SAP BOPC – TDI Integration

Applies to:

SAP Business Objects Planning and Consolidation and IBM Tivoli Identity Management Solutions Integration

Summary

This paper will serve as a high level overview of the integration of IBM Tivoli Identity Manager into a SAP BusinessObjects Planning and Consolidation Solution

Author: Daniel Jacinto

Company: SAP South Africa Pty LTD

Created on: February 27, 2012

Table of Contents

Executive Summary

This document will serve to provide an overview of the integration between the SAP BusinessObjects Planning and Consolidation solution with IBM Tivoli Identity Manager.

In today's age of ever increasing IT systems, the need for a central identity management solution becomes imperative in order to minimize the complexity and total cost of ownership of the different solutions within a client's IT landscape. In certain client landscapes, due to the strict security requirements, the need arose for the ability of the current identity management solution to be able to provision and manage the user access to the SAP BusinesObjects Planning and Consolidation system in a centralized and monitored and auditable way.

The SAP BusinessObjects Planning and Consolidation system is based on the Microsoft version and uses Microsoft Active Directory or SAP BusinessObjects CMS for its user authentication. The user access is managed through the administration console from SAP BusinessObjects Planning and Consolidation. Unfortunately the user administration for SAP Business Objects Planning and Consolidation is separate from the identity management provisioning process. Due to this the need for a central single identity management process has brought about the need for custom development in order to integrate the SAP BusinessObjects Planning and Consolidation solution with the IBM Tivoli Identity Management solution.

This paper will detail the integration between the two solutions and the different mechanisms in which the integration will occur. It will also detail the known limitations and the source code along with comments regarding the different mechanisms.

Technical Overview

In this section it will detail the basic technical overview of how the solutions will integrate through the use of custom development. In SAP BusinessObjects Planning and Consolidation the security process is managed through the concept of user authentication and user access.

In SAP BusinessObjects Planning and Consolidation the user authentication is managed through different available mechanisms. It can either use BusinessObjects CMS authentication in which the user authentication is managed through a BusinessObjects Enterprise instance, or it can make use of Windows authentication.

SAP BusinessObjects Planning and Consolidation does not store any reference to the user's password, either encrypted or unencrypted. It stores reference to the actual user name in order to verify that it is a valid user in the application set.

In order to verify that the users are valid to an application set, it will store reference to the user's details. For example: In the system database it will store the username ***JacintoD*** where it is using CMS authentication or ***DOMAIN\JacintoD*** if it is using Windows Authentication.

In SAP BusinessObjects Planning and Consolidation the concept of what the user can do within the application is controlled through the Task Profile, what the user can see in terms of data is managed through the use of member access profile. These security mechanisms are managed through the administration console of the application. Managing these security concepts is not planned for the level of integration between the different solutions.

In the first phase, the level of integration is meant to only be able to provision and decommission a user using the IBM Tivoli Identity Manager solution in SAP BusinessObjects Planning and Consolidation.

This will be done using custom development using a SQL Stored Procedure which will reside on the SAP BusinessObjects Planning and Consolidation SQL Server. This stored procedure will be the bridge between the two solutions and provide the mechanism in which the user will be either be provisioned or decommissioned with an application set, thus allowing them to login into the application set or not be able to login into an application set.

The SQL Stored procedure will insert the necessary username values in the necessary tables and associate them with a team within the SAP BusinessObjects Planning and Consolidation application set. Once the team name has been retrieved and stored, the user will have to be associated with that team when the user is provisioned or decommissioned. It is not recommended to manage the teams, task profiles or member access profiles outside of the SAP BusinessObjects Planning and Considtaion administration console. There is too many risks associated with this and represents a major challenge if the process is not managed correctly and in accordance with best practices.

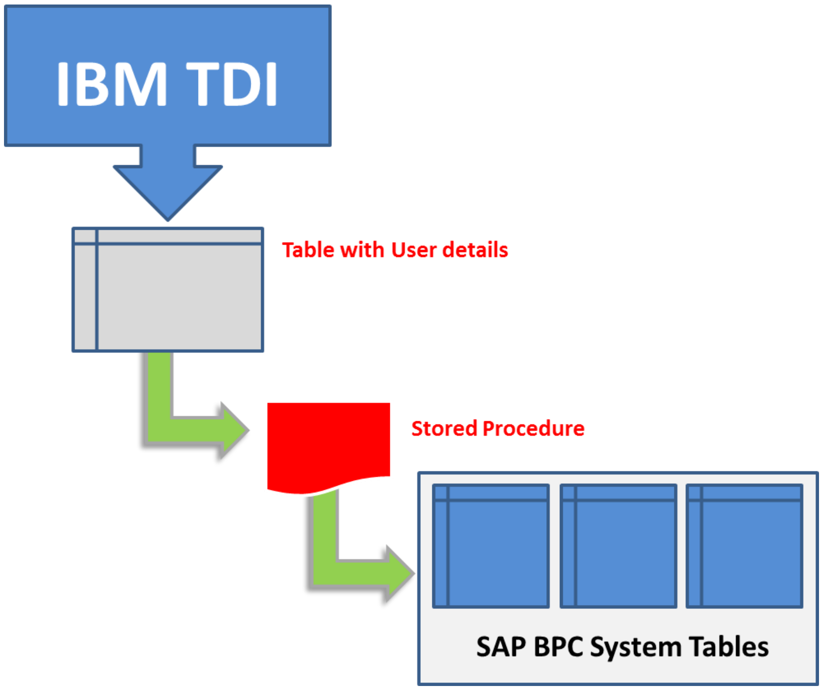
The stored procedure will be from the IBM Tivoli Identity Manager solution. The stored procedure will called using JDBC, and based on the return codes of the stored procedure, it will determine whether or not it was successful or not. Based on the return codes, the various actions can be defined and managed through a separate business process.

Introduction

This solution will comprise of the following components: The SQL code which is responsible for the provisioning and decommissioning of users in the SAP BusinessObjects Planning and Consolidation solution and the IBM Tivoli Directory Integrator project.

This document will be broken down into the SAP BusinessObjects Planning and Consolidation configuration referred to as SAP BPC configuration and IBM Tivoli Directory Integrator referred to IBM TDI project. It will detail a process flow of how the integration occurs and will detail some key areas which need to be understood in the source code.

At a high level the process flow of the integration will occur in the following process:



The IBM TDI Assembly line will read the values from a flat file in this example and then insert the values into the SQL table *tblTIMUsers.* The SQL Stored procedure *sp_bopc_tim* will then read the values in the table and insert or remove the values from the relevant SAP BPC system tables.

The action defined in the *Action* column will define what needs to be done to the user object. For example: If the action is Del, then the user will be decommissioned from SAP BPC and removed from the relevant system tables, conversely this will occur if the action ‘Add’ to add the user to SAP BPC. Please see **Figure 1** for an example of the values stored in the SQL table *tblTIMUsers*

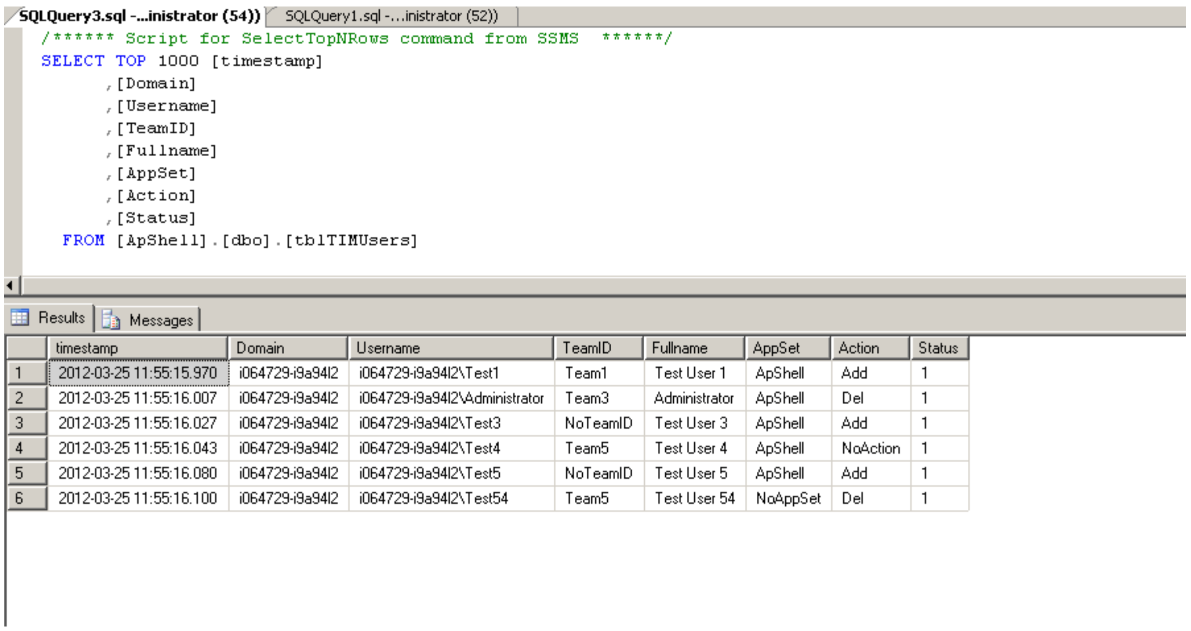
Key Features:

- **Transactional Support**, the stored procedure will group the provisioning and decommisiong or user objects in a transaction. This will ensure that in the event of any issues or errors which might occur, a rollback will not commit the changes. This will ensure the integrrity of the SAP BPC solution.
- **Full auditability**, the stored procedure will detail each step in the *tblTIMLog* along with the SQL being generated. This will ensure that there is a an audit track for the IT auditors and in the event of any issues it can be used to debug and troubleshoot issues
- **Ability to do mass provisioning and decommisioning of user objects,** because the the user details are stored in the *tblTIMUsers* table. The stored procedure will go through the records in the table with the status of a new record (STATUS=1) and perform the neccesary actions defined. This is particulary useful because it means that the SQL stored procedure will not have to be called each time for the user in order to perform the desired action.

  For example: The code has been written to run once on mulitple users, and not be called for each user. Regardless of how many users have been provisioned or decommisioned, the stored procedure will only have to run once.
- **Error Checking**, the stored procedure will do error checking and some pre-emptive checks before executing the action defined.

  For example: The code will check to see if the user exists, if it does, no action will be taken. Additionally it will check the existence of the teams, if the team being defined doesn't exist, the user will not be provisioned.

Figure 1: Screenshot of SQL table tblTIMUsers


SQLQuery3.sql -...inistrator (54))
SQLQuery1.sql -...inistrator (52))
Results
Messages

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [timestamp]
      ,[Domain]
      ,[Username]
      ,[TeamID]
      ,[Fullname]
      ,[AppSet]
      ,[Action]
      ,[Status]
  FROM [ApShell].[dbo].[tblTIMUsers]
```

	timestamp	Domain	Username	TeamID	Fullname	AppSet	Action	Status
1	2012-03-25 11:55:15.970	i064729-i9a94l2	i064729-i9a94l2\Test1	Team1	Test User 1	ApShell	Add	1
2	2012-03-25 11:55:16.007	i064729-i9a94l2	i064729-i9a94l2\Administrator	Team3	Administrator	ApShell	Del	1
3	2012-03-25 11:55:16.027	i064729-i9a94l2	i064729-i9a94l2\Test3	NoTeamID	Test User 3	ApShell	Add	1
4	2012-03-25 11:55:16.043	i064729-i9a94l2	i064729-i9a94l2\Test4	Team5	Test User 4	ApShell	NoAction	1
5	2012-03-25 11:55:16.080	i064729-i9a94l2	i064729-i9a94l2\Test5	NoTeamID	Test User 5	ApShell	Add	1
6	2012-03-25 11:55:16.100	i064729-i9a94l2	i064729-i9a94l2\Test54	Team5	Test User 54	NoAppSet	Del	1

SAP BusinessObjects Planning and Consolidation Configuration

In order for the integration to occur, the following objects need to be created in the destination SAP BPC application set: The SQL tables and the stored procedure which contains all of the logic which will perform the necessary integration and actions.

The following tables need to be created in the SAP BPC application set database: *tblTIMUsers* and *tblTIMLog*. The table *tblTIMUsers* will contain the imported users from the IBM TDI assembly line. The *tblTIMLog* will be used for troubleshooting purposes. It will contain information which can be used for auditing purposes and debugging any possible issues.

Step 1: Create the *tblTIMUsers* SQL table
You can run the following piece of SQL Script in order to create the SQL table *tblTIMUsers*.
Or alternatively you can run the SQL script ***create_table_sql.sql.*** If you have run the commands in this file, you can skip **Step 2** and proceed to **Step 3**
This table will contain the values that the stored procedure will use to either provision or decommission the user in SAP BPC. Without this table the stored procedure will not work and thus the integration will not occur.

```
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[tblTIMUsers](
    [timestamp] [datetime] NOT NULL,
    [Domain] [nvarchar](50) NOT NULL,
    [Username] [nvarchar](50) NOT NULL,
    [TeamID] [nvarchar](50) NULL,
    [Fullname] [nvarchar](70) NOT NULL,
    [AppSet] [nvarchar](50) NOT NULL,
    [Action] [nvarchar](50) NOT NULL,
    [Status] [int] NULL
) ON [PRIMARY]
GO
```

Step 2: Create the *tblTIMLog* SQL table
You can run the following piece of SQL Script in order to create the table *tblTIMLog.* This table will contain text that can be used either for informational and auditable purposes, or it can be used for debugging purposes.

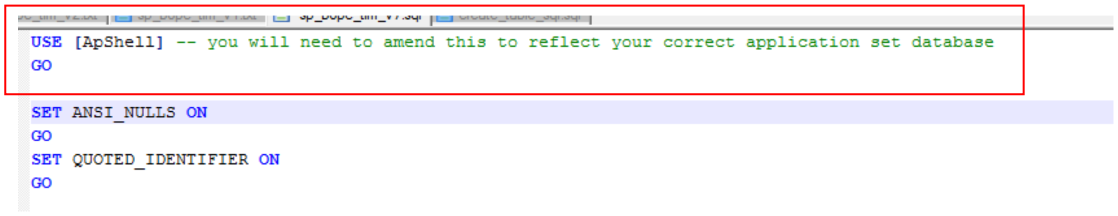
```
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tblTIMLog](
    [timestamp] [datetime] NULL,
    [action] [nvarchar](10) NOT NULL,
    [username] [nvarchar](50) NOT NULL,
    [teamID] [nvarchar](50) NOT NULL,
    [appset] [nvarchar](50) NOT NULL,
    [strSQL] [varchar](8000) NOT NULL,
    [strMessage] [nvarchar](1000) NOT NULL
) ON [PRIMARY]

GO

SET ANSI_PADDING OFF
GO

ALTER TABLE [dbo].[tblTIMLog] ADD  CONSTRAINT [DF_tblTIMLog_timestamp]  DEFAULT
(getdate()) FOR [timestamp]
GO
```

Step 3: Create the *sp_bopc_tim* SQL stored procedure
You can open the file *sp_bopc_tim_v7.sql* which contains the SQL Script in order to create the SQL procedure *sp_bopc_tim.* This stored procedure will contain all of the logic which will perform the provisioning and decommissioning of the users into SAP BPC.

Open the file in SQL Server Management Studio, edit the below statement to reflect your environment

```
    USE [ApShell] -- you will need to amend this to reflect your correct application set
    database
    GO
```

Figure 2: Statement which needs to be amended before running the SQL script file



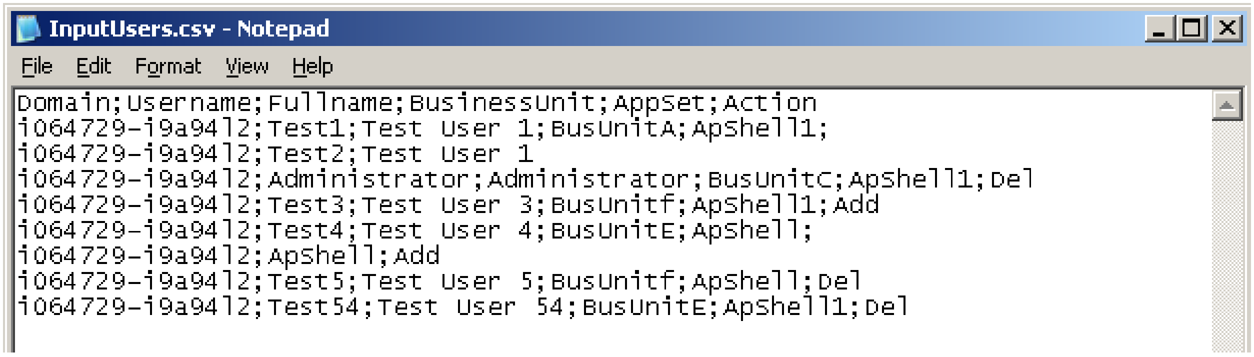
```
USE [ApShell] -- you will need to amend this to reflect your correct application set database
GO

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

IBM Tivoli Directory Integrator Configuration

This document is not meant to serve as a reference document for IBM TDI, the Assembly line presented in this document is meant to guide and be a basis for building your assembly line in order to perform the necessary integration. The Assembly Line ***AL_SAP_BOPC_Users*** is a basic representation of the possible configurations which might occur in order to integrate the two solutions.

The source of the users which will be provisioned or decommissioned into SAP BPC is InputUsers.csv. For testing purposes this was the approach taken, but in a real world scenario it would be a LDAP directory or an Active Directory domain. Depending on the source of the users, the destination for the user objects to be provisioned will have to be the table *tblTIMUsers*; this is the only source for the SQL stored procedure.

Figure 3: The InputUsers.csv which contains the users to be provisioned into SAP BPC



```
Domain;Username;Fullname;BusinessUnit;AppSet;Action
i064729-i9a9412;Test1;Test User 1;BusUnitA;ApShell1;
i064729-i9a9412;Test2;Test User 1
i064729-i9a9412;Administrator;Administrator;BusUnitC;ApShell1;Del
i064729-i9a9412;Test3;Test User 3;BusUnitf;ApShell1;Add
i064729-i9a9412;Test4;Test User 4;BusUnitE;ApShell;
i064729-i9a9412;ApShell;Add
i064729-i9a9412;Test5;Test User 5;BusUnitf;ApShell;Del
i064729-i9a9412;Test54;Test User 54;BusUnitE;ApShell1;Del
```

Step 1: Import the TDI configuration project file

Please extract the downloaded IBM TDI project files to a location which is accessible from the TDI configuration Editor. Create a new project called *SAP_BOPC_Integration*. Once you have created a new project, you can import the configurations from the file ***SAP_BOPC_Integration.xml***

Figure 4: The IBM TDI Import Wizard screen

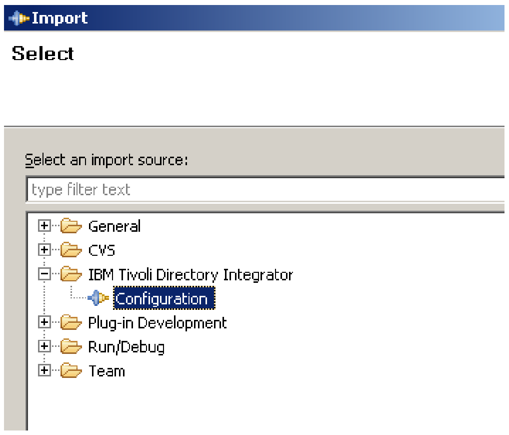
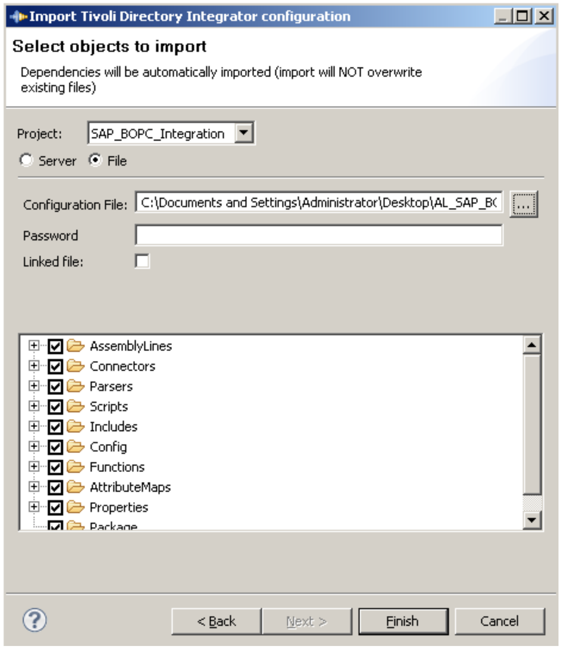


Figure 5: IBM TDI Import Wizard Configuration screen



Import Tivoli Directory Integrator configuration

Select objects to import

Dependencies will be automatically imported (import will NOT overwrite existing files)

Project: SAP_BOPC_Integration

Server [✓ File]

Configuration File: C:\Documents and Settings\Administrator\Desktop\AL_SAP_B(

Password

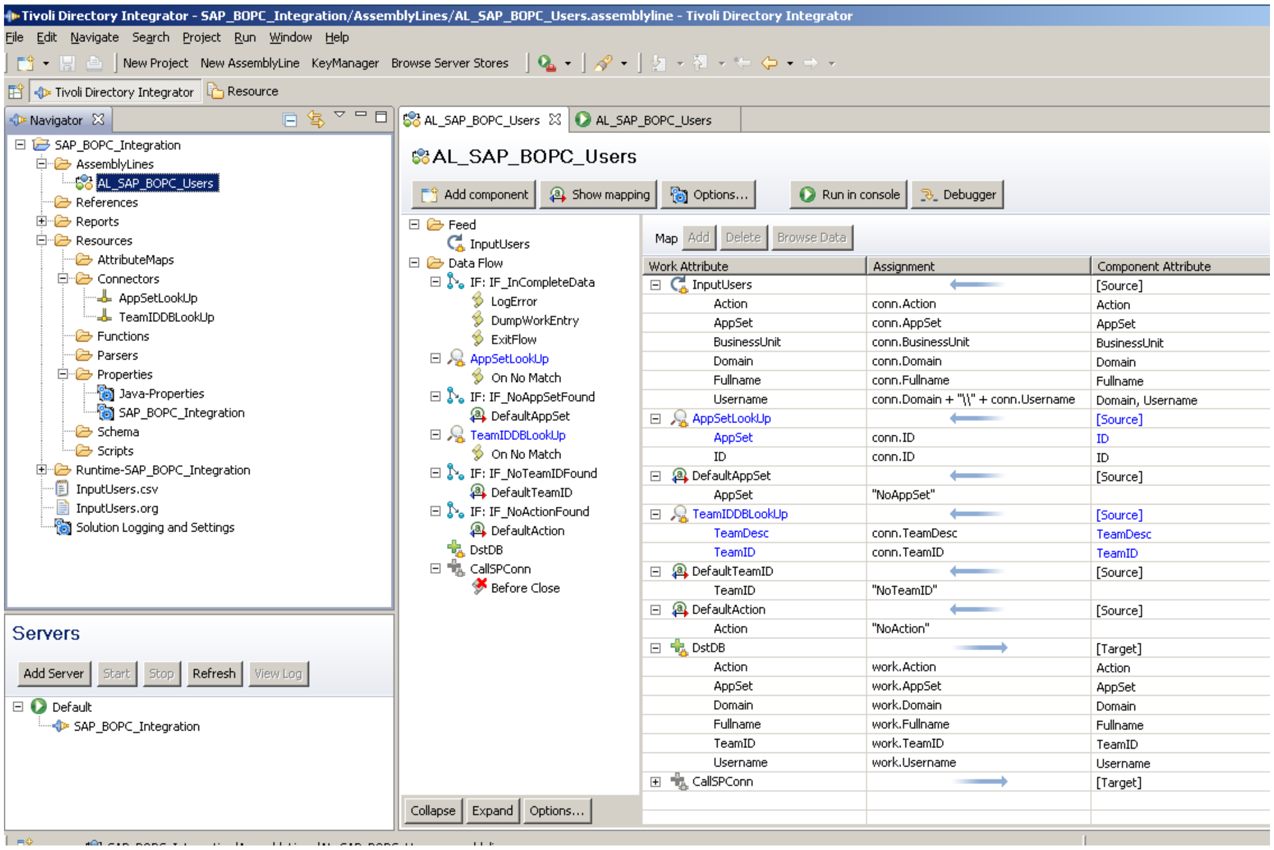
Linked file:

- [✓] AssemblyLines
- [✓] Connectors
- [✓] Parsers
- [✓] Scripts
- [✓] Includes
- [✓] Config
- [✓] Functions
- [✓] AttributeMaps
- [✓] Properties
- [✓] Package

Step 2: Modify the InputUsers connector object path to the correct locations in your environment

Once you have succesfully imported the configuration file into IBM TDI Configuration Editor, the assembly line ***AL_SAP_BOPC_Users*** should be imported. When double clikcing on the assembly line, the below configuration editor screen appears.

Figure 6: Assembly line *AL_SAP_BOPC_Users* configuration editor screen



Tivoli Directory Integrator - SAP_BOPC_Integration/AssemblyLines/AL_SAP_BOPC_Users.assemblyline - Tivoli Directory Integrator

```
SAP_BOPC_Integration
  AssemblyLines
    AL_SAP_BOPC_Users
  References
  Reports
  Resources
    AttributeMaps
    Connectors
      AppSetLookUp
      TeamIDDBLookUp
    Functions
    Parsers
    Properties
      Java-Properties
      SAP_BOPC_Integration
    Schema
    Scripts
  Runtime-SAP_BOPC_Integration
  InputUsers.csv
  InputUsers.org
  Solution Logging and Settings
```

```
Default
  SAP_BOPC_Integration
```

AL_SAP_BOPC_Users

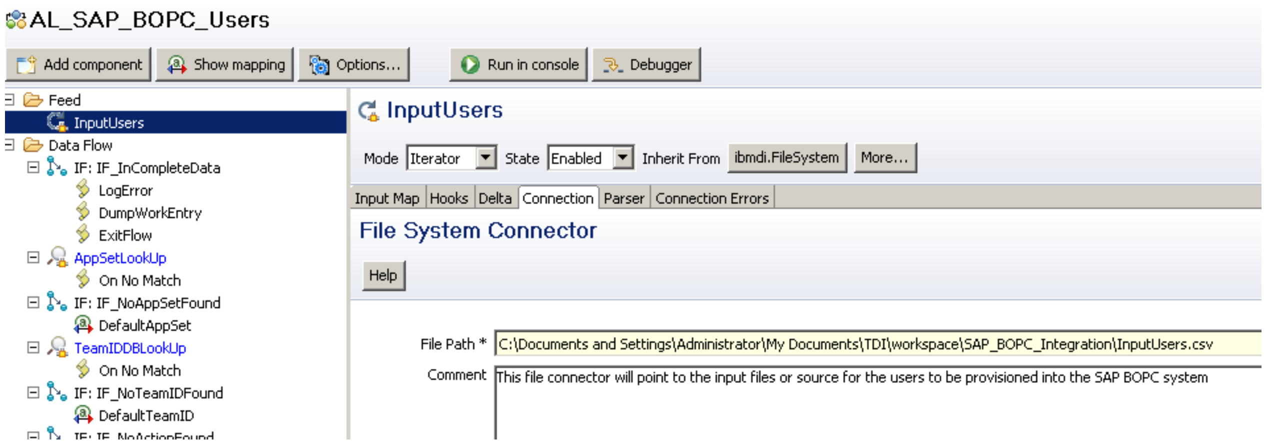
```
Feed
  InputUsers
Data Flow
  IF: IF_InCompleteData
    LogError
    DumpWorkEntry
    ExitFlow
  AppSetLookUp
    On No Match
  IF: IF_NoAppSetFound
    DefaultAppSet
  TeamIDDBLookUp
    On No Match
  IF: IF_NoTeamIDFound
    DefaultTeamID
  IF: IF_NoActionFound
    DefaultAction
  DstDB
  CallSPConn
    Before Close
```

Work Attribute	Assignment	Component Attribute
InputUsers	←	[Source]
Action	conn.Action	Action
AppSet	conn.AppSet	AppSet
BusinessUnit	conn.BusinessUnit	BusinessUnit
Domain	conn.Domain	Domain
Fullname	conn.Fullname	Fullname
Username	conn.Domain + "\\" + conn.Username	Domain, Username
AppSetLookUp	←	[Source]
AppSet	conn.ID	ID
ID	conn.ID	ID
DefaultAppSet	←	[Source]
AppSet	"NoAppSet"	
TeamIDDBLookUp	←	[Source]
TeamDesc	conn.TeamDesc	TeamDesc
TeamID	conn.TeamID	TeamID
DefaultTeamID	←	[Source]
TeamID	"NoTeamID"	
DefaultAction	←	[Source]
Action	"NoAction"	
DstDB	→	[Target]
Action	work.Action	Action
AppSet	work.AppSet	AppSet
Domain	work.Domain	Domain
Fullname	work.Fullname	Fullname
TeamID	work.TeamID	TeamID
Username	work.Username	Username
CallSPConn	→	[Target]

You will need to amend the *InputUsers* connector component path to the file InputUsers.csv. You can achive this by double clicking on the InputUsers connector component and navigating to the connection tab.

The value that needs to change to reflect the correct path in your environment is *FilePath*.

Figure 7: *InputUsers* file connector component connection configuration editor screen



Mode Iterator State Enabled Inherit From ibmdi.FileSystem

File Path * C:\Documents and Settings\Administrator\My Documents\TDI\workspace\SAP_BOPC_Integration\InputUsers.csv

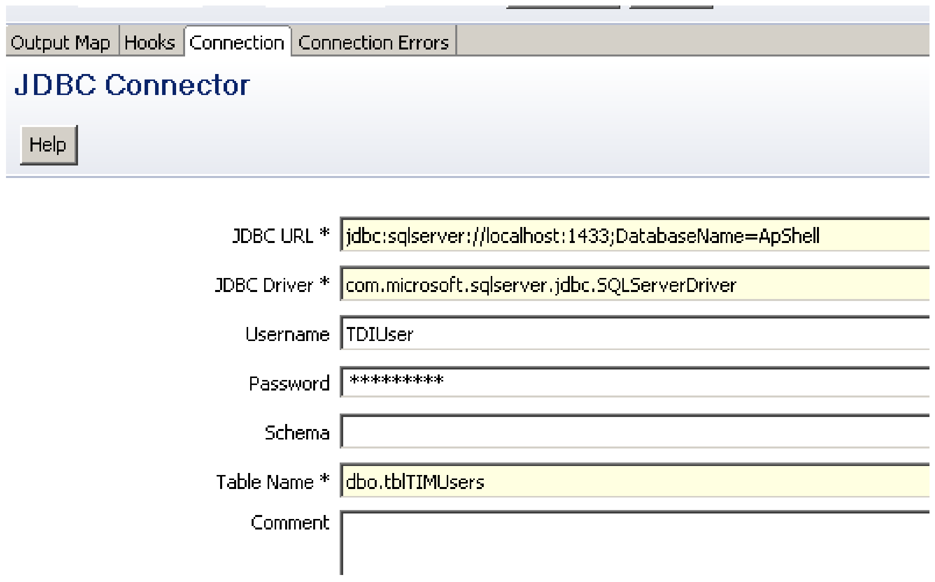
Comment This file connector will point to the input files or source for the users to be provisioned into the SAP BOPC system

Step 3: Modify the *DstDB* connector object to point to the SQL Server in your environment

Double click on the *DstDB* connector object and click on the connection tab. Change the values according to reflect the correct values for your environment.

Note: The implementation class for this ODBC driver is SQL 2008, you may need to adjust this if you are using another version of SQL server. Please refer to the IBM TDI and Microsoft SQL JDBC driver documentation for additional reference and troubleshooting.

Figure 8: *DstDB* SQL connector component connection configuration editor screen



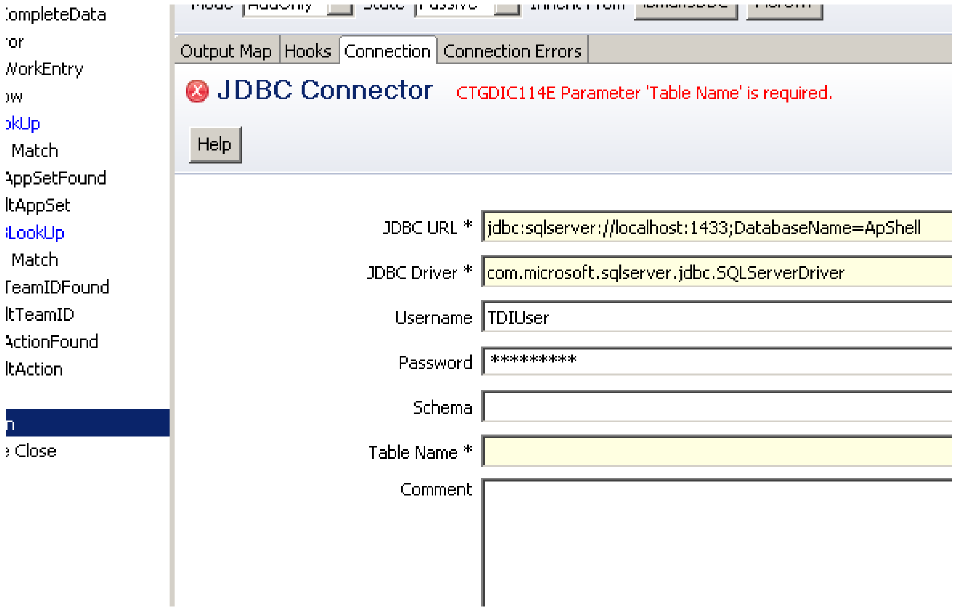
JDBC URL *	jdbc:sqlserver://localhost:1433;DatabaseName=ApShell
JDBC Driver *	com.microsoft.sqlserver.jdbc.SQLServerDriver
Username	TDIUser
Password	*********
Schema	
Table Name *	dbo.tblTIMUsers
Comment	

Step 4: Modify the *CallSPConn* connector object to point to the SQL Server in your environment

Double click on the *CallSPConn* connector object and click on the connection tab. Change the values according to reflect the correct values for your environment.

This connector will be responsible for the calling of the SQL stored procedure. In the configuration editor when inputing the values for your environment, because this connector object will be callings a Stored Procedure you do not have to fill in a table name. The below screen will appear.

Figure 9: *CallSPConn* SQL connector component connection configuration editor screen



JDBC Connector CTGDIC114E Parameter 'Table Name' is required.

JDBC URL * jdbc:sqlserver://localhost:1433;DatabaseName=ApShell

JDBC Driver * com.microsoft.sqlserver.jdbc.SQLServerDriver

Username TDIUser

Password *********

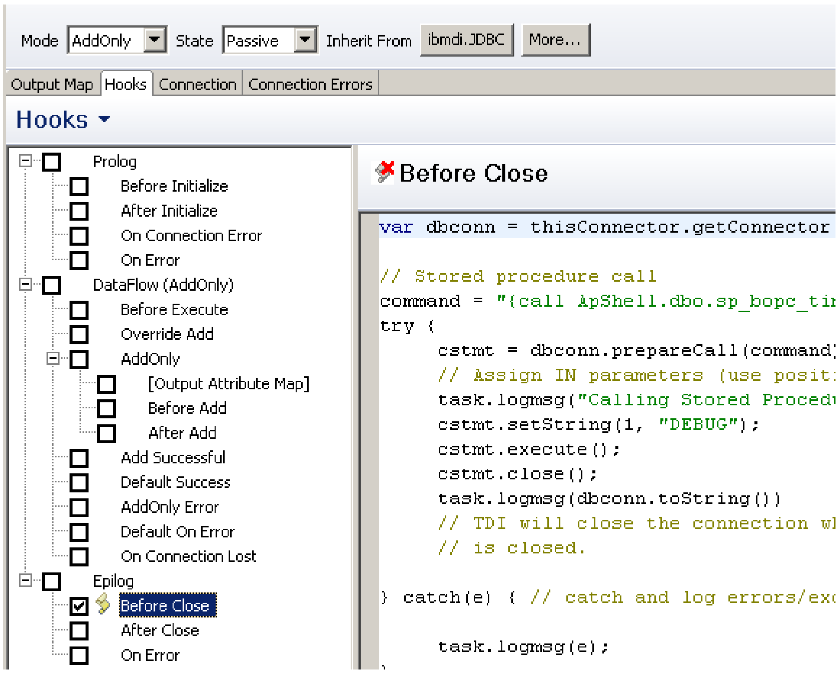
Schema

Table Name *

Comment

In the *Hooks* tab add the ‘*Before Close’* hook to the *CallSPConn* connector object.

Figure 10: *CallSPConn* SQL connector component *Hooks* configuration editor screen



```
var dbconn = thisConnector.getConnector

// Stored procedure call
command = "{call ApShell.dbo.sp_bopc_ti
try {
     cstmt = dbconn.prepareCall(command)
     // Assign IN parameters (use posit
     task.logmsg("Calling Stored Proced
     cstmt.setString(1, "DEBUG");
     cstmt.execute();
     cstmt.close();
     task.logmsg(dbconn.toString())
     // TDI will close the connection w
     // is closed.

} catch(e) { // catch and log errors/ex

     task.logmsg(e);
```

In the hooks configuratio editor screen, you will need to modify the piece of code :

```
command = "{call ApShell.dbo.sp_bopc_tim (?) }";
```

to the reflect your application set database name

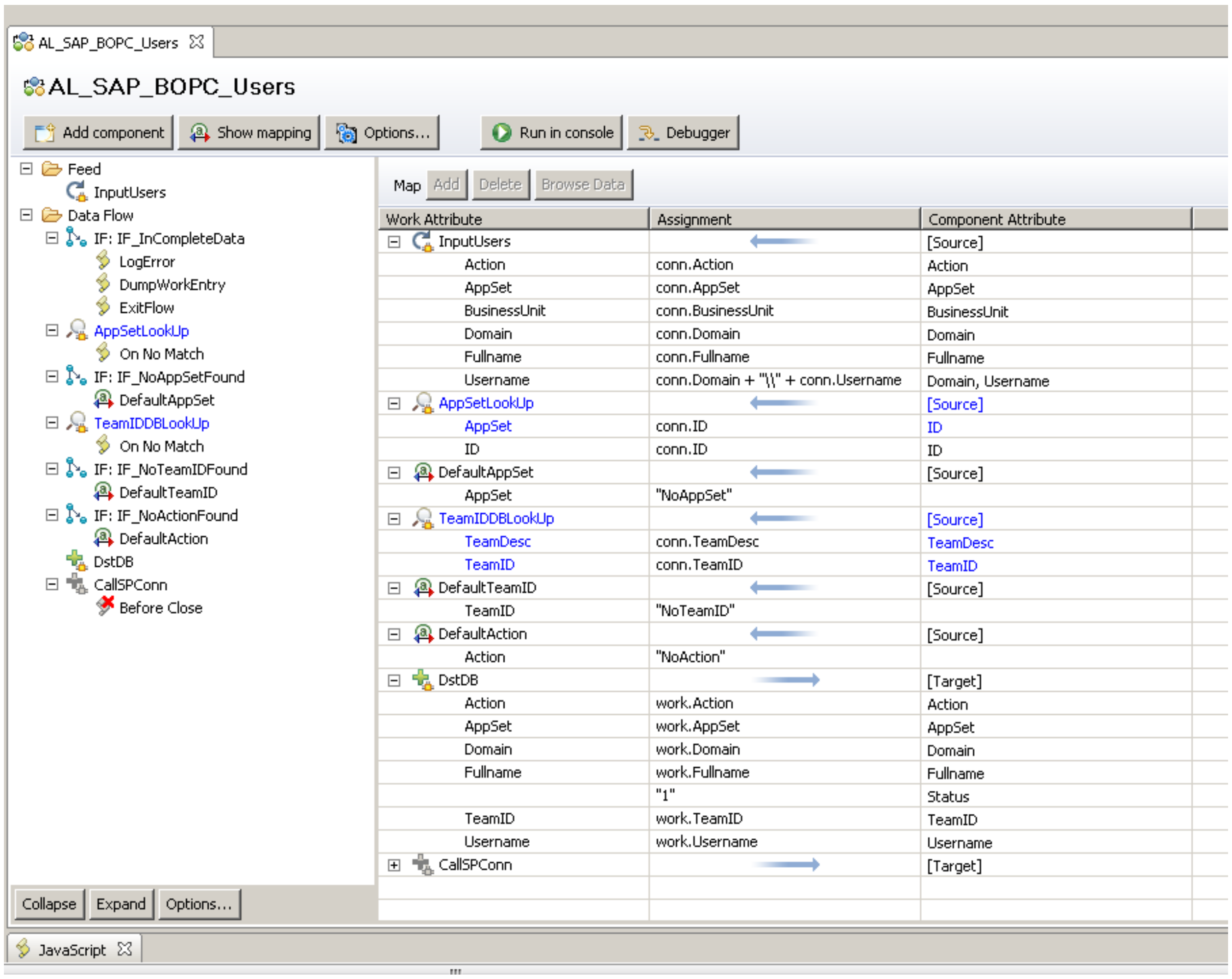
```
command = "{call <YourDBName>.dbo.sp_bopc_tim (?) }";
```

This piece of code is what actually calls the SQL stored procedure.

Once you have completed these intial configuration steps, you should be able to run the assembly line to input the users from a flat file called *InputUsers.csv* to the <AppSet>.dbo.tblTIMUsers table.

The intial assembly line configuration should appear as per below **Figure 11**

Figure 11: Assembly Line *AL_SAP_BOPC_Users* configuration and work flow screen



```
⊟ Feed
      InputUsers
⊟ Data Flow
   ⊟ IF: IF_InCompleteData
         LogError
         DumpWorkEntry
         ExitFlow
   ⊟ AppSetLookUp
         On No Match
   ⊟ IF: IF_NoAppSetFound
         DefaultAppSet
   ⊟ TeamIDDBLookUp
         On No Match
   ⊟ IF: IF_NoTeamIDFound
         DefaultTeamID
   ⊟ IF: IF_NoActionFound
         DefaultAction
     DstDB
   ⊟ CallSPConn
         Before Close
```

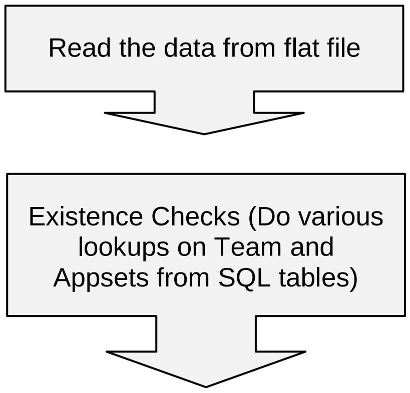
Work Attribute	Assignment	Component Attribute
⊟ InputUsers	←	[Source]
Action	conn.Action	Action
AppSet	conn.AppSet	AppSet
BusinessUnit	conn.BusinessUnit	BusinessUnit
Domain	conn.Domain	Domain
Fullname	conn.Fullname	Fullname
Username	conn.Domain + "\\" + conn.Username	Domain, Username
⊟ AppSetLookUp	←	[Source]
AppSet	conn.ID	ID
ID	conn.ID	ID
⊟ DefaultAppSet	←	[Source]
AppSet	"NoAppSet"	
⊟ TeamIDDBLookUp	←	[Source]
TeamDesc	conn.TeamDesc	TeamDesc
TeamID	conn.TeamID	TeamID
⊟ DefaultTeamID	←	[Source]
TeamID	"NoTeamID"	
⊟ DefaultAction	←	[Source]
Action	"NoAction"	
⊟ DstDB	→	[Target]
Action	work.Action	Action
AppSet	work.AppSet	AppSet
Domain	work.Domain	Domain
Fullname	work.Fullname	Fullname
	"1"	Status
TeamID	work.TeamID	TeamID
Username	work.Username	Username
⊞ CallSPConn	→	[Target]

Integration Process Flows

This section will detail how the integration process flows through the different components.

Source Phase – IBM TDI

The originator of the integration will be the IBM TDI Assembly line: *AL_SAP_BOPC_Users.*


Read the data from flat file
Existence Checks (Do various lookups on Team and Appsets from SQL tables)

The assembly line will do lookups to check for the existence of the Team and Appset; it assigns a default value if the lookup value fails. For the lookup of the Appset, It does a look up to the following table *AppServer.dbo.tblAppSetInfo* to check for the existence of the application set. If the lookup fails then a default value of *NoAppSet* is defined and inserted into the column of the table.

For the lookup of the teams, It does a look up to the following table *<AppSet>.dbo.teams* to check for the existence of the team. If the lookup fails then a default value of *NoTeamID* is defined and inserted into the column of the table.

If no action is defined in the source file, then the assembly line will define a default value of *NoAction* and inserted into the column of the table.

All records which are written into the table tblTIMUsers are written with a status record of **1**, which indicates a new record from IBM TDI

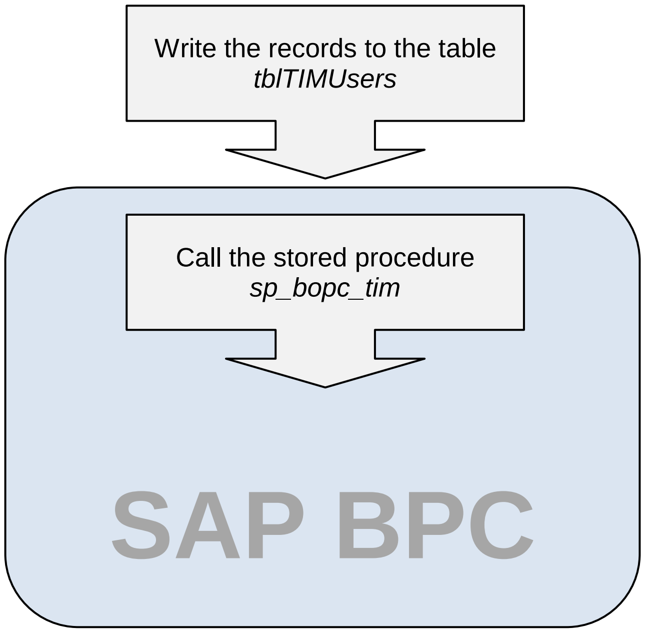
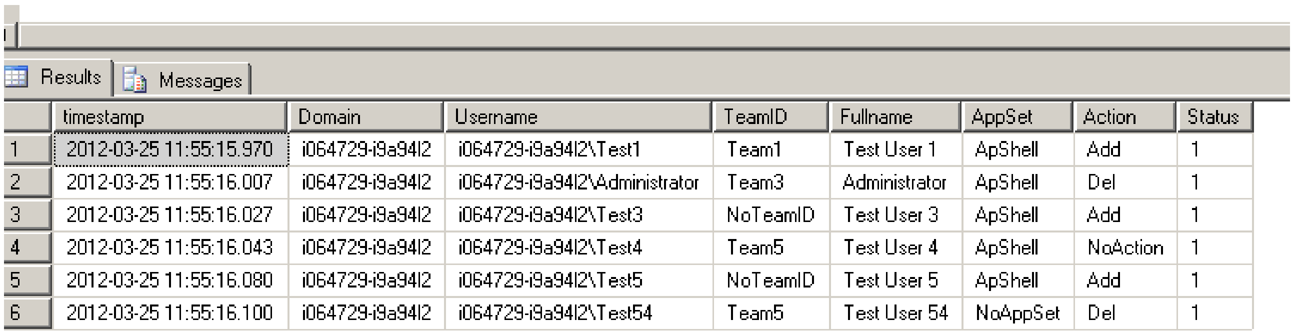
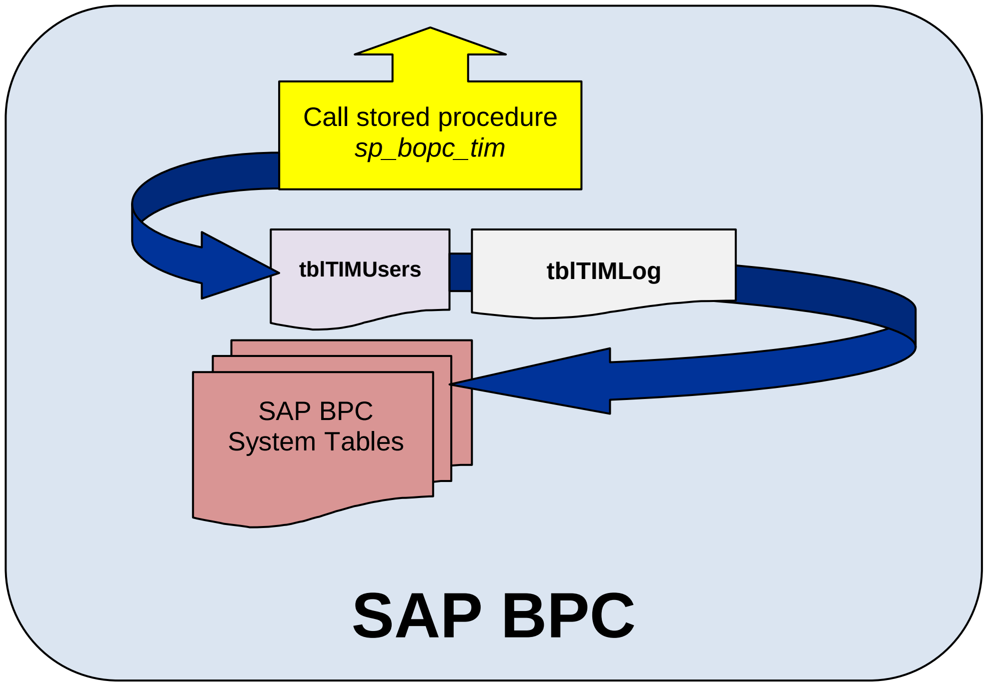

Write the records to the table *tblTIMUsers*
Call the stored procedure *sp_bopc_tim*
SAP BPC

Figure 12: Records in the table *tblTIMUsers* with invalid records


Results | Messages

	timestamp	Domain	Username	TeamID	Fullname	AppSet	Action	Status
1	2012-03-25 11:55:15.970	i064729-i9a94l2	i064729-i9a94l2\Test1	Team1	Test User 1	ApShell	Add	1
2	2012-03-25 11:55:16.007	i064729-i9a94l2	i064729-i9a94l2\Administrator	Team3	Administrator	ApShell	Del	1
3	2012-03-25 11:55:16.027	i064729-i9a94l2	i064729-i9a94l2\Test3	NoTeamID	Test User 3	ApShell	Add	1
4	2012-03-25 11:55:16.043	i064729-i9a94l2	i064729-i9a94l2\Test4	Team5	Test User 4	ApShell	NoAction	1
5	2012-03-25 11:55:16.080	i064729-i9a94l2	i064729-i9a94l2\Test5	NoTeamID	Test User 5	ApShell	Add	1
6	2012-03-25 11:55:16.100	i064729-i9a94l2	i064729-i9a94l2\Test54	Team5	Test User 54	NoAppSet	Del	1

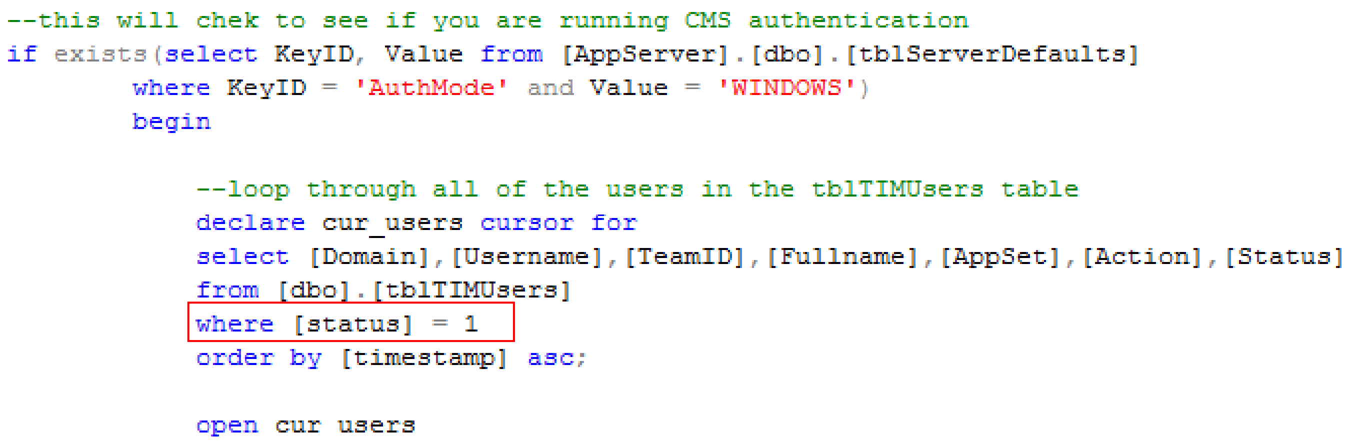
Destination Phase – SAP BPC



Once IBM TDI has succesfully completed inserting the records into the table *tblTIMUsers*, the next step to complete the integration will be to call the stored procedure. The stored procedure *sp_bopc_tim* contains all of the logic which is needed in order to provision and decommision the users to SAP BPC.

The SQL Stored procedue will only retireve records with STATUS=1 from the table *tblTIMUsers*, this means that it will only retrieve valid new records from the table. It builds a restultset and propluates it into a SQL cursor,once the cursor has a recordset to work with, it steps through each record one at a time.

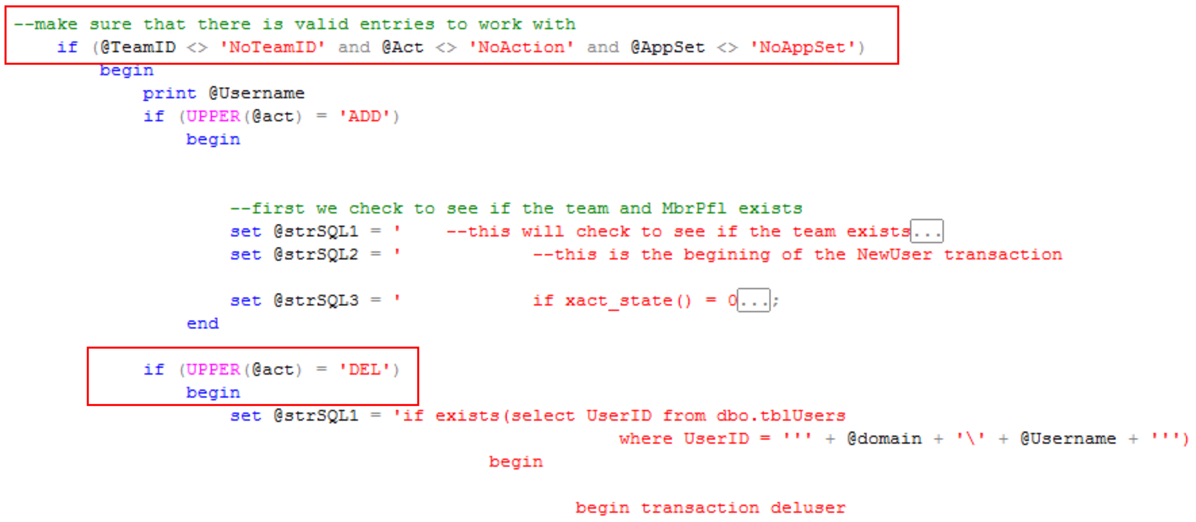
Figure 13: SQL Code which retrieves records from table *tblTIMUsers* with *STATUS=1*



```
--this will chek to see if you are running CMS authentication
if exists(select KeyID, Value from [AppServer].[dbo].[tblServerDefaults]
        where KeyID = 'AuthMode' and Value = 'WINDOWS')
        begin

            --loop through all of the users in the tblTIMUsers table
            declare cur_users cursor for
            select [Domain],[Username],[TeamID],[Fullname],[AppSet],[Action],[Status]
            from [dbo].[tblTIMUsers]
            where [status] = 1
            order by [timestamp] asc;

            open cur users
```

Once it has a record to work with, it will do various checks, for example: Check if the user already exists and if the action defined is valid action. In the below figure the actions which are only applicable are ADD – Which will provision the user and DEL- Which will remove the user from SAP BPC.

Figure 14: SQL Code to check to if it there is no invalid data from table *tblTIMUsers*

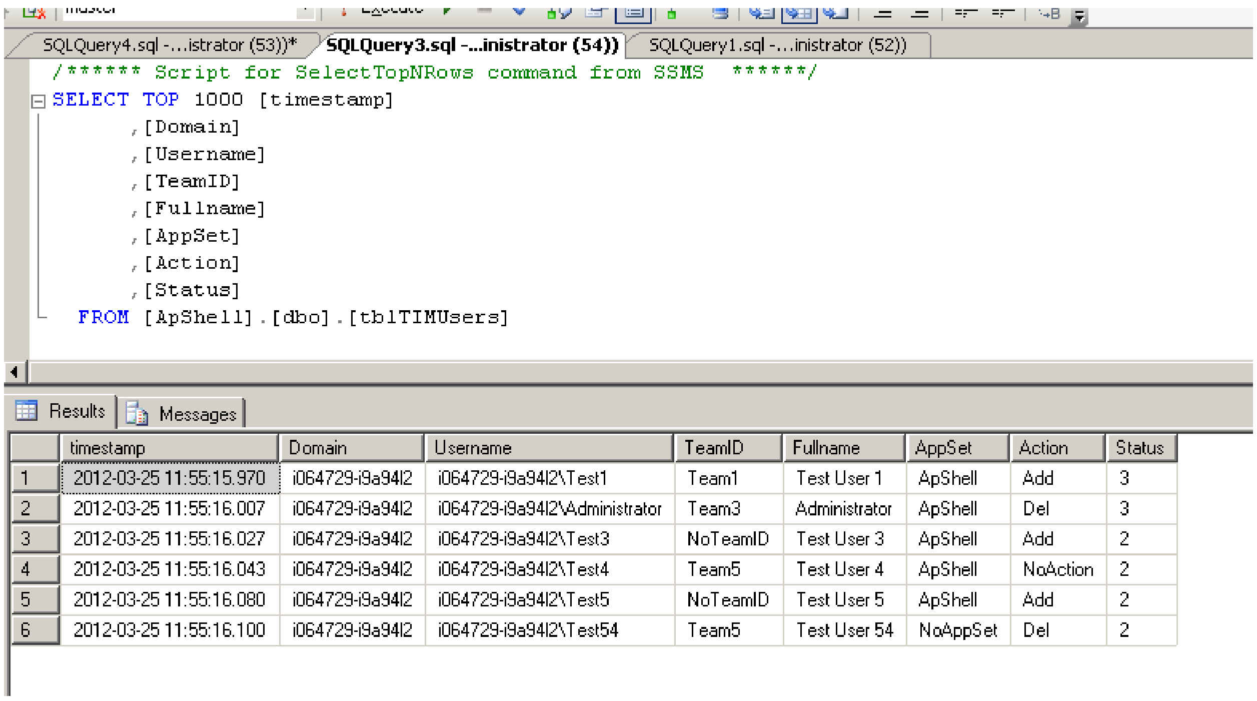


```
--make sure that there is valid entries to work with
    if (@TeamID <> 'NoTeamID' and @Act <> 'NoAction' and @AppSet <> 'NoAppSet')
        begin
            print @Username
            if (UPPER(@act) = 'ADD')
                begin


                    --first we check to see if the team and MbrPfl exists
                    set @strSQL1 = '    --this will check to see if the team exists...
                    set @strSQL2 = '            --this is the begining of the NewUser transaction

                    set @strSQL3 = '            if xact_state() = 0...;
                end

            if (UPPER(@act) = 'DEL')
                begin
                    set @strSQL1 = 'if exists(select UserID from dbo.tblUsers
                                                        where UserID = ''' + @domain + '\' + @Username + ''')
                                                begin

                                                        begin transaction deluser
```

Once the stored procedure has run succesfully it will update the table *tblTIMUsers* status column accorindlgy. The following status records will apply.

Status Value	Status Description
1	New Record inserted into the table (New User)
2	Invalid Record, or there was an error with one of the column values
3	Succesfully processed the record into SAP BPC system tables

Figure 15: Table tblTIMUsers when the stored procedure has completed


SQLQuery4.sql -...istrator (53))*
SQLQuery3.sql -...inistrator (54))
SQLQuery1.sql -...inistrator (52))
Results
Messages

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [timestamp]
      ,[Domain]
      ,[Username]
      ,[TeamID]
      ,[Fullname]
      ,[AppSet]
      ,[Action]
      ,[Status]
  FROM [ApShell].[dbo].[tblTIMUsers]
```

	timestamp	Domain	Username	TeamID	Fullname	AppSet	Action	Status
1	2012-03-25 11:55:15.970	i064729-i9a94l2	i064729-i9a94l2\Test1	Team1	Test User 1	ApShell	Add	3
2	2012-03-25 11:55:16.007	i064729-i9a94l2	i064729-i9a94l2\Administrator	Team3	Administrator	ApShell	Del	3
3	2012-03-25 11:55:16.027	i064729-i9a94l2	i064729-i9a94l2\Test3	NoTeamID	Test User 3	ApShell	Add	2
4	2012-03-25 11:55:16.043	i064729-i9a94l2	i064729-i9a94l2\Test4	Team5	Test User 4	ApShell	NoAction	2
5	2012-03-25 11:55:16.080	i064729-i9a94l2	i064729-i9a94l2\Test5	NoTeamID	Test User 5	ApShell	Add	2
6	2012-03-25 11:55:16.100	i064729-i9a94l2	i064729-i9a94l2\Test54	Team5	Test User 54	NoAppSet	Del	2

Recommended Approaches

This section will outline the recommended approaches on how to implement the integration between the two solutions. In order to ensure that there are no issues with the implementation, it is imperative to remember certain key principles:

If the user already exists then it will not run on that user record. Hence if you need the integration code to move users to different teams or member access profiles then this scenario will not work. It is recommended to use the SAP BPC Admin console to amend the security profile of the user. This integration code will only be used for the provisioning and decommissioning of the user object.

It is recommended that you ***create the necessary Teams in SAP BPC and then assign the necessary Member Access Profiles to the team.*** The rationale is that you simply add the user to the team and then manage the security for the user object using the SAP BPC Admin console. Although it is possible to modify and extend the integration code to amend the user objects security, for example: assigning the user to a different team. It will not be recommended as you will be modifying the security of a user object outside of SAP BPC.

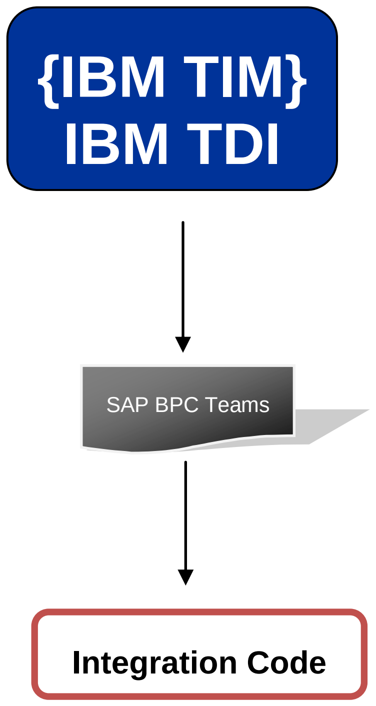
Expose the teams created in SAP BPC. This means that when provisioning the user in IBM TDI, it will assign it to the correct teams and wont cause an issue with the user object being assigned to a team in which doesn't exist. For example: It is recommened to put in a place a business process in which will query SAP BPC for existing teams and then assign the user to an existing team.

It is recommeneded that you clearly define your security within SAP BPC, into manageable componenets. The easiest and simpliest way is to predefine and create your teams, then when you have a request to add a new user using IBM TIM, you simply define which team the user will be a member of. Then the integration code simply adds the user to a team and SAP BPC will take care of the rest of the security, because it will already have a predefined member access profile

The easiest way to query SAP BPC for the existing teams is to have a lookup connection using the following SQL Query

```
select TeamID from <AppSet>.dbo.Teams
```

Recommended Integration Process Flow



It is recommended that when provisiong a new user using the IBM TIM interface, that it does a lookup to SAP BPC to verify that the team exists. This will greatly simplify the process and reduce any complexity in the support phase.

If ***UserA*** is being provisioned for access to SAP BPC, then it checks to see that the team in which ***UserA*** exists. In SAP BPC the team will already have the member access profile defined.

Although this approach is the recommended one it is dependent on having the security already defined within SAP BPC. This process would have to be controlled through your existing change management and support business process in order to ensure that there are no gaps in the process.

For example: What happens if your security model in SAP BPC is modeled around Business Units and a new Business Unit has been created but it hasn't filtered through to being created in SAP BPC. If the user provisioning process has not catered for any exceptions, then it could be a bottleneck in your process. It is recommended that there be exception based processing as well. For example: If the team doesn't exist, send an email to approving manager or the requestor to create the team in SAP BPC.

Ultimately this process will need to be documented and managed in accordance to your organizations change management process.

Limitations

This section will outline the limitations with the integration code:

- ***Only provisioning or decommissioning of user objects***. This means it is possible to extend the integration code to perform other activities. For example: It is possible to use the integration code to modify the user's security access and change their team membership and member access profiles. Although it is possible it is not recommended as you will be creating an extra layer of complexity and another possible point of failure.
- ***Windows Authentication Only***: This integration code will only check for the authentication mode, if it is set to CMS, it will not run. It is possible to change and make it run, you will just need to amend the code not to prefix the domain when it is using CMS mode.
- ***SAP BPC version 7.5.*** Although it was tested on SAP BPC version 10, it was only basic testing and not enough scenarios were tested to ensure that it is compatible with SAP BPC version 10. It is anticipated that in the next release of the integration code it will support SAP BPC version 10.

- ***Single Team provision*** means that it will only be able to provision the user to a single team in SAP BPC using the integration code. Although it is possible to assign a user to multiple teams, in this initial release it was only possible to assign it to a single team.

Troubleshooting

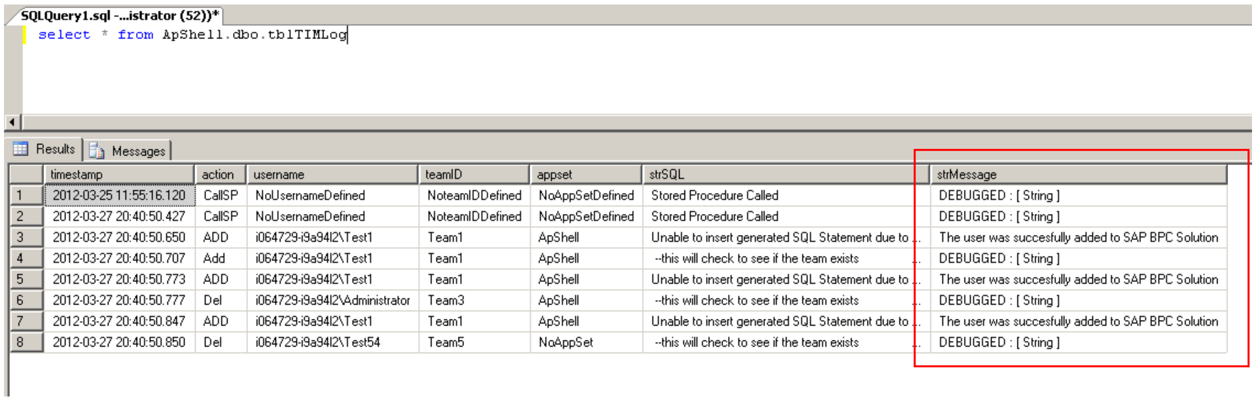
This section will outline some basic steps which could be used to troubleshoot the integration process and code. Please Note: This document will not cover debugging the process in IBM TDI; it is recommended that you refer to the official IBM TDI documentation for further information.

- **DEBUGING** the stored procedure

  In order to debug the stored procedure, you can pass the parameter 'DEBUG'. This will record the strSQL parameter values in the tblTIMLog table. This table will have detailed information regarding the execution of the stored procedure.

  In order to enable the stored procedure to provide verbose logging information, pass the DEBUG parameter when calling the stored procedure in IBM TDI.

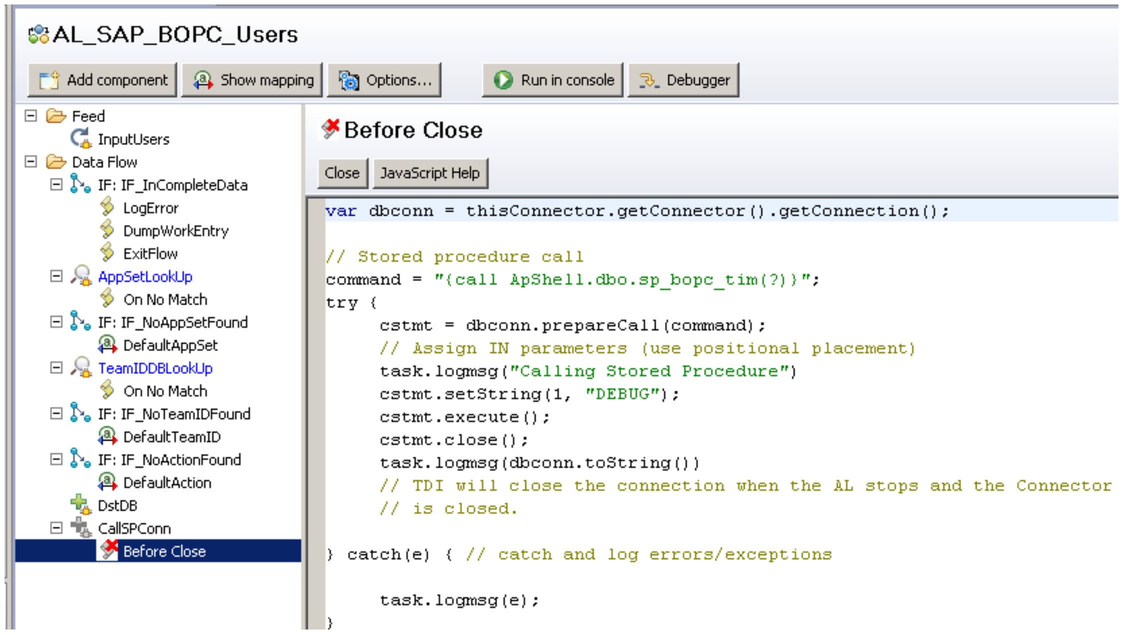
Figure 16: Records in the tblTIMLog with the DEBUG parameter enabled



```
select * from ApShell.dbo.tblTIMLog
```

	timestamp	action	username	teamID	appset	strSQL	strMessage
1	2012-03-25 11:55:16.120	CallSP	NoUsernameDefined	NoteamIDDefined	NoAppSetDefined	Stored Procedure Called	DEBUGGED : [String]
2	2012-03-27 20:40:50.427	CallSP	NoUsernameDefined	NoteamIDDefined	NoAppSetDefined	Stored Procedure Called	DEBUGGED : [String]
3	2012-03-27 20:40:50.650	ADD	i064729-i9a94l2\Test1	Team1	ApShell	Unable to insert generated SQL Statement due to ...	The user was succesfully added to SAP BPC Solution
4	2012-03-27 20:40:50.707	Add	i064729-i9a94l2\Test1	Team1	ApShell	--this will check to see if the team exists ...	DEBUGGED : [String]
5	2012-03-27 20:40:50.773	ADD	i064729-i9a94l2\Test1	Team1	ApShell	Unable to insert generated SQL Statement due to ...	The user was succesfully added to SAP BPC Solution
6	2012-03-27 20:40:50.777	Del	i064729-i9a94l2\Administrator	Team3	ApShell	--this will check to see if the team exists ...	DEBUGGED : [String]
7	2012-03-27 20:40:50.847	ADD	i064729-i9a94l2\Test1	Team1	ApShell	Unable to insert generated SQL Statement due to ...	The user was succesfully added to SAP BPC Solution
8	2012-03-27 20:40:50.850	Del	i064729-i9a94l2\Test54	Team5	NoAppSet	--this will check to see if the team exists ...	DEBUGGED : [String]

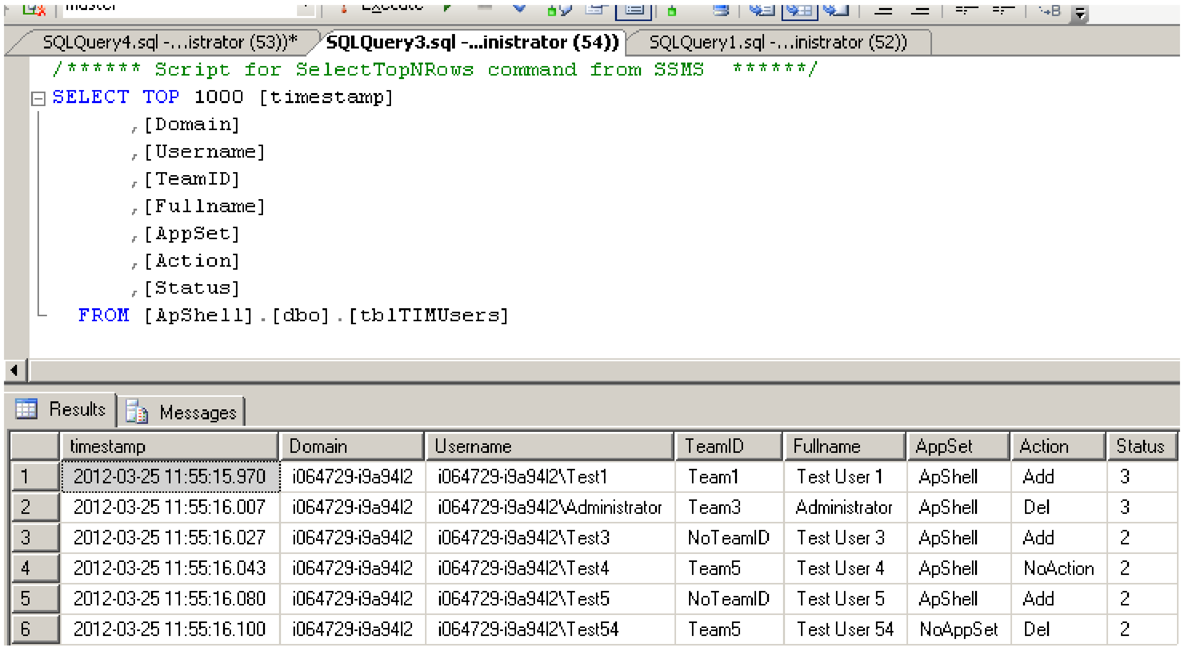
Figure 17: Example code: Passing the DEBUG parameter when calling the stored procedure



```
var dbconn = thisConnector.getConnector().getConnection();

// Stored procedure call
command = "{call ApShell.dbo.sp_bopc_tim(?)}";
try {
      cstmt = dbconn.prepareCall(command);
      // Assign IN parameters (use positional placement)
      task.logmsg("Calling Stored Procedure")
      cstmt.setString(1, "DEBUG");
      cstmt.execute();
      cstmt.close();
      task.logmsg(dbconn.toString())
      // TDI will close the connection when the AL stops and the Connector
      // is closed.

} catch(e) { // catch and log errors/exceptions

      task.logmsg(e);
}
```

- **Verify the records in the tblTIMUsers table;** the easiest way to determine if the stored procedure has run successfully is to verify that the records in the tblTIMUsers table have a status value of 3 against the user object. This means that the user object has either been added or removed successfully.

Figure 18: tblTIMUsers table with invalid and valid records *(Please take note of the status column)*



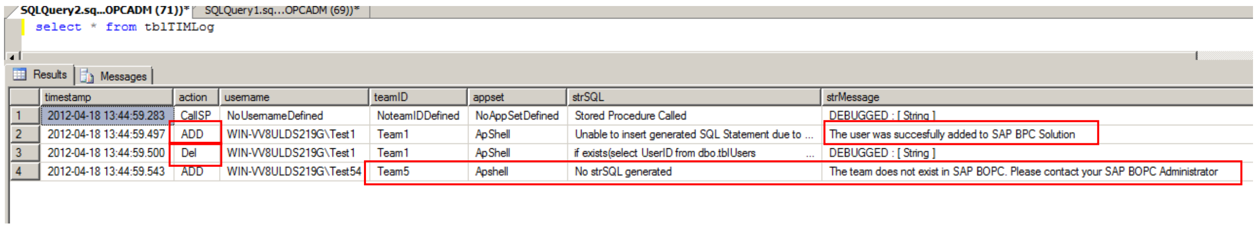
SQLQuery4.sql -...istrator (53))* | SQLQuery3.sql -...inistrator (54)) | SQLQuery1.sql -...inistrator (52))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [timestamp]
      ,[Domain]
      ,[Username]
      ,[TeamID]
      ,[Fullname]
      ,[AppSet]
      ,[Action]
      ,[Status]
  FROM [ApShell].[dbo].[tblTIMUsers]
```

	timestamp	Domain	Username	TeamID	Fullname	AppSet	Action	Status
1	2012-03-25 11:55:15.970	i064729-i9a94l2	i064729-i9a94l2\Test1	Team1	Test User 1	ApShell	Add	3
2	2012-03-25 11:55:16.007	i064729-i9a94l2	i064729-i9a94l2\Administrator	Team3	Administrator	ApShell	Del	3
3	2012-03-25 11:55:16.027	i064729-i9a94l2	i064729-i9a94l2\Test3	NoTeamID	Test User 3	ApShell	Add	2
4	2012-03-25 11:55:16.043	i064729-i9a94l2	i064729-i9a94l2\Test4	Team5	Test User 4	ApShell	NoAction	2
5	2012-03-25 11:55:16.080	i064729-i9a94l2	i064729-i9a94l2\Test5	NoTeamID	Test User 5	ApShell	Add	2
6	2012-03-25 11:55:16.100	i064729-i9a94l2	i064729-i9a94l2\Test54	Team5	Test User 54	NoAppSet	Del	2

- **What happens if status = 3 in the table tblTIMUsers, but the user is not able to log into SAP BPC.** This would mean that there was an issue with inserting or removing the stored procedure. The first step of troubleshooting this scenario would be to check the *tblTIMLog* table for additional information. The stored procedure code has been created to provide the most information in an effort to troubleshoot the process.

Figure 19: tblTIMLog table with records after the execution of the stored procedure



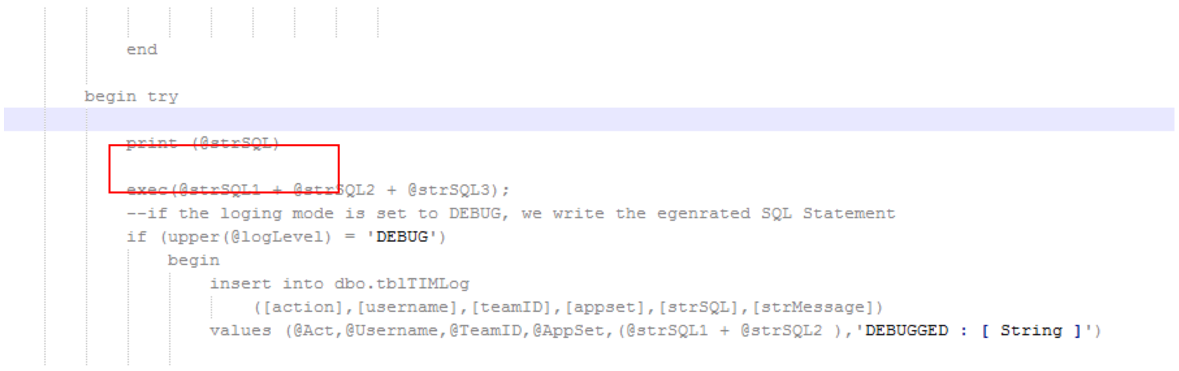
SQLQuery2.sq...OPCADM (71))* | SQLQuery1.sq...OPCADM (69))*

```
select * from tblTIMLog
```

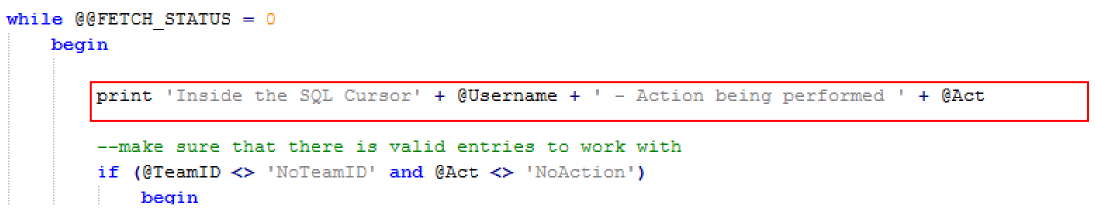
	timestamp	action	username	teamID	appset	strSQL	strMessage
1	2012-04-18 13:44:59.283	CallSP	NoUsernameDefined	NoteamIDDefined	NoAppSetDefined	Stored Procedure Called	DEBUGGED : [String]
2	2012-04-18 13:44:59.497	ADD	WIN-VV8ULDS219G\Test1	Team1	ApShell	Unable to insert generated SQL Statement due to ...	The user was succesfully added to SAP BPC Solution
3	2012-04-18 13:44:59.500	Del	WIN-VV8ULDS219G\Test1	Team1	ApShell	if exists(select UserID from dbo.tblUsers ...	DEBUGGED : [String]
4	2012-04-18 13:44:59.543	ADD	WIN-VV8ULDS219G\Test54	Team5	Apshell	No strSQL generated	The team does not exist in SAP BOPC. Please contact your SAP BOPC Administrator

- **Modify the stored procedure:** You can add the *print* commands to add additional troubleshooting information to the stored procedure. Please note: This troubleshooting step will entail that you run the stored procedure natively and not through the IBM TDI framework. You will need to call the stored procedure in SQL Server management studio to view the contents of the print commands.

Figure 19: Example of adding '*Print*' command to add additional troubleshooting to the stored procedure



```
            end

        begin try

            print (@strSQL)

            exec(@strSQL1 + @strSQL2 + @strSQL3);
            --if the loging mode is set to DEBUG, we write the egenrated SQL Statement
            if (upper(@logLevel) = 'DEBUG')
                begin
                    insert into dbo.tblTIMLog
                        ([action],[username],[teamID],[appset],[strSQL],[strMessage])
                    values (@Act,@Username,@TeamID,@AppSet,(@strSQL1 + @strSQL2 ),'DEBUGGED : [ String ]')
```

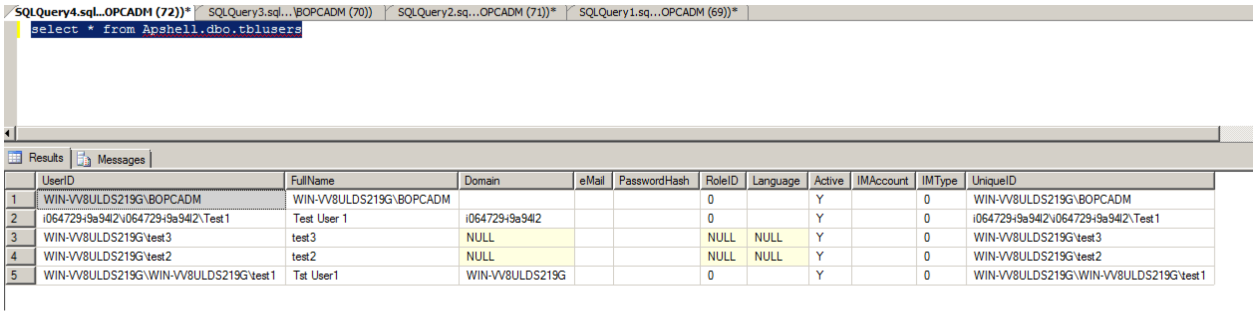


```
while @@FETCH_STATUS = 0
    begin

        print 'Inside the SQL Cursor' + @Username + ' - Action being performed ' + @Act

        --make sure that there is valid entries to work with
        if (@TeamID <> 'NoTeamID' and @Act <> 'NoAction')
            begin
```

- **Query the SAP BPC system tables to verify if the records exist;** if you have run the stored procedure and verified that the status column = 3 for the user object that you wish to add, but the user is not able to log into SAP BPC, you can query the tables directly to verify that the records exist.

```
select * from <Appsetname>.dbo.tblusers
```

Figure 20: Example of querying the tblUsers SAP BPC System table



SQLQuery4.sql...OPCADM (72))*
SQLQuery3.sql...\BOPCADM (70))
SQLQuery2.sq...OPCADM (71))*
SQLQuery1.sq...OPCADM (69))*
Results
Messages

```
select * from Apshell.dbo.tblusers
```

	UserID	FullName	Domain	eMail	PasswordHash	RoleID	Language	Active	IMAccount	IMType	UniqueID
1	WIN-VV8ULDS219G\BOPCADM	WIN-VV8ULDS219G\BOPCADM				0		Y		0	WIN-VV8ULDS219G\BOPCADM
2	i064729-i9a94l2\i064729-i9a94l2\Test1	Test User 1	i064729-i9a94l2			0		Y		0	i064729-i9a94l2\i064729-i9a94l2\Test1
3	WIN-VV8ULDS219G\test3	test3	NULL			NULL	NULL	Y		0	WIN-VV8ULDS219G\test3
4	WIN-VV8ULDS219G\test2	test2	NULL			NULL	NULL	Y		0	WIN-VV8ULDS219G\test2
5	WIN-VV8ULDS219G\WIN-VV8ULDS219G\test1	Tst User1	WIN-VV8ULDS219G			0		Y		0	WIN-VV8ULDS219G\WIN-VV8ULDS219G\test1

Related Content

SAP BPC – IBM TDI Integration Project code:

SQL Script: Stored procedure – sp_bopc_tim

SQL Script: Create table script

IBM TDI Assembly Line: Project files

IBM TDI Documentation:

IBM Tivoli Directory Integrator Users Group

IBM Tivoli Directory Integrator How To

IBM Tivoli Directory Integrator

Learning TDI

Making your first Tivoli Directory Integrator AssemblyLine - a Video Tutorial

IBM Tivoli Directory Integrator: Getting Started Guide

Copyright