HP OpenView Select Identity

External Call Developer Guide

Software Version: 3.0

July 2004

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

This product includes software developed by the Apache Software Foundation (http://www.apache.org/). Portions Copyright (c) 1999-2003 The Apache Software Foundation. All rights reserved.

Select Identity uses software from the Apache Jakarta Project including:

- Commons-beanutils.
- Commons-collections.
- Commons-logging.
- Commons-digester.
- Commons-httpclient.

- Element Construction Set (ecs).
- Jakarta-poi.
- Jakarta-regexp.
- Logging Services (log4j).

Additional third party software used by Select Identity includes:

- JasperReports developed by SourceForge.
- iText (for JasperReports) developed by SourceForge.
- BeanShell.
- Xalan from the Apache XML Project.
- Xerces from the Apache XML Project.
- Java API for XML Processing from the Apache XML Project.
- SOAP developed by the Apache Software Foundation.
- JavaMail from SUN Reference Implementation.
- Java Secure Socket Extension (JSSE) from SUN Reference Implementation.
- Java Cryptography Extension (JCE) from SUN Reference Implementation.
- JavaBeans Activation Framework (JAF) from SUN Reference Implementation.
- OpenSPML Toolkit from OpenSPML.org.
- JGraph developed by JGraph.
- Hibernate from Hibernate.org.

Trademark Notices

HP OpenView Select Identity is a trademark of Hewlett-Packard Development Company, L.P.

Microsoft, Windows, the Windows logo, and SQL Server are trademarks or registered trademarks of Microsoft Corporation.

Sun™ workstation, Solaris Operating Environment™ software, SPARCstation™ 20 system, Java technology, and Sun RPC are registered trademarks or trademarks of Sun Microsystems, Inc. JavaScript is a trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

This product includes the Sun Java Runtime. This product includes code licensed from RSA Security, Inc. Some portions licensed from IBM are available at http://oss.software.ibm.com/icu4j/.

IBM, DB2 Universal Database, DB2, WebSphere, and the IBM logo are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

This product includes software provided by the World Wide Web Consortium. This software includes xml-apis. Copyright © 1994-2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. http://www.w3.org/Consortium/Legal/

Intel and Pentium are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

AMD and the AMD logo are trademarks of Advanced Micro Devices, Inc.

BEA and WebLogic are registered trademarks of BEA Systems, Inc.

VeriSign is a registered trademark of VeriSign, Inc. Copyright © 2001 VeriSign, Inc. All rights reserved.

All other product names are the property of their respective trademark or service mark holders and are hereby acknowledged.

Support

Please visit the HP OpenView web site at:

http://openview.hp.com/

There you will find contact information and details about the products, services, and support that HP OpenView offers.

You can go directly to the support web site at:

http://support.openview.hp.com/

The support web site includes:

- Downloadable documentation
- Troubleshooting information
- Patches and updates
- Problem reporting
- Training information
- Support program information

contents

1

Introduction

HP OpenView Select Identity supports the ability to invoke calls to external systems. Specifically, you can use external calls to perform the following:

- Value generation — generates the values of an attribute
- Value constraint — provides a list of possible values for an attribute
- Value validation — validates the value of an attribute
- Workflow action — performs a task as part of a workflow, enabling you to integrate approval processes with external processes and systems

You must code the classes that are called by external calls using the External Call API and Workflow API. After you create the Java file(s) that comprise an external call, you can register it with Select Identity through the External Calls home page on the Select Identity client. Refer to the *HP OpenView Select Identity Administrator Guide* for more information.

The Select Identity External Call and Workflow APIs define a Java-based interface for creating external callouts. Although the Select Identity-facing portion of the interface must be Java, it can be a “wrapper” for a program written in any language.

For workflow external calls, the APIs support synchronous communication. Select Identity requires the external system to complete its processing and provide status information as part of the callout, which is required to return status that indicates how Select Identity will proceed with the workflow.

Overview of the APIs

Creating an external call entails coding one or more Java classes (Java 2 Platform, Enterprise Edition). Thus, you must have an understanding of the Java Developer Kit (JDK), version 1.4. For information about the J2EE APIs, refer to **http://java.sun.com/j2se/1.4.2/docs/api/index.html**.

The following classes and interfaces are provided by the External Call API and are available for creating value external calls:

- **TAValueContraintIntf**

  Defines classes that return a list of value constraints. External calls that generate possible attribute values must implement this interface.

- **TAValueConstraintIntf.TAValueConstraintBeanIntf**

  Contains the constrained values. External calls the constrain attribute values should import this interface.

- **TAValueGenerationIntf**

  Defines a class that generates the value of an attribute. External calls that generate values must implement this interface.

- **TAValueValidationIntf**

  Defines a class that validates the value of an attribute. External calls that validate values must implement this interface.

- **TAAttributeDefinitionException** and **TAAttributeValueValidationException**

  Exceptions defined for use by external calls.

The following classes and interfaces are provided by the External Call API for use in workflow external calls:

- **IWfClient**

  Used to invoke a workflow template, thereby creating a workflow instance. This interface also enables you to resume a passivated workflow instance and to terminate an instance.

- **IWfQuery**

  Retrieves runtime and configuration information about a workflow at the template, instance, block, or activity level.

- **IWfDataUpdate**

  Updates workflow variables.

- **StatelessServiceObjectFactory**

  Creates a component that implements the IWfClient, IWfQuery, and IWfDataUpdate interfaces. It forces the interfaces to behave like stateless EJB components while hiding EJB-specific code and deployment information from component developers.

- **WfExternalCall**

  Provides the contract between Select Identity and an external stage in a workflow. Select Identity can invoke classes implementing this interface to perform actions in a workflow. External calls that are invoked by workflow instances must implement this interface.

- **WfExternalCallException**

  Exception defined for use by the Workflow API.

- **WfExternalCallStatus**

  Returns the status of an approval stage to Select Identity. Along with the status of the stage, it can also return changes to the user profile including attributes and entitlements.

The following classes may be used to further explore the Select Identity's request and approval frameworks:

- **AttributeRecord**

  Represents an attribute

- **ChangeRecord**

  Represents a change in a attribute. Objects of this class can be used to communicate changes in a user profile from external calls.

- **Request**

  Defines methods that retrieve and set all information related to an incoming request in the Select Identity system.

- **RequestJobItem**

  Defines the job that handles the request in the Select Identity system.

- **RequestTarget**

  Defines the target of a request.

- **RequestTargetParam**

  Defines the parameters of a request target.

- **RequestTargetParamValue**

  Enables you to set and get parameter values.

- **TAFilter**

  A general class that stores filter criteria for a selection (search) procedure.

- **TARequestAction**

  Represents the action to take place on the target resource.

- **TARequestEvent**

  Represent the event for the request.

- **TARequestType**

  Represents the type of request.

Creating an External Call

The sections provided here describe how to implement and compile an external call of each type.

Value External Calls

To create an external call, follow these guidelines:

- Value Generation external calls must implement the `TAValueGenerationIntf` interface. See Value Generation External Call on page 19 for code examples.
- Value Constraint external calls must implement the `TAValueConstraintIntf` interface. See Value Constraint External Call on page 22 for code examples.
- Value Validation external calls must implement the `TAValueValidationIntf` interface. See Value Validation External Call on page 25 for code examples.

In addition, the following provides general information about how to obtain to input parameters, Select Identity attributes, and what needs to be returned:

- To retrieve an External Call parameter that is defined in the ExternalCall registration page, use the **containsKey** and **get** methods, as in this example:

```
if (attrs.containsKey("parametername"))
   String parameter = (String)
attrs.get("parametername")
```

- For Value Validation external calls, throw **TAAttributeValueValidationException** if the external call is unsuccessful.
- Compile the Java class(es) you create using the following command:

```
javac -cp clientintf.jar <files>
```

Once you create the external call, copy the class files and any associated file(s) to a directory on the Select Identity server. Copying the files to a directory in the Select Identity installation path will save you a step in the deployment procedure. Note that classes implementing external call interfaces cannot reside in the application server's classpath. Instead, specify the location in the classpath of the external call definition.

Also, external calls can use any logging mechanism. The examples shown in this book use an internal logging mechanism that is not available externally.

Workflow External Calls

The following provides general information about how to implement a workflow external call, how to obtain to input parameters, Select Identity attributes, and what needs to be returned:

- Workflow external calls must implement the `WfExternalCall` interface.
- The main method of the external call must be called `process()`. This method expects four input parameters. Here is the call signature of the process() method:

```
WfExternalCallStatus process(String stageId,
  RequestTarget requestTarget,
  AttributeRecord [] availGrp,
```

```
  AttributeRecord [] availRole,
  AttributeRecord [] availEntilements,
  Map map) throws WfExternalCallException
```

- The workflow external call must return `WfExternalCallStatus`. In addition, when an external call returns information, it must return data that is valid in Select Identity.
- To retrieve an external call parameter that is defined in the ExternalCall action in a workflow template, use the **containsKey** and **get** methods, as in this example:

```
if (attrs.containsKey("parametername"))
   String parameter = (String)
attrs.get("parametername")
```

- Select Identity defines the following standard parameters in the map:
  - WfExternalCall.WF_PARAM_SERVICENAME — for the service name. This is a String object.
  - WfExternalCall.WF_PARAM_ADMINUSERID — for the requestor's user name. This is a String object.
  - WfExternalCall.WF_PARAM_REQUESTID — for the request identifier. This is an Integer object.
  - WfExternalCall.WF_PARAM_WORKFLOWINSTID — for the workflow instance ID. This is an Integer object.
- Retrieve Select Identity attributes using one of the get methods, such as **getSingleRequestParamStr**, provided by the RequestTarget class. Here is an example to retrieve a single-value attribute:

```
String attributeValue=(String)requestTarget.
  getSingleRequestParamStr("attributename");
```

  To retrieve the values of a multi-value attribute, use the following method:

```
Set attributeValues=(Set)requestTarget.
  getRequestParam("attributename").getRequestTArgetParamValue()
```

- To retrieve workflow variables, use the IWfQuery interface as in the following example:

```
IWfQuery query =(IWfQuery)StatelessServiceObjectFactory.
  create(IWfQuery.class);
```

```
int instanceId = query.
  getInstanceInfoByInstActivityId(instActivityId).getInstId();
Map varMap = query.getCurrentVariableMap(instanceId);
String comment = (String) varMap.get("$ApproverComments");
```

- Set a workflow variable as follows:

```
IWfDataUpdate du =(IWfDataUpdate)StatelessServiceObjectFactory.
     create(IWfDataUpdate.class);
du.setWorkflowVar(instId, "workflowvariablename", "value");
```

Then, update a Select Identity attribute as follows:

```
WfExternalCallStatus ecs = new WfExternalCallStatus(stageId);
ChangeRecord changeRecord = new ChangeRecord();

// Update the attribute in the change record
changeRecord.setName("attributename");
changeRecord.addValue("newattributevalue");

ecs.addAttributeChange(changeRecord);
```

Set the return status of a workflow external call as follows:

```
ecs = new WfExternalCallStatus(stageId);
```

If the call returns successfully:

```
ecs.setStatus(WfExternalCallStatus.STATUS_APPROVED);
```

If the call is unsuccessful:

```
ecs.setStatus(WfExternalCallStatus.STATUS_REJECT_TERMINATE);
```

You can also update workflow variables that need to be persisted using setStatus(). All variable names that starts with $ is assumed to be a workflow variable and persisted when the call returns.

- You can use any of the following variables in your external call:

Variable Name	Description
_joinCommand	Passed into the workflow by an external application that is invoked as an action in a workflow template activity. This variable is not persistent. Possible values include • **exit** — unconditionally exits the current block • **exitAll** — unconditionally exits the current block and all parent blocks • **reset** — resets the value of the joinCount property set in the block For example, the application could pass this variable to the workflow to instruct it to exit a block unconditionally (even if, for instance, the _joinCount value is not met).
_pushVar	Passed into the workflow by an external application that is invoked as an action in a workflow template activity. The object is then added to workflow's internal pushList block variable. The pushed objects in the list can be displayed later in a tabular format in a report. You can specify any Java object as the value of this variable.
_instActivityId	An internal variable representing the instance activity ID for the current activity in execution. This variable is maintained by the engine and cannot be updated in template. It is assigned an integer value.
_activityId	The activity ID string for current activity in execution. This variable is maintained by the engine and cannot be updated in template. A string value is assigned to this variable.

Variable Name	Description
$_blockId	The block ID for the current block. You can assign any string value to this variable.
$_instId	The workflow instance ID. This variable is maintained by the engine and cannot be updated. An integer value is assigned to this variable.

- Compile the Java class(es) you create using the following command:

  javac -cp clientintf.jar <*files*>

Once you create the external call, copy the class files and any associated file(s) to a directory on the Select Identity server. Copying the files to a directory in the Select Identity installation path will save you a step in the deployment procedure. Note that classes implementing external call interfaces cannot reside in the application server's classpath. Instead, specify the location in the classpath of the external call definition.

Also, external calls can use any logging mechanism. The examples shown in this book use an internal logging mechanism that is not available externally.

2

Examples

This chapter provides an example for each type of external call. Refer to the Javadoc in the `docs/api_help/external_calls` and `workflow` directories on the Select Identity CD for information about the APIs.

Value Generation External Call

This class generates the password from a Social Security Number or ID. The configuration arguments to the function are the SSN_FIELD and PERSONNUMBER_FIELD attributes.

```
package com.trulogica.truaccess.externalcall.generatefunction;

import java.util.*;

import com.trulogica.truaccess.attribute.TAValueGenerationIntf;
import com.trulogica.truaccess.attribute.exception.*;
import com.trulogica.truaccess.attribute.concero.ejb.*;
import com.trulogica.truaccess.attribute.model.*;
import com.trulogica.truaccess.wfengine.wfexternalcall.*;
import com.trulogica.truaccess.externalcall.model.*;
import com.trulogica.truaccess.externalcall.ejb.*;
import com.trulogica.truaccess.externalcall.util.*;
import com.trulogica.truaccess.externalcall.*;
import com.trulogica.truaccess.request.model.*;
```

```
import com.trulogica.truaccess.util.locator.ServiceLocator;
import com.trulogica.truaccess.util.TAClassLoader;
import com.trulogica.truaccess.attribute.TAValueConstraintIntf.
  TAValueConstraintBeanIntf;

public class PasswordFromSSNorPN implements TAValueGenerationIntf
{
  private TAExternalCallBI mExtCallEJB = null;
  private static final String SSN_FIELD="ssnfield";
  private static final String PERSONNUMBER_FIELD="personnumberfield";
  private static final String PASSWORD_SUFFIX="p2wd";

  public Object generateValue(String attribName, RequestTarget reqTarget,
  Map args) throws TAAttributeDefinitionException
  {
    try {
      String ssnField = (String)args.get(SSN_FIELD);
      if (null == ssnField) {
        throw new Exception("Unable to find the argument:"+SSN_FIELD);
      }

      String personNumberField = (String)args.get(PERSONNUMBER_FIELD);
      if (null == personNumberField) {
        throw new Exception("Unable to find the
         argument:"+PERSONNUMBER_FIELD);
      }

      String password = "";
      String ret = getLast4Chars(reqTarget, ssnField);
      if (ret.length() != 0) {
        password = PASSWORD_SUFFIX + ret;
        return new UserNameValueBean(password);
      }

      ret = getLast4Chars(reqTarget, personNumberField);
      if (ret.length() == 0) {
        throw new Exception("Unable to generate password");
      }

      password = PASSWORD_SUFFIX + ret;
      return new UserNameValueBean(password);

    } catch (Throwable t) {
       attribute:"+attribName+”:”+t.getMessage());

      TAAttributeDefinitionException exp =  new
       TAAttributeDefinitionException("Unable to generate attribute value
       for attribute:"+attribName);

      exp.setCausedBy(t);
```

```
      throw exp;
    }

  }

  private String getLast4Chars(RequestTarget reqTarget, String attrName)
   throws Exception
  {
    String value = reqTarget.getParamValueString(attrName);
    if (value == null)
    return "";

    value = value.trim();
    if (value.length() < 4)
    return "";

    return value.substring(value.length() - 4);
  }
}



class UserNameValueBean implements TAValueConstraintBeanIntf
{
  String value = null;
  public UserNameValueBean(String _value)
  {
    value = _value;
  }
  public String getName() {
    return null;
  }

  public Object getValue() {
    return value;
  }

}
```

Value Constraint External Call

The `SearchTable.java` file defines a class that implements a simple table-based lookup of possible constraint values. It uses the external arguments poolname and SQL string to query the database tables.

```
package com.trulogica.truaccess.externalcall.constraintfunction;

import java.util.*;

import com.trulogica.truaccess.base.TAFilter;
import com.trulogica.truaccess.attribute.TAValueConstraintIntf;
import com.trulogica.truaccess.attribute.exception.
  TAAttributeDefinitionException;

import com.trulogica.truaccess.externalcall.constraintfunction.
  SearchResult;

import com.trulogica.truaccess.util.Tools;
import com.trulogica.truaccess.util.DBTool;
import javax.naming.InitialContext;
import javax.sql.DataSource;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.Types;

public class SearchTable implements TAValueConstraintIntf {

  public static final String PARAM_POOLNAME = "poolname";
  public static final String PARAM_SQLSTRING = "query";
  public static final String PARAM_VALUEFIELD = "valuefield";

  private String poolname = null;
  private String query = null;
  private String valuefield = null;

  /**
   * Returns a List of all possible values.
   * Since this list can be very large it should be used sparingly.
   */

  public List getValueConstraint(String attribName, Map args)
    throws TAAttributeDefinitionException
  {
    return getValueConstraint(attribName,null,args);
  }
```

```
public List getValueConstraint(String attribName,TAFilter filter,
  Map args) throws TAAttributeDefinitionException
{
  Connection connection = null;
  PreparedStatement pStmt = null;
  ResultSet rs = null;
  ArrayList ar = new ArrayList();
  try {
    poolname = (String)args.get(PARAM_POOLNAME);
    query = (String)args.get(PARAM_SQLSTRING);
    valuefield = (String)args.get(PARAM_VALUEFIELD);

    checkParam(poolname,PARAM_POOLNAME);
    checkParam(query,PARAM_SQLSTRING);
    checkParam(valuefield,PARAM_VALUEFIELD);

    String modFilter = "";
    if (null != filter) {

      switch (filter.getOperation()) {
        case TAFilter.EQUALITY:
          modFilter = "  "+valuefield+" = ?";
          break;
        case TAFilter.BEGINS_WITH:
          modFilter = " "+valuefield+" LIKE ?";
          filter.setValue(filter.getValue()+"%");
          break;
        case TAFilter.ENDS_WITH:
          modFilter = " "+valuefield+" LIKE ?";
          filter.setValue("%"+filter.getValue());
          break;
        case TAFilter.CONTAINS:
          modFilter = " "+valuefield+" LIKE ?";
          filter.setValue("%"+filter.getValue()+"%");
          break;
      }

      query = query.toUpperCase();

      if (modFilter.length() > 0) {
        if (query.indexOf(" WHERE ") > 0) {
          query = query + " AND " + modFilter;
        }
      }
    }
    connection = this.getConnection();

    pStmt = connection.prepareStatement(query);
```

```
    if (modFilter.length() > 0) {
      pStmt.setString(1,filter.getValue());
    }

    rs = pStmt.executeQuery();
    ResultSetMetaData rsMetaData = rs.getMetaData();
    int nameColumn=0;
    //Select the first String field whose column name does not match
    // the value field
    for (int i = 1; i <= rsMetaData.getColumnCount();i++)
    {
      if (((rsMetaData.getColumnType(i)==Types.VARCHAR)||
        (rsMetaData.getColumnType(i)==Types.CHAR))
        && (!rsMetaData.getColumnName(i).equalsIgnoreCase(valuefield))) {
        nameColumn=i;
        break;
      }
    }

    while (rs.next())
    {
      SearchResult sr = new SearchResult();

      sr.setValue(rs.getString(valuefield));

      if (nameColumn>0)
        sr.setName(rs.getString(nameColumn));
      else
        sr.setName((String) sr.getValue());

      ar.add(sr);
    }

    return ar;
  } catch (Exception e) {
    TAAttributeDefinitionException exp =
      new TAAttributeDefinitionException();
    exp.setCausedBy(e);
    throw exp;
  } finally {
    DBTool.close(rs);
    DBTool.close(pStmt);
    DBTool.close(connection);
  }
}

private Connection getConnection() throws Exception
{
  InitialContext ctx = null;
```

```
    try {
      ctx = new InitialContext();
      DataSource ds = (DataSource)ctx.lookup(poolname);
      return ds.getConnection();
    } finally {

      Tools.close(ctx);
    }
  }

  private void checkParam(String value,String name) throws Exception
  {
    if ((null == value)||(value.trim().length() == 0))
    throw new Exception("The value of "+name+" is needed");
  }
}
```

Value Validation External Call

The `IsAlphaNumeric.java` file defines a class that determines whether the value is an alphanumeric.

```
package com.trulogica.truaccess.externalcall.validation;
import com.trulogica.truaccess.attribute.TAValueValidationIntf;
import com.trulogica.truaccess.attribute.exception.
  TAAttributeValueValidationException;
import java.util.*;
public class IsAlphaNumeric implements TAValueValidationIntf
{
  public void validateValue(String attribName,Object value, Map args)
    throws TAAttributeValueValidationException
  {
    String password = (String ) value;
    boolean containsDigit = false;
    boolean containsLetter = false;
    for (int i = 0, n = password.length(); i < n; i++)
    {
      // checkNum(word.charAt(i));
      if (Character.isDigit(password.charAt(i)))
      {
        containsDigit = true;
        break;
      }
    }
```

```
    for (int i = 0, n = password.length(); i < n; i++)
    {
      if (Character.isLetter(password.charAt(i)))
      {
        containsLetter = true;
        break;
      }
    }

    if( !containsLetter || !containsDigit )
    {
      throw new TAAttributeValueValidationException("The password is not
      alphanumeric. Password has to contain at least one numeric and one
      non-numeric field");
    }
  }
}
```

Workflow External Call

The following files comprise an example of an external call that can be used by a workflow template. The `WorkflowStepSample.java` file generates the new attribute value. The `PersonNumberCallOut.java` file defines a class that obtains the name of the user.

WorkflowStepSample.java

This class simulates an external call step. It collects information from the existing request and concatenates it to produce a new attribute value.

```
package com.trulogica.truaccess.wfenginesvcs.wfexternalcall.support;

import com.trulogica.truaccess.wfengine.wfexternalcall.WfExternalCall;
import com.trulogica.truaccess.wfengine.wfexternalcall.AttributeRecord;
import com.trulogica.truaccess.wfengine.wfexternalcall.
  WfExternalCallStatus;
import com.trulogica.truaccess.wfengine.wfexternalcall.
  WfExternalCallException;
import com.trulogica.truaccess.request.model.RequestTarget;
import com.trulogica.truaccess.request.model.RequestTargetParam;
import com.trulogica.truaccess.request.model.RequestTargetParamValue;
import com.trulogica.truaccess.wfengine.wfexternalcall.ChangeRecord;
import com.trulogica.truaccess.base.constants.UserAttributeConstants;
```

```
import java.util.*;
//import com.trulogica.truaccess.workflow.external.util.*;

public abstract class WorkflowStepSample implements WfExternalCall
{
  String attribName = "";

  protected WorkflowStepSample(String _attribName)
  {
    attribName = _attribName;
  }

  public WfExternalCallStatus process(String stageId,
    RequestTarget requestTarget,
    AttributeRecord [] availGrp,
    AttributeRecord [] availRole,
    AttributeRecord [] availEntilements,
    Map map) throws WfExternalCallException
  {
    try{

      WfExternalCallStatus ecs = new WfExternalCallStatus(stageId);
      String propPrepName = "", propName = "";

      propName = attribName + ".attributes";

      String attrs = System.getProperty(propName);
      if (null == attrs) {
        ecs.setStatus(WfExternalCallStatus.STATUS_REJECT_TERMINATE);
        return ecs;
      }

      StringBuffer sb = new StringBuffer();
      ChangeRecord changeRecord = new ChangeRecord();
      changeRecord.setName(attribName);

      StringTokenizer st = new StringTokenizer( attrs,"," );
      while( st.hasMoreTokens() ) {
        String  attrName = st.nextToken();
        String value = requestTarget.getParamValueString(attrName);
        if (null == value) {
          throw new Exception("Unable to generate the value for
            attribute:"+attrName);
        }

        sb.append(value);
      }

      changeRecord.addValue(sb.toString());
      ecs.addAttributeChange( changeRecord );
```

```
        getSingleRequestParamStr(UserAttributeConstants.FIRST_NAME );

      ecs.setStatus(WfExternalCallStatus.STATUS_APPROVED);
      return ecs;
    }

    catch( Exception e ) {
      throw new WfExternalCallException( e );
    }
  }
}
```

PersonNumberCallOut.java

```
package com.trulogica.truaccess.wfenginesvcs.wfexternalcall.support;

import com.trulogica.truaccess.wfenginesvcs.wfexternalcall.support.
  WorkflowStepSample;

public class PersonNumberCallout extends WorkflowStepSample
{
  public PersonNumberCallout() {
    super("personId");
  }
}
```