI on site and the client.