HP OpenView Business Process Insight

For the Windows® Operating System

Software Version: 02.10

Integration Training Guide - Monitoring Service Desk (Versions 4.5 and 5.x)

Document Release Date: Jan 2007
Software Release Date: Jan 2007

Documentation Updates

This manual’s title page contains the following identifying information:

- Software version number, which indicates the software version
- Document release date, which changes each time the document is updated
- Software release date, which indicates the release date of this version of the software

To check for recent updates, or to verify that you are using the most recent edition of a document, go to:

http://ovweb.external.hp.com/lpe/doc_serv/

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Please visit the HP OpenView support web site at:

http://www.hp.com/managementsoftware/support

This web site provides contact information and details about the products, services, and support that HP OpenView offers.

HP OpenView online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valuable support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit enhancement requests online
- Download software patches
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and log in. Many also require a support contract.

To find more information about access levels, go to:

http://www.hp.com/managementsoftware/access_level

To register for an HP Passport ID, go to:

http://www.managementsoftware.hp.com/passport-registration.html

Contents

1 Introduction

This training guide looks at the OpenView Service Desk Process Insight module, which provides template flows and adaptors to help you to monitor OpenView Service Desk (OVSD) using OpenView Business Process Insight (OVBPI).

This chapter provides an overview of what makes up the OVSD Process Insight module.

Prerequisites

To configure and/or extend the OVSD Process Insight module you need to be comfortable modeling OVBPI flows, configuring adaptors, configuring SQL triggers, and working with OVBPI dashboards.

This training guide assumes that you have read the following guides:

- *OVBPI Integration Training Guide - Modeling Flow*
- *OVBPI Integration Training Guide - Business Events*
- *OVBPI Integration Training Guide - Customizing the Business Process Dashboard*

OVSD Process Insight Components

With the OVSD Process Insight module, you are able to visualize your ITIL processes within OVBPI, and monitor how they are performing.

The OVSD Process Insight module consists of the following:

- A set of ITIL flow definitions:

  ```
  Service Calls
  Incidents
  Problems
  Changes
  Work Orders
  ```

- A set of adaptors

  One adaptor for each ITIL flow.

- A set of OVSD database triggers.
- A customized dashboard.

  You can use the OVBPI Business Process Dashboard, however, a customized dashboard, the OVSD Process Insight custom dashboard, is provided to give an example of how you can tailor a dashboard specifically for OVSD to focus on your ITIL processes.

Integration Templates

The OVSD Process Insight module provides five ITIL flow definitions and the associated adaptors, SQL triggers and customized dashboard. However, not every OVSD installation is going have the same ITIL flow definitions.

The OVSD Process Insight module is provided as a set of integration templates. It is fully expected that OVBPI Consultants are engaged to enhance and tailor the integration to match your OVSD installation.

How to customize the OVSD Process Insight module is covered in Chapter 6, Advanced Customization.

Supported OVSD Product Version

The OVSD Process Insight module supports either OVSD 4.5 (Service Pack 13 or greater) or OVSD 5.x. This manual covers both integrations.

Overall, the architecture of the integration is the same whether it is with OVSD 4.5 or OVSD 5.x. However, due to differences between OVSD 4.5 and OVSD 5.x, the actual steps required to integrate with OVBPI differ slightly. Chapter 3, Process Insight with OVSD 4.5 discusses the detailed steps required to integrate with OVSD 4.5. Chapter 4, Process Insight with OVSD 5.x discusses the detailed steps required to integrate with OVSD 5.x.

When it comes to customizing the integration, the basic concepts are also the same. However, when it comes to some of the advanced customizations, there are again differences between OVSD 4.5 and OVSD 5.x. That is why Chapter 6, Advanced Customization contains two worked examples - one for OVSD 4.5 and one for OVSD 5.x.

Moving from OVSD 4.5 to OVSD 5.x

If you have OVSD 4.5 flows currently active in your OVBPI installation, and you now want OVBPI to monitor OVSD 5.x flows, then you must follow these steps:

1. Use the OVBPI Intervention Client to completely remove all existing OVSD 4.5 flows, flow instances, data and data instances from the OVBPI Engine.
2. Run the OVBPI Modeler and delete the five OVSD flows:
   - Changes
   - Incidents
   - Problems
   - Service Calls
   - Work Orders
3. Follow the directions outlined in Chapter 4, Process Insight with OVSD 5.x to set up the monitoring of your OVSD 5.x machine.

The Custom Dashboard

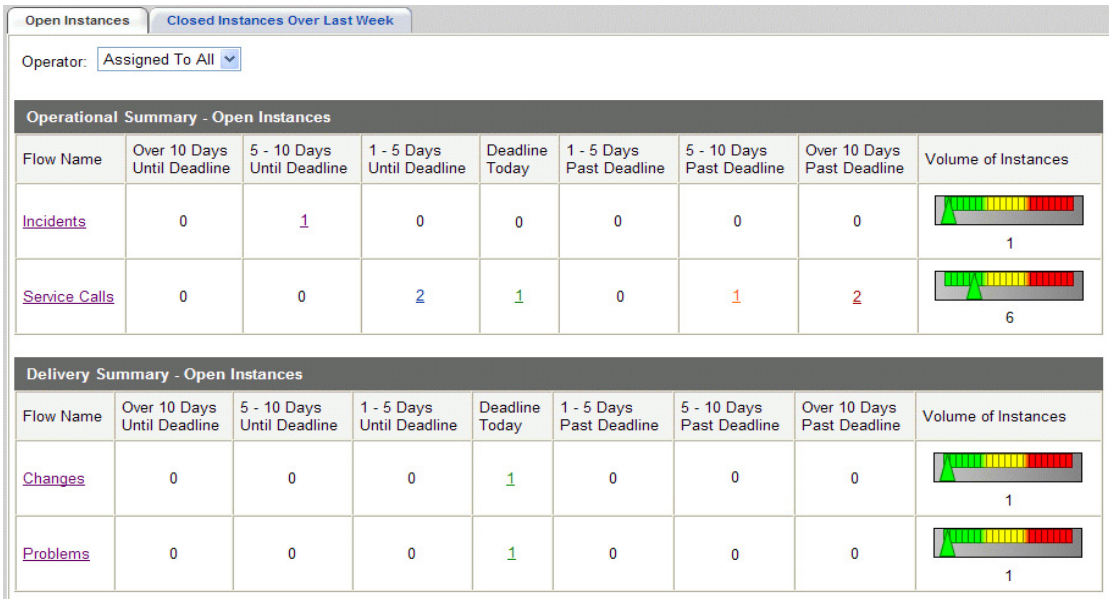
Although you can use the standard OVBPI Business Process Dashboard to view your ITIL flow instances within OVBPI, the customized dashboard provided with the OVSD Process Insight module presents the information in a way that is more focused around the five ITIL flows of the OVSD Process Insight module.

Main Page

The OVSD Process Insight custom dashboard allows you to see at a glance the overall status of your OVSD items.

The main page displays all items relative to their assigned deadline, as shown in Figure 1:

Figure 1 Main Page - Deadline Tracking



Operator: Assigned To All

Operational Summary - Open Instances

Flow Name	Over 10 Days Until Deadline	5 - 10 Days Until Deadline	1 - 5 Days Until Deadline	Deadline Today	1 - 5 Days Past Deadline	5 - 10 Days Past Deadline	Over 10 Days Past Deadline	Volume of Instances
Incidents	0	1	0	0	0	0	0	1
Service Calls	0	0	2	1	0	1	2	6

Delivery Summary - Open Instances

Flow Name	Over 10 Days Until Deadline	5 - 10 Days Until Deadline	1 - 5 Days Until Deadline	Deadline Today	1 - 5 Days Past Deadline	5 - 10 Days Past Deadline	Over 10 Days Past Deadline	Volume of Instances
Changes	0	0	0	1	0	0	0	1
Problems	0	0	0	1	0	0	0	1

where:

- You can see how many instances are due to be completed today (the `Deadline Today` column).

- You can see how many items were due to complete but have overrun by a number of days. You can also see how many items are coming up to their completion deadline.

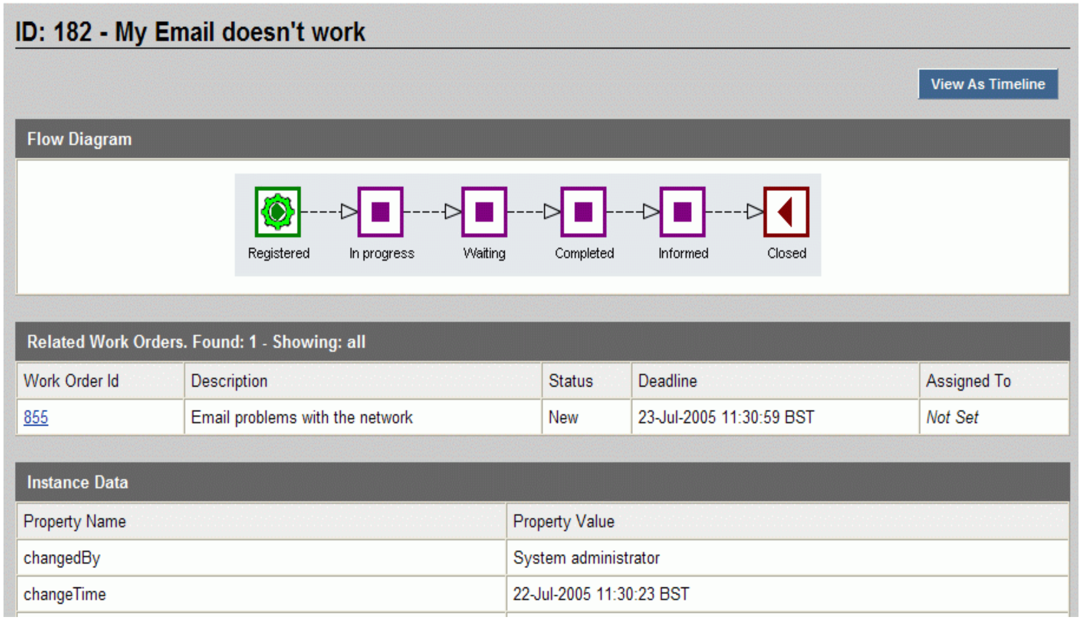
  For example, the Service Call listing shows that there are two calls where the deadline is in one to five days time, one call is due to be fixed today, one call is at least five days overdue, and two calls are at least ten days overdue.

- You can click on the individual numbers to see just those instances, and then drill into them to see further specific details.
- You can click on the flow names to see the details screens for each flow.
- You can filter the items by operator - to show only the items assigned to a particular operator.
- You can configure the individual column headings, and ranges, to be appropriate for your installation (see Chapter 5, Basic Customization).

The main screen (as shown in Figure 1 on page 13) shows the overall status of four of the five ITIL flows.

If you drill into a particular instance you can see how that instance is progressing, and it is at this level that you see any work orders, as shown in Figure 2:

Figure 2 Drill Down



Related Work Orders. Found: 1 - Showing: all

Work Order Id	Description	Status	Deadline	Assigned To
855	Email problems with the network	New	23-Jul-2005 11:30:59 BST	*Not Set*

Instance Data

Property Name	Property Value
changedBy	System administrator
changeTime	22-Jul-2005 11:30:23 BST

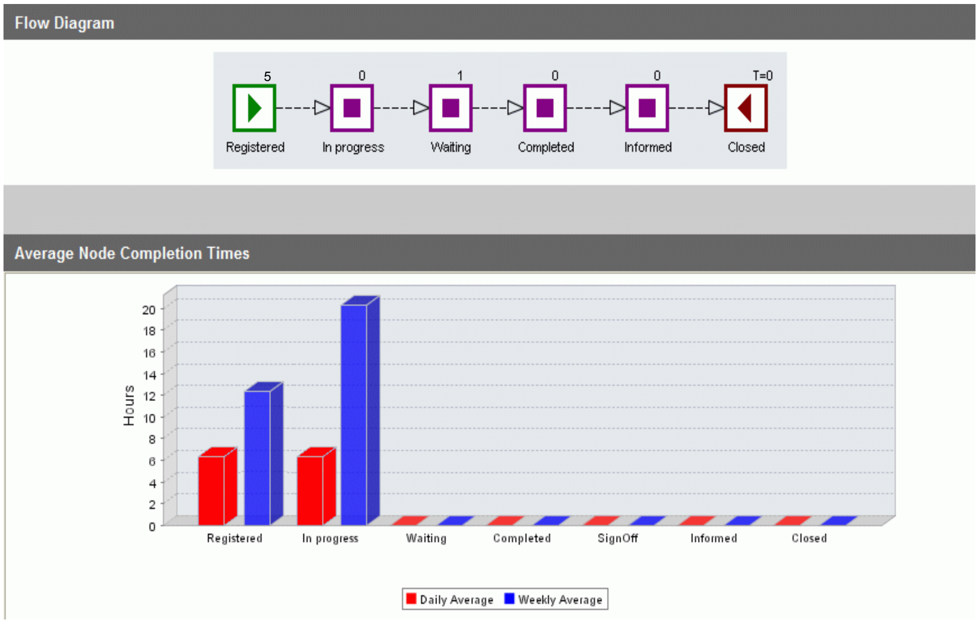
where:

- You see where the item is within its “flow”. It is currently at the `Registered` stage.
- If there are any related work items, they are listed here, and you can drill down into each work order to see how that is progressing.
- The example flow is for a Service Call in OVSD 4.5.

ITIL Flow Overview

From the main screen, if you click on the actual flow name (for example, `Service Calls`) it takes you to the overview page for that flow, as shown in Figure 3:

Figure 3 Overview Page - Service Calls



where:

- Each node in the flow diagram represents a status that a service call can have.
- The numbers above each node in the flow diagram show the number of service calls currently at that step in the process. So you can see how your items are distributed throughout the overall flow.
- The "Average Node Completion Times" bar graph shows the average time that a service call is spending in each of these steps.

  The graph shows both the daily average as well as a rolling weekly average.
- The example flow is for Service Calls in OVSD 4.5.

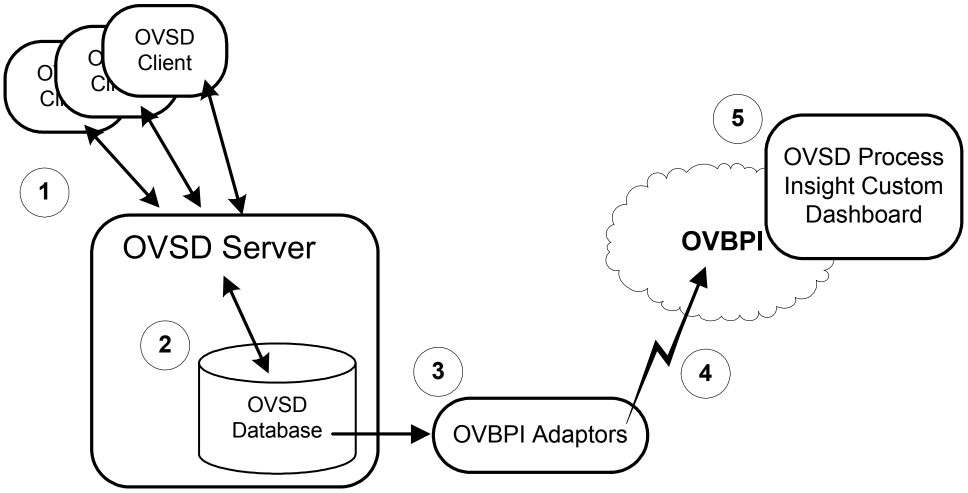
2 Architecture

This chapter looks at the architecture of the OVSD Process Insight module. This chapter highlights the “touch points” of the OVBPI Integration with OVSD. That is, it looks at where, and how, OVBPI connects into OVSD to be able to monitor the OVSD activity.

The Big Picture

The overall architecture is shown in Figure 4:

Figure 4 OVSD Process Insight Module - Overview



where:

1. OVSD Clients enter and update Service Calls, Problems, etc. in the normal way.
2. The OVSD Server maintains these updates in the OVSD Database.
3. OVBPI adaptors monitor these updates to the OVSD Database.
4. The OVBPI adaptors send all the updated information to the OVBPI server as business data events.

   OVBPI stores and collates these business events against your ITIL flows, in the normal way.
5. You use the OVSD Process Insight custom dashboard, from within a Web browser, to view your ITIL flows and monitor how they are performing.

Server Configurations - Local and Remote

The OVSD Process Insight module does not enforce any strategy as far as where you run OVBPI relative to OVSD. That is, they can both be installed on the same machine or separate machines. The choice is yours.

However, for a production system you would probably run OVBPI on a machine separate from the OVSD server.

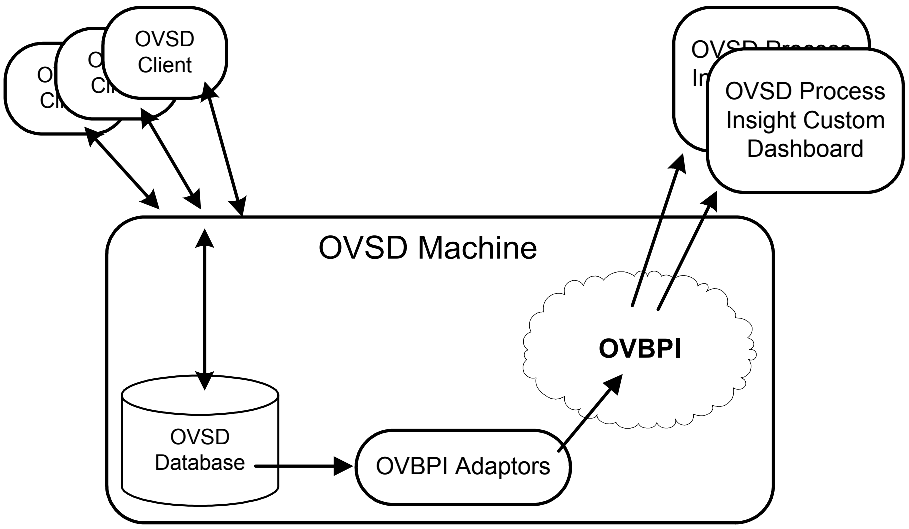
You then have the choice of where to run the OVBPI adaptors. These can run local to the OVSD database on the OVSD machine, or remotely from the OVSD database on the OVBPI machine.

So you have the following configuration strategies available to you:

- Install OVBPI and OVSD on the same machine; see Same Machine on page 20.
- Install OVBPI and the adaptors on a machine separate from the OVSD machine; see Separate Machines - Remote Adaptors on page 21
- Install OVBPI on one machine, and install the adaptors on the OVSD machine; see Separate Machines - Local Adaptors on page 22

Same Machine

Figure 5 Installation - Same Machine

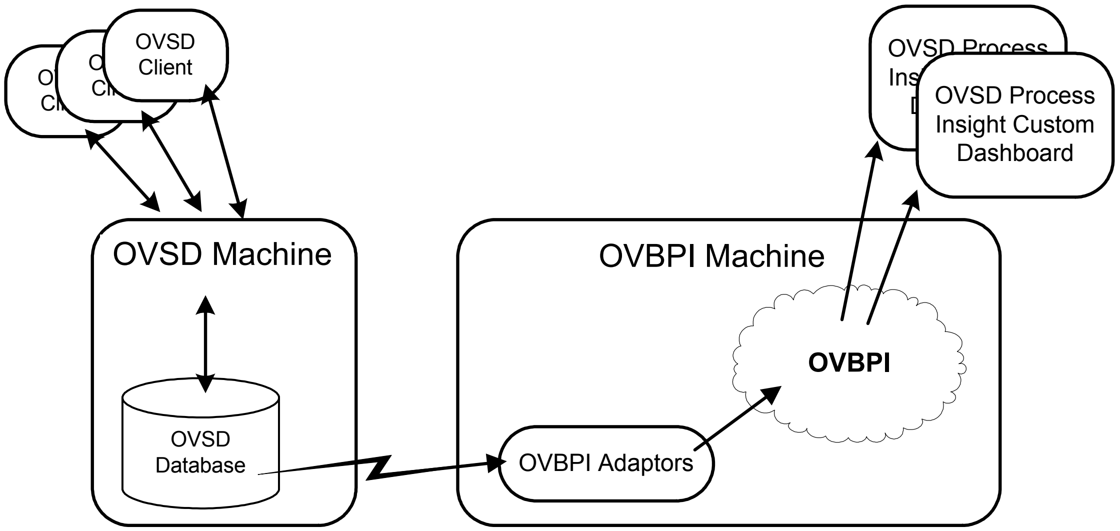


This is most likely to be the configuration you use when giving a demonstration, where you only have a single PC.

This configuration is not ideal for production use.

Separate Machines - Remote Adaptors

Figure 6 Installation - remote adaptors

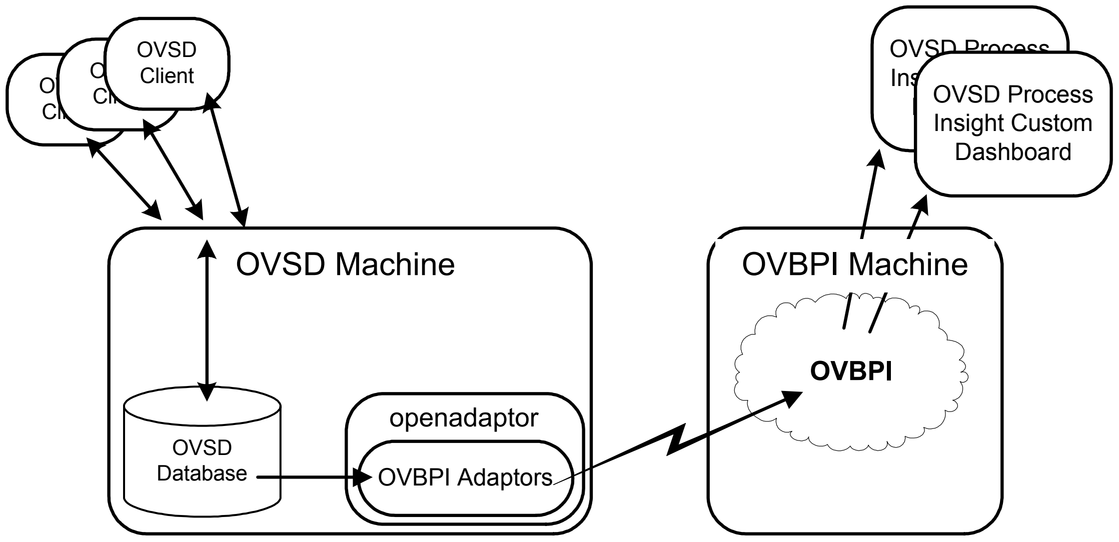


With OVBPI and OVSD on separate machines, you have the choice of where to run the actual OVBPI adaptors. The configuration shown in Figure 6 means that you do not need to separately install openadaptor on the OVSD machine. openadaptor in installed as part of the OVBPI installation, so the OVBPI adaptor automatically has the necessary environment to run.

You would need to test the network impact of the OVBPI adaptor issuing its SQL statements across the network to the OVSD database. This would be the deciding factor in choosing this installation architecture.

Separate Machines - Local Adaptors

Figure 7 Installation - local adaptors



If you run the OVBPI adaptors on the OVSD server then you also need to install openadaptor on the OVSD server. Refer to the *OVBPI Integration Training Guide - Business Events* for details of how to install a standalone version of openadaptor.

Obtaining the Data from OVSD

So how do the OVBPI adaptors monitor the OVSD database and extract the data to send to OVBPI? Let's expand step `3` of Figure 4 on page 18.

To provide reliable data feeds with minimal impact on the normal operation of OVSD, the extraction is implemented using SQL Triggers on a small number of OVSD database tables.

SQL Triggers

Consider for a moment how you might track a Service Call. When a Service Call is first created, you want to be told about it. Whenever this Service Call is subsequently modified, you want to be told about it.

Within the OVSD database, when a Service Call is created or modified, a new record is written to the Service Call history data table. Thus, by placing a single trigger on this history table, OVBPI is able to see each and every Service Call and any modifications.

When a new record is written to the Service Call history data table, the SQL trigger is executed. The trigger pulls out some key information from the record being inserted into the history table, and inserts this key information as a new record into a special OVBPI data table, called `ovbpi_item_change_hist`.

There are similar history tables for Incidents, Changes, Problems and Work Orders, and there are triggers for each of these tables.

SQL Triggers are placed on the following OVSD database tables:

- For OVSD 4.5, the list of data tables is as follows:
  - itsm_historylines_servicecall
  - itsm_historylines_incident
  - itsm_historylines_change
  - itsm_historylines_problem
  - itsm_historylines_workorder

- For OVSD 5.x, the list of data tables is as follows:
  - sd_historylines_servicecall
  - sd_historylines_change
  - sd_historylines_problem
  - sd_historylines_workorder
  - cdm_historylines_incident
  - cdm_incident2services

All the triggers write their output to the one table - the `ovbpi_item_change_hist` table.

OVBPI Item Change History Table

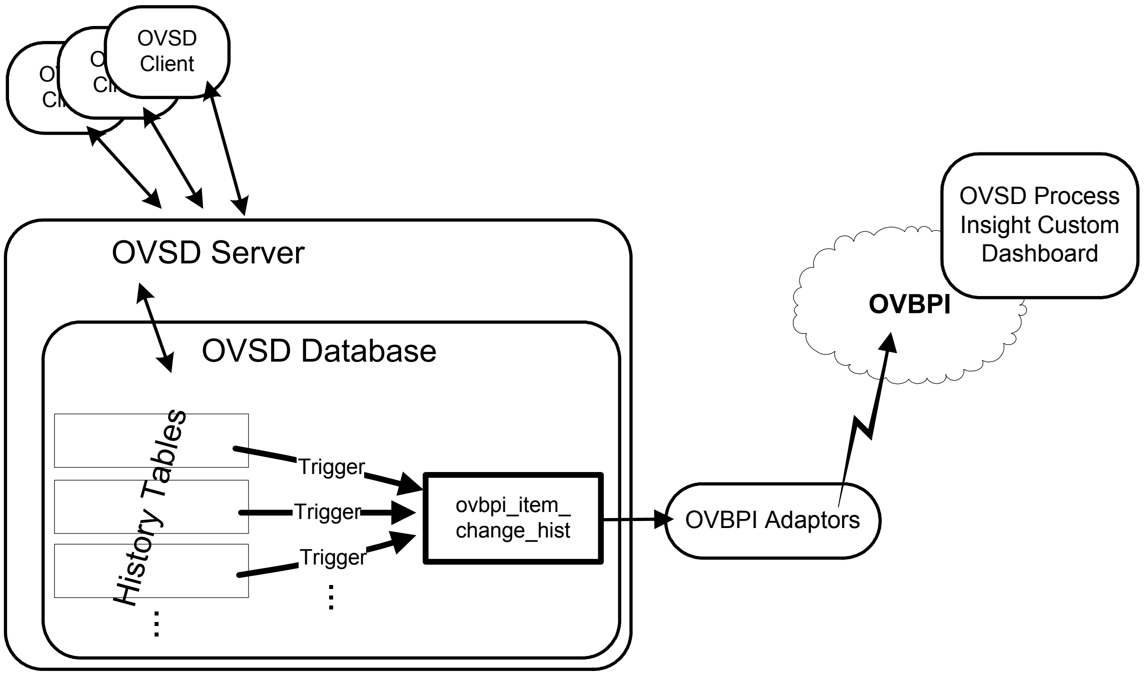
The OVSD Process Insight module installs a new data table called: `ovbpi_item_change_hist`. This table is a staging area (often called a "buffer table") for all the updates that happen to your Service Calls, Incidents, Changes, Problems and Work Orders. The `ovbpi_item_change_hist` buffer table needs to be within the OVSD database.

A set of OVBPI adaptors polls this buffer table, picking up all the updates from OVSD and sending them, as business data events, to OVBPI.

As each record is processed from this buffer table, the record is removed - so the `ovbpi_item_change_hist` should not grow without bound.

The overall architecture, with the SQL triggers, is shown in Figure 8:

Figure 8 OVSD Process Insight - SQL Triggers



where:

- As new Service Calls, Incidents, etc. come in, or are modified, the SQL triggers write key information into the `ovbpi_item_change_hist` buffer table.
- The OVBPI adaptors process the records in this `ovbpi_item_change_hist` buffer table, and send the details to OVBPI for monitoring purposes.

OVSD Views

The data records written into the `ovbpi_item_change_hist` buffer table are fairly small. Each record is basically just the internal OVSD Object ID (OID) of the item that has been modified, a timestamp of when this action occurred, and what type of modification this is. The execution of the SQL trigger needs to be efficient so as not to adversely affect the normal day-to-day running of the OVSD system.

This means that, as each OVBPI adaptor processes these records from the `ovbpi_item_change_hist` buffer table, the adaptor needs to enrich the data by using this OID to obtain additional information about each particular record. This additional data is obtained by accessing the OVSD data table `rep_codes_text` and the OVSD Reporting Views.

The OVSD Reporting Views are an optional feature within OVSD, but for the OVSD Process Insight module these views must exist.

List of Views Required

Of the views created when you generate them from within the OVSD Administrator Console, the following is the list of views used by the OVBPI adaptors:

- `v_historylineincident`
- `v_incident`
- `v_service` (OVSD 5.x only)
- `v_incident2service` (OVSD 5.x only)
- `v_historylineservicecall`
- `v_servicecall`
- `v_historylinechange`
- `v_change`
- `v_historylineworkorder`
- `v_workorder`
- `v_historylineproblem`
- `v_problem`
- `v_person`
- `v_priority`

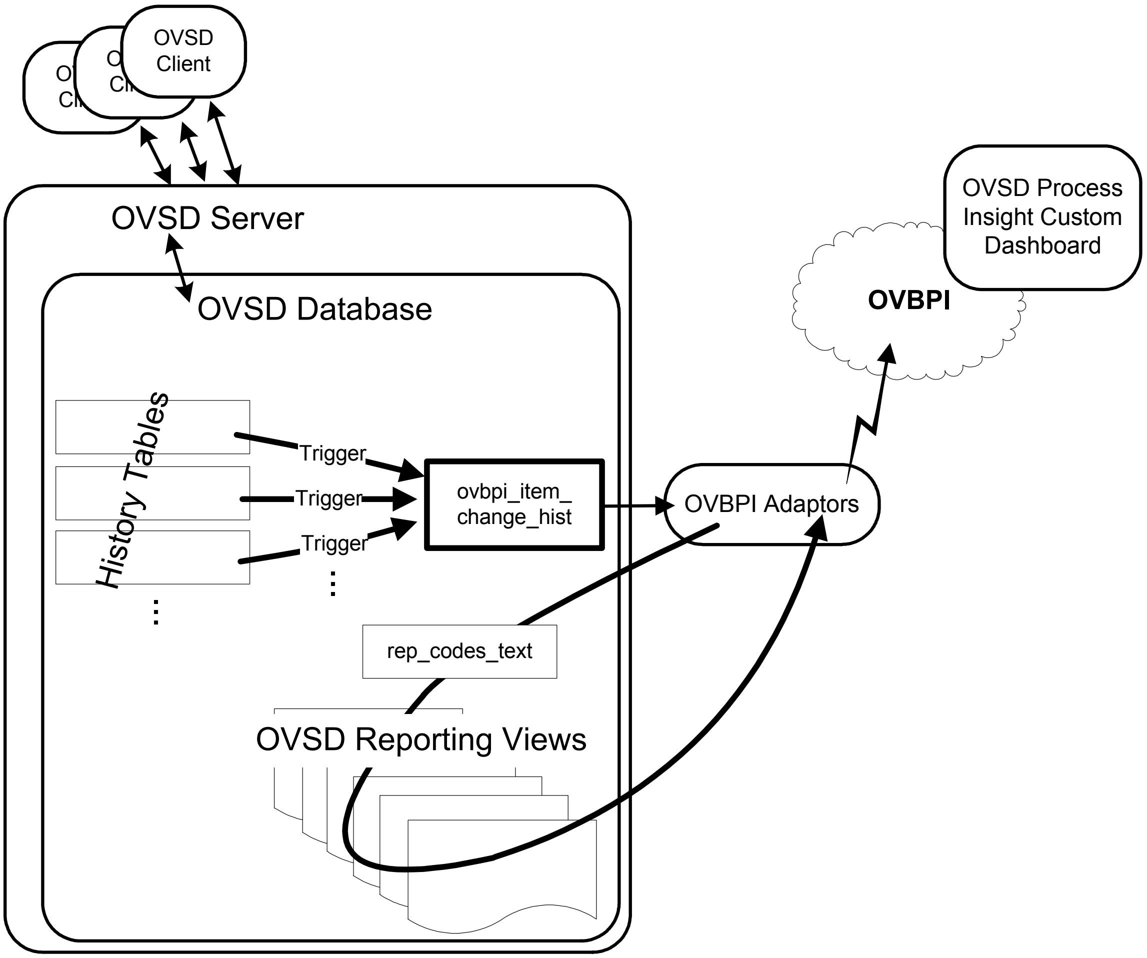
When you generate the views (from within the OVSD Administration Console), the localization options you choose can affect the names of these views listed above. This is all configurable when installing the OVBPI adaptors - see The Adaptors on page 36 for more details.

Detailed Architecture Diagram

The overall architecture of the OVSD Process Insight module can be detailed further by showing the use of the OVSD views.

The overall architecture diagram is shown in Figure 9.

Figure 9 OVSD Process Insight - in Detail



3 Process Insight with OVSD 4.5

This chapter explains how to configure and deploy the OVSD Process Insight module with OVSD 4.5 (SP13 or greater).

For details about using the OVSD Process Insight module with OVSD 5.x refer to Chapter 4, Process Insight with OVSD 5.x.

Module Files

The files that make up the OVSD Process Insight module are all located under your OVBPI installation directory:

- *`OVBPI-install-dir`*`\examples\bia\ServiceDeskFlows`

  This directory holds the ITIL flow definitions. These are ZIP files ready to be imported directly into the OVBPI Modeler.

- *`OVBPI-install-dir`*`\examples\bia\ServiceDeskAdaptors`

  This directory contains everything to do with your adaptors. It includes things such as adaptor property files, adaptor start and stop scripts, and SQL trigger definition scripts.

- *`OVBPI-install-dir`*`\nonOV\jakarta-tomcat-5.0.19\`
  `webapps\ovbpi_sd_dashboard2-10.war`

  This is the OVSD Process Insight custom dashboard. This WAR file is already expanded for you (under the `webapps\ovbpi_sd_dashboard2-10` directory) and ready for use.

The above files are all installed when you install OVBPI.

The next steps are to deploy each flow, configure the adaptors and use the OVSD Process Insight custom dashboard.

Deploy The Flows

To deploy the flows you need to carry out the following steps on your OVBPI server:

1. Start up the OVBPI components.
2. Run the OVBPI Modeler.
3. For each flow in the directory:

   `*OVBPI-install-dir*\examples\bia\ServiceDeskFlows`

   — Import the flow definition.

   — Deploy the flow definition.

Prepare the OVSD Installation

The OVSD Process Insight integration requires the OVSD Database Reporting Views to be created.

When you (re)generate the Database Reporting Views from within the OVSD Console, you can select various options regarding localization. These options determine the actual names used to create the views and the columns within the views. You can also set the time zone that is to be used when representing dates within these views. The options you select when (re)generating these views, affect the configuration you need to do later on when you come to configure the OVBPI adaptors.

Creating the OVSD Views

The OVSD reporting views are created from within the OVSD Administrator Console.

To start the OVSD Administrator Console:

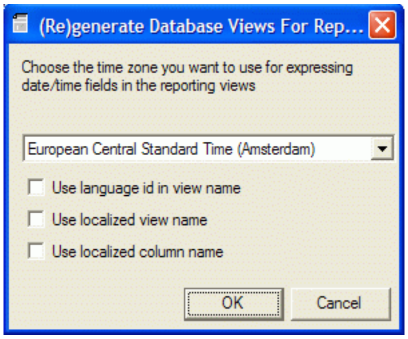
- Log in to the OVSD Client as an OVSD user with administrator capabilities.
- Then select: `Tools->System...`

From the OVSD Administrator Console:

1. Click on `System Panel` - in the left-hand pane.
2. Double-click on `Report Settings` - in the right hand pane.
3. On the `General` TAB, click on the `(Re)generate database views for reporting...` button (in the lower part of the dialog)

This then gives you a dialog box as follows:

Figure 10 OVSD Database Views Creation



(Re)generate Database Views For Rep...

Choose the time zone you want to use for expressing date/time fields in the reporting views

European Central Standard Time (Amsterdam)

- [ ] Use language id in view name
- [ ] Use localized view name
- [ ] Use localized column name

You then select your time zone, and any options for localized names, and click OK.

If you leave all three checkboxes empty (as shown in Figure 10), then the view names, and their column names, that are generated should match those provided by default in the `SD_Adaptor_Config.properties` file.

➤ You need to make a note of the time zone you select when (re)generating the OVSD views. You will need to set this time zone value when you come to configure the OVBPI adaptors, as described in SD_Adaptor_Config.properties on page 36.

Prepare the Adaptor Machine

You need to decide how you are going to run your adaptors. That is, whether to run the adaptors on the OVSD machine, or run them on the OVBPI machine.

Once you have decided which machine the adaptors will run on, you then need to set up the adaptor configuration files on that machine (see Export the Adaptor Configuration Files on page 34), and install openadaptor (see Install openadaptor on page 35).

Export the Adaptor Configuration Files

The adaptor configuration files are installed under the `*OVBPI-install-dir*\examples\bia\ServiceDeskAdaptors` directory on the OVBPI machine...however...you should not edit these files. The idea is to leave this `ServiceDeskAdaptors` directory in place and to take a copy and work in that copy directory. To make this easy, there is a script provided called: `sd_exportadaptors.bat`.

The `sd_exportadaptors.bat` script actually creates a copy of the `ServiceDeskAdaptors` directory, includes some additional OVBPI Java libraries, and creates a zip file containing a mini-environment that you can then take across to your adaptor machine. You are then able to configure and run your adaptors from there.

You run the `sd_exportadaptors.bat` script on the OVBPI machine as follows:

- Open a Console window on the OVBPI machine, and change directory into the `*OVBPI-install-dir*\examples\bia\ServiceDeskAdaptors` directory.
- Within this console window, set the two environment variables:
  - `OVBPI_ROOT`

    Set this to the installation directory for OVBPI on your machine.

    The default is: `C:\Program Files\HP Openview\OVBPI`
  - `JAVA_HOME`

    Set this to the installation directory for Java on your machine.

    The default is: `C:\java`

- Then run the script: `sd_exportadaptors.bat`

  This creates the file `ovbpi-sd-adaptors.zip` in your local directory.

You now take this `ovbpi-sd-adaptors.zip` file to the machine on which you intend to configure and run the openadaptor adaptors, and unzip it. You can unzip it to any location you choose.

➤ You should use this `ovbpi-sd-adaptors.zip` file even if you plan to run the adaptors on your OVBPI machine.

This manual refers to the unzipped location as the *`Work-dir`*.

The `ovbpi-sd-adaptors.zip` file contains all the files necessary to configure your adaptors. However, it does not contain openadaptor itself (see Install openadaptor on page 35).

Unix Specifics

If you need to unzip the `ovbpi-sd-adaptors.zip` file onto a Unix machine, you can use the `jar` command as follows:

```
$ cd destination-directory
$ jar -xvf ovbpi-sd-adaptors.zip
```

You then need to make the Unix scripts executable:

```
$ cd destination-directory/examples/bia/ServiceDeskAdaptors
$ chmod +x *.sh
```

Install openadaptor

If you are going to run the adaptors on the same machine as OVBPI then you already have openadaptor installed at the location:
`OVBPI-install-dir``\nonOV\openadaptor\1_6_5`.

If you are running the adaptors an a server that does not have OVBPI installed, then you need to install the openadaptor software. Refer to the *OVBPI Integration Training Guide - Business Events* for details on how to install a standalone version of openadaptor.

The Adaptors

Once you have unzipped the `ovbpi-sd-adaptors.zip` file, and installed openadaptor, you are ready to configure the adaptors.

It would be great if all the adaptors were configured and ready to run. However, because OVSD is quite customizable, there are a number of key items that you need to provide before the adaptors can work for your installation.

SD_Adaptor_Config.properties

All the key configuration items that can be specific to your OVSD installation are set up as properties within the file:

```
Work-dir\examples\bia\ServiceDeskAdaptors\
   SD_Adaptor_Config.properties.
```

➤ Remember, *`Work-dir`* refers to the directory in which you unzipped the `ovbpi-sd-adaptors.zip` file. This is on the machine that you are going to be running the openadaptor adaptors.

You must edit the `SD_Adaptor_Config.properties` file, and go through **every** property, making sure that they are correct for your OVSD installation.

When you first look at the file you may be surprised at the large number of configurable properties. Let's have a look at them.

The properties basically fall into the following categories:

1. OVSD DB View and Column names

   These make up the bulk of the file. These are the names of each of the OVSD database views that the adaptors need to access, and the column names within these OVSD views. (See Creating the OVSD Views on page 32 for more details, as there is a way to set up OVSD such that the bulk of the database column names match.)

   You need to go through and check every property, and ensure they are correct for your OVSD database.

For example:

The default `SD_Adaptor_Config.properties` file has the line entry:

```
INCIDENT_VIEW           = v_incident
```

This is saying that the property `INCIDENT_VIEW` needs to be set to the name of the incident view in your OVSD database. On your installation it might be named `v1033_incident`, or it could be something else. You need to find the name for the incident view in your database and set the property in the `SD_Adaptor_Config.properties` file to that name.

2. OVBPI host name details
3. OVSD database connection details

   These are details such as the name of the OVSD Database, the JDBC driver, JDBC connection URL details, the user name and password.

   There are two blocks of OVSD database properties in the configuration file - one for MSSQL and one for Oracle. You comment in/out the necessary lines.

➤ Note that the OVSD database password needs to be specified in encoded form. To produce an encoded password, run the command:

```
java -classpath Work-dir\java\bia-event.jar
               org.openadaptor.adaptor.util.Encoder password
```

(all on one command line)

where:

— You supply the full path for the `bia-event.jar` file

— You supply your password in plain text

— The encoded version of the password is output to the command window display. You then paste this result into your adaptor configuration file.

4. The OVSD database time zone

   The `DB_TIME_ZONE` property needs to be set to the time zone that you chose when you (re)generated the OVSD Database Views. (See Creating the OVSD Views on page 32.)

   Refer to Valid Time Zones on page 118 to see the list of possible values that you can set for the `DB_TIME_ZONE` property.

5. Adaptor Remote Control Ports

   These values just need to be numbers of ports available on your system.

6. Adaptor Polling Period

   Specified in milliseconds. Defaults to 300000 (5 minutes.)

7. Adaptor Commit SQL command

   This property lets you set the behavior for the adaptors once they have processed records from the `ovbpi_item_change_hist` table.

If you do not set all properties in `SD_Adaptor_Config.properties` correctly for your installation, you may not realize that there is a problem until you actually have everything up and running; see Run Time Adaptor Errors on page 113 for an example of errors that might occur.

➤ It is essential to take your time setting the `SD_Adaptor_Config.properties` file and get all the values correct for your installation.

Generate the Adaptors and Triggers

Once you have set up the `SD_Adaptor_Config.properties` file, you are ready to generate the adaptor configuration files and the SQL trigger scripts.

On the machine where you intend to run the adaptors, you run the script `sd_createadaptors(.bat/.sh)` from the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors` directory, as follows:

- Open a Console window and change directory to the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors` directory.

  For Unix, you need to be logged on as the **root** user.

- Within this console window, set the following environment variables:
  - `OVBPI_ROOT`

    Although the variable is called `OVBPI_ROOT`, it needs to be set to the mini-environment that you installed when you unzipped the `ovbpi-sd-adaptors.zip` file; that is, the `Work-dir` directory.

    Do not set this to your OVBPI installation, if you have one on the machine, as this can cause the `sd_createadaptors` script to fail.
  - `OA_ROOT`

    Set this to the installation directory for openadaptor on your machine.

    The default is: *`OVBPI_ROOT`*`\nonOV\openadaptor\1_6_5`
  - `JAVA_HOME`

    Set this to the installation directory for Java on your machine.

    The default is: `C:\java`
- Then run the script: `sd_createadaptors(.bat/.sh)`

  The script reads the template adaptor configuration files, and SQL trigger files (found under the `template` subdirectory), and substitutes the property values as specified in your `SD_Adaptor_Config.properties` file.

The `sd_createadaptors` script creates a number of files in the `*Work-dir*\examples\bia\ServiceDeskAdaptors` directory, including:

- The adaptor configuration files:

  ```
  SD_Poll_ServiceCall.props
  SD_Poll_Incident.props
  SD_Poll_Change.props
  SD_Poll_Problem.props
  SD_Poll_WorkOrder.props
  ```

  ...and the adaptor log4j logging configuration files:

  ```
  ServiceCallAdaptor_log4j.props
  IncidentAdaptor_log4j.props
  ChangeAdaptor_log4j.props
  ProblemAdaptor_log4j.props
  WorkOrderAdaptor_log4j.props
  ```

- The database trigger files:

  ```
  CreateServiceDeskTriggers.sql
  DropServiceDeskTriggers.sql
  ```

- The Windows-Service/Unix-Daemon configuration files:

  ```
  OVBPISDServiceCallAdaptor.cfg
  OVBPISDIncidentAdaptor.cfg
  OVBPISDChangeAdaptor.cfg
  OVBPISDProblemAdaptor.cfg
  OVBPISDWorkOrderAdaptor.cfg
  ```

  ...and the following additional files on a Unix system:

  ```
  OVBPISDServiceCallAdaptorWrapper.sh
  OVBPISDIncidentAdaptorWrapper.sh
  OVBPISDChangeAdaptorWrapper.sh
  OVBPISDProblemAdaptorWrapper.sh
  OVBPISDWorkOrderAdaptorWrapper.sh
  ovbpisd.sh
  ovbpisd.cfgsh
  ```

The `sd_createadaptors` script also installs the five resulting adaptors as windows services, if running on a PC, or Unix daemons, if running on a Unix machine; see Restartable Services/Daemons on page 120 for more information about how to enable these services to be restartable.

Regenerating the Adaptors and Triggers

If you need to make alterations to any of the properties in your `SD_Adaptor_Config.properties` file, simply re-run the `sd_createadaptors` script, and the changes are reflected throughout all the regenerated files.

Install the Triggers

The SQL triggers need to be installed on the OVSD server's database.

After running the `sd_createadaptors` script, you have two new SQL files in the `*Work-dir*\examples\bia\ServiceDeskAdaptors` directory:

- `CreateServiceDeskTriggers.sql`

  The script to create the necessary triggers for the OVSD Process Insight integration.

- `DropServiceDeskTriggers.sql`

  A script for removing the OVSD Process Insight triggers.

To install the triggers you need to logon to the OVSD database as an administrator, and execute the following script:

```
CreateServiceDeskTriggers.sql
```

The `CreateServiceDeskTriggers.sql` script can be run from within your favorite SQL utility.

For example: (for MSSQL)

```
isql -U sduser -P sd -i CreateServiceDeskTriggers.sql
```

where:

- The user name `sduser` is the Service Desk administrative user.
- The password in this example is `sd`.

Verify OVSD with the Triggers

Having installed the OVSD Process Insight triggers into the OVSD database, it is important to verify that the triggers work and that the SD Client is still able to create and/or modify items such as Service Calls, Incidents, Changes, Problems and Work Orders.

Enter Some Dummy Data

In a development installation you could simply create a new item of each type - Service Call, Incident, Change, Problem and Work Order.

If the triggers are installed correctly there should be no database errors when creating any of these new items.

After creating each new item you should see new data records written to the `ovbpi_item_change_hist` table. You also see one or more records written to this database table for each new item you create.

If you are able to create new items within your SD Client and you see data being written to the `ovbpi_item_change_hist` table, then the triggers are installed correctly and your OVSD installation is functioning correctly.

If you are working on a production OVSD system you probably do not wish to create “dummy” items. That’s fine. As Calls/Incidents/etc. come in, your triggers are actioned and this verifies that everything is working.

Removing Process Insight Triggers

If you encounter problems and you are unable to create items from within your SD Client, you need to drop all the OVSD Process Insight triggers whilst you try and work out what is wrong with them.

To help you drop the triggers you can use the script: `DropServiceDeskTriggers.sql`.

Start the Adaptors

Now that the triggers are in place and working, you are able to start up the adaptors. The adaptors are all configured to run as either Windows-Services, or Daemons, depending on the operating system.

To start the adaptors:

1. Open a console window
2. Change directory to `Work-dir\examples\bia\ServiceDeskAdaptors`
3. Run: `sd_startadaptors(.bat/.sh)`

   This starts all adaptors.

Each adaptor is configured to log trace output to files within the `Work-dir\data\log` directory. For example, the Service Call adaptor outputs trace information to the log file called `OVBPISDServiceCallAdaptor.log`.

➤ Note that the adaptors append to their respective trace file. So if you stop and restart an adaptor, the trace output is **appended** to the end of the existing log file.

Each adaptor is configured to auto-start when the machine reboots. For more details about configuring the behavior of these adaptors see Restartable Services/Daemons on page 120.

Stop All Adaptors

The script `sd_stopadaptors(.bat/.sh)` is used to stop all the adaptors.

The Custom Dashboard

With the triggers installed and the adaptors running, you can run the standard OVBPI Business Process Dashboard and you are able to see your five ITIL flows.

Indeed, if you have entered any new Service Calls (as suggested in the verification step Verify OVSD with the Triggers on page 42) then you should be able to see them listed in the OVBPI Business Process Dashboard.

The OVBPI Business Process Dashboard helps you to test that things are working and that changes in OVSD are indeed making it over to OVBPI. However, there is a custom dashboard provided as part of the OVSD Process Insight module. The OVSD Process Insight custom dashboard presents a tailored view of the five ITIL flows.

What the Custom Dashboard Displays?

The adaptors monitor the following changes:

- Status changes.
- Deadline changes.
- Assigned Person changes.
- Priority changes.

As these types of changes occur in OVSD, you see them reflected in the OVBPI ITIL flows within your OVSD Process Insight custom dashboard ..**however**... a flow instance does not occur until an item has changed its state! That is, an OVBPI flow instance is not created for an item until a state change has occurred. Once an item **has changed state,** a flow instance is created and OVBPI then continues to monitor all changes for this item.

So, if you are testing out that changes in the SD Client do indeed cause changes to appear in the OVSD Process Insight custom dashboard, don't forget that until an item has changed its state (for example, changed from `Registered` to `In progress` - for a service call), it does not appear in the custom dashboard. It does not appear in either the OVBPI Business Process Dashboard or the OVSD Process Insight custom dashboard.

Run the Custom Dashboard

The OVSD Process Insight custom dashboard is already installed under the `webapps` directory and is ready for use...

To run the OVSD Process Insight custom dashboard, point a Web browser at the URL:

```
http://hostname:44080/ovbpi_sd_dashboard2-10
```

where:

- *`hostname`* is the host name of your OVBPI machine
- *`44080`* is the default port for the Servlet Engine

For example:

```
http://localhost:44080/ovbpi_sd_dashboard2-10
```

This works if OVBPI is installed on the same machine as your Web browser.

Logging On

When OVBPI is first installed, no logon is required to use the OVSD Process Insight custom dashboard.

The OVBPI Administration Console allows the administrator to specify whether you require people to give a username and password when running the OVSD Process Insight custom dashboard. You then have a choice of authorization mechanisms. See the *OVBPI System Administration* guide for more details.

Set the SD Version Flag

The OVSD Process Insight custom dashboard needs to know whether it is working against a set of flows that are monitoring data from OVSD 4.5 or OVSD 5.x.

There is a property setting, called `SDVersion`, within the OVSD Process Insight custom dashboard's configuration file, `SD_DashboardConfig.properties`. This property needs to be set to `4.5` as follows:

```
SDVersion=4.5
```

At installation time, the `SDVersion` property defaults to the value `4.5`.

Refer to The Configuration Properties File on page 74 for details of how to locate and edit the OVSD Process Insight custom dashboard's configuration file.

OVBPI Engine Instance Cleaner

The main page of the OVSD Process Insight custom dashboard allows you to select the tab labelled: `Closed Instances Over Last Week`. However, this `Closed Instances Over Last Week` tab requires that your OVBPI Business Impact Engine instance cleaner is configured to leave at least one week's (seven days) worth of completed flow instances in the OVBPI database.

For example, if your OVBPI Business Impact Engine instance cleaner is cleaning out completed flow instances after only 3 days, then clicking on the `Closed Instances Over Last Week` tab is only able to present the last 3 days worth of data.

You should set the OVBPI Business Impact Engine instance cleaner to clean out completed instances after `7` days or more.

Summary of Main Steps

Here is a checklist summarizing the main steps, outlined above, to set up the OVBPI Process Insight module:

On the OVBPI Machine

- Locate the OVBPI Process Insight files.
- Deploy the ITIL flows.
- Use `sd_exportadaptors.bat` to export the adaptor environment/files (creating the `ovbpi-sd-adaptors.zip` file).

On the OVSD Machine

- Generate the OVSD Database Reporting Views.

On the Machine running the Adaptors

- Unzip the `ovbpi-sd-adaptors.zip` file.
- Install openadaptor.
- Set up all installation specific properties by editing `SD_Adaptor_Config.properties`
- Create the adaptors by running the script `sd_createadaptors`

On the OVSD Machine

- Install the triggers into the OVSD database.

On the Machine running the Adaptors

- Run the adaptors by using the script `sd_startadaptors`.

On the OVBPI Machine

- Use the OVSD Process Insight custom dashboard to view your ITIL flows.

4 Process Insight with OVSD 5.x

This chapter explains how to configure and deploy the OVSD Process Insight module with OVSD 5.x.

For details about using the OVSD Process Insight module with OVSD 4.5 refer to Chapter 3, Process Insight with OVSD 4.5.

Module Files

The files that make up the OVSD Process Insight module are all located under your OVBPI installation directory:

- `*OVBPI-install-dir*\examples\bia\ServiceDeskFlows\SD5.*x*`

  The ITIL flow definitions are held under the appropriate directory. For example, the flows for OVSD 5.0 are held under the `ServiceDeskFlows\SD5.0` directory. The flows for OVSD 5.1 are held under the `ServiceDeskFlows\SD5.1` directory.

  The ITIL flows are ZIP files ready to be imported directly into the OVBPI Modeler.

- `*OVBPI-install-dir*\examples\bia\ServiceDeskAdaptors`

  This directory contains everything to do with your adaptors. It includes things such as adaptor property files, adaptor start and stop scripts, and SQL trigger definition scripts.

- `*OVBPI-install-dir*\nonOV\jakarta-tomcat-5.0.19\webapps\ovbpi_sd_dashboard2-10.war`

  This is the OVSD Process Insight custom dashboard. This WAR file is already expanded for you (under the `webapps\ovbpi_sd_dashboard2-10` directory) and ready for use.

The above files are all installed when you install OVBPI.

The next steps are to deploy each flow, configure the adaptors and use the OVSD Process Insight custom dashboard.

Deploy The Flows

To deploy the flows you need to carry out the following steps on your OVBPI server:

1. Start up the OVBPI components.
2. Run the OVBPI Modeler.
3. For each flow in the appropriate directory:

   `OVBPI-install-dir\examples\bia\ServiceDeskFlows\SD5.x`

   — Import the flow definition.

   — Deploy the flow definition.

➤ Remember to use the correct ITIL flows definitions for your version of OV Service Desk. For OVSD 5.0, the flows are found in the `SD5.0` directory. For OVSD 5.1, the flows are found in the `SD5.1` directory.

Prepare the OVSD Installation

In a while you will install the necessary database triggers into OVSD. These triggers will be placed on the following OVSD history tables:

- sd_historylines_servicecall
- sd_historylines_change
- sd_historylines_problem
- sd_historylines_workorder
- cdm_historylines_incident
- cdm_incident2services

These history tables exist within the OVSD 5.x database at installation time, however, by default they are not being written into.

The OVSD Process Insight integration requires that you enable OVSD Auditing such that the necessary information is maintained by OVSD within these history tables.

Enable OVSD DB Auditing

You need to enable Auditing for each of the five OVSD objects that the OVSD Process Insight integration is monitoring. And for each object, you need to specify which attributes you wish to have audited.

To enable auditing:

1. Log in to the OVSD Client as an OVSD user with administrator capabilities.
2. Click on `Users & Security` within the `OV Configuration` section in the left-hand pane.

3. Expand the entry called: `Audit`
4. Click on `Audit Rules`

   In the right-hand pane you will see a list of OVSD objects.

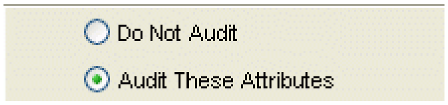
5. You now need to go through enabling auditing, and specifying the required attributes, for the five objects as follows:
   - Change object; see Change on page 53
   - Incident object; see Incident on page 54
   - Problem object; see Problem on page 54
   - Service Call object; see Service Call on page 55
   - Work Order object; see Work Order on page 55

Change

1. Double-click on `Change`

In the `Audit rules` window that pops-up:

2. Ensure that the `Audit These Attributes` radio button is checked, as follows:



Do Not Audit [✓ Audit These Attributes]

3. Then scroll through the list of attributes below and place a tick in the following:

```
Assignment;To Person
Deadline
Priority
Status
```

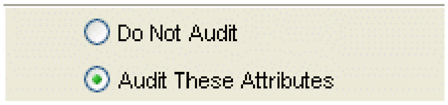
4. Select: `File->Save and Close`

Incident

1. Double-click on `Incident`

In the `Audit rules` window that pops-up:

2. Ensure that the `Audit These Attributes` radio button is checked, as follows:



Do Not Audit

[✓ Audit These Attributes]

3. Then scroll through the list of attributes below and place a tick in the following:

```
Assignment;To Person
Deadline
Priority
State
```

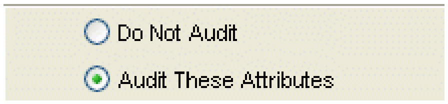
4. Select: `File->Save and Close`

Problem

1. Double-click on `Problem`

In the `Audit rules` window that pops-up:

2. Ensure that the `Audit These Attributes` radio button is checked, as follows:



Do Not Audit

[✓ Audit These Attributes]

3. Then scroll through the list of attributes below and place a tick in the following:

```
Assignment;To Person
Deadline
Priority
Status
```

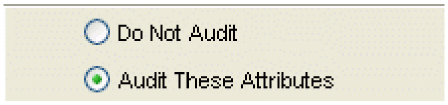
4. Select: `File->Save and Close`

Service Call

1. Double-click on `Service Call`

In the `Audit rules` window that pops-up:

2. Ensure that the `Audit These Attributes` radio button is checked, as follows:



○ Do Not Audit

● Audit These Attributes

3. Then scroll through the list of attributes below and place a tick in the following:

```
Assignment;To Person
Deadline
Priority
Status
```

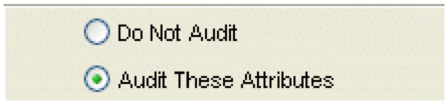
4. Select: `File->Save and Close`

Work Order

1. Double-click on `Work Order`

In the `Audit rules` window that pops-up:

2. Ensure that the `Audit These Attributes` radio button is checked, as follows:



○ Do Not Audit

● Audit These Attributes

3. Then scroll through the list of attributes below and place a tick in the following:

```
Assignment;To Person
Deadline
Priority
Status
```

4. Select: `File->Save and Close`

You now need to regenerate the OVSD Database Reporting Views. So follow the steps outlined in OVSD Database Reporting Views on page 56

OVSD Database Reporting Views

The OVSD Process Insight integration requires the use of the OVSD Database Reporting Views. Once you have selected the OVSD attributes to be audited, you need to generate (or regenerate) the OVSD Database Reporting Views.

When you (re)generate the Database Reporting Views from within the OVSD Console, you can select various options regarding localization. These options determine the actual names used to create the views and the columns within the views. You can also set the time zone that is to be used when representing dates within these views. The options you select when (re)generating these views, affect the configuration you need to do later on when you come to configure the OVBPI adaptors.

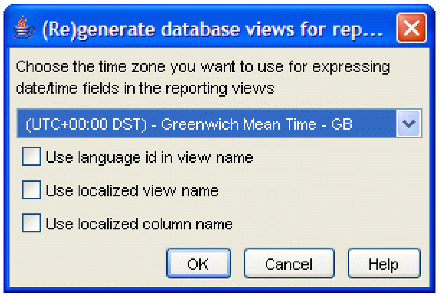
Creating the OVSD Views

To (re)generate the OVSD reporting views:

1. Log in to the OVSD Client as an OVSD user with administrator capabilities.
2. Click on `System settings` within the `OV Configuration` section in the left-hand pane.
3. Double-click on `Report Settings` in the right-hand pane.
4. On the `General` TAB, click on the `(Re)generate database views for reporting...` button (in the lower part of the dialog)

   This then gives you a dialog box as shown in Figure 11:

Figure 11 OVSD Database Views Creation


(Re)generate database views for rep...

Choose the time zone you want to use for expressing date/time fields in the reporting views

(UTC+00:00 DST) - Greenwich Mean Time - GB

☐ Use language id in view name

☐ Use localized view name

☐ Use localized column name

You then select your time zone, and any options for localized names, and click OK.

If you leave all three checkboxes empty (as shown in Figure 11), then the view names, and their column names, that are generated should match the defaults provided in the OVSD Process Insight integration.

➤ You need to make a note of the time zone you select when (re)generating the OVSD views. You will need to set this time zone value when you come to configure the OVBPI adaptors, as described in SD5.0_Adaptor_Config.properties on page 60.

Oracle 10

If your OVSD database is Oracle 10 then you may encounter some performance issues later on when you come to run the OVSD Process Insight adaptors. The solution for this is discussed in Oracle 10 Performance Issues on page 116.

Prepare the Adaptor Machine

You need to decide how you are going to run your adaptors. That is, whether to run the adaptors on the OVSD machine, or run them on the OVBPI machine.

Once you have decided which machine the adaptors will run on, you then need to set up the adaptor configuration files (see Export the Adaptor Configuration Files on page 58) on that machine, and install openadaptor (see Install openadaptor on page 59).

Export the Adaptor Configuration Files

The adaptor configuration files are installed under the `*OVBPI-install-dir*\examples\bia\ServiceDeskAdaptors` directory on the OVBPI machine...however...you should **not** edit these files. The idea is to leave this `ServiceDeskAdaptors` directory in place and to take a copy and work in that copy directory. To make this easy, there is a script provided called: `sd_exportadaptors.bat`.

The `sd_exportadaptors.bat` script actually creates a copy of the `ServiceDeskAdaptors` directory, includes some additional OVBPI Java libraries, and creates a zip file containing a mini-environment that you can take across to your adaptor machine. You are then able to configure and run your adaptors from there.

You run the `sd_exportadaptors.bat` script on the OVBPI machine, as follows:

- Open a Console window on the OVBPI machine, and change directory into the `*OVBPI-install-dir*\examples\bia\ServiceDeskAdaptors` directory.
- Within this console window, set the two environment variables:
  - `OVBPI_ROOT`

    Set this to the installation directory for OVBPI on your machine.

    The default is: `C:\Program Files\HP Openview\OVBPI`
  - `JAVA_HOME`

    Set this to the installation directory for Java on your machine.

    The default is: `C:\java`

- Then run the script: `sd_exportadaptors.bat`

  This creates the file `ovbpi-sd-adaptors.zip` in your local directory.

You now take this `ovbpi-sd-adaptors.zip` file to the machine on which you intend to configure and run the openadaptor adaptors, and unzip it. You can unzip it to any location you choose.

➤ You should use this `ovbpi-sd-adaptors.zip` file even if you plan to run the adaptors on your OVBPI machine.

This manual refers to the unzipped location as the *`Work-dir`*.

The `ovbpi-sd-adaptors.zip` file contains all the files necessary to configure your adaptors. However, it does not contain openadaptor itself (see Install openadaptor on page 59).

Unix Specifics

If you need to unzip the `ovbpi-sd-adaptors.zip` file onto a Unix machine, you can use the `jar` command as follows:

```
$ cd destination-directory
$ jar -xvf ovbpi-sd-adaptors.zip
```

You then need to make the Unix scripts executable:

```
$ cd destination-directory/examples/bia/ServiceDeskAdaptors
$ chmod +x *.sh
```

Install openadaptor

If you are going to run the adaptors on the same machine as OVBPI then you already have openadaptor installed at the location:
`OVBPI-install-dir``\nonOV\openadaptor\1_6_5`.

If you are running the adaptors an a server that does not have OVBPI installed, then you need to install the openadaptor software. Refer to the *OVBPI Integration Training Guide - Business Events* for details on how to install a standalone version of openadaptor.

The Adaptors

Once you have unzipped the `ovbpi-sd-adaptors.zip` file, and installed openadaptor, you are ready to configure the adaptors.

➤ Note: The configuration of the adaptors is the same for all 5.x versions of OVSD.

It would be great if all the adaptors were configured and ready to run. However, because OVSD is quite customizable, there are a number of key items that you need to provide before the adaptors can work for your installation.

SD5.0_Adaptor_Config.properties

All the key configuration items that can be specific to your OVSD 5.x installation are set up as properties within the file:

```
Work-dir\examples\bia\ServiceDeskAdaptors\
   SD5.0_Adaptor_Config.properties.
```

➤ Remember, *`Work-dir`* refers to the directory in which you unzipped the `ovbpi-sd-adaptors.zip` file. This is on the machine that you are going to be running the openadaptor adaptors.

You must edit the `SD5.0_Adaptor_Config.properties` file, and go through **every** property, making sure that they are correct for your OVSD installation.

When you first look at the file you may be surprised at the large number of configurable properties. Let's have a look at them.

The properties basically fall into the following categories:

1. OVSD DB View and Column names

   These make up the bulk of the file. These are the names of each of the OVSD database views that the adaptors need to access, and the column names within these OVSD views. (See OVSD Database Reporting Views on page 56 for more details, as there is a way to set up OVSD such that the bulk of the database column names match.)

You need to go through and check every property, and ensure they are correct for your OVSD database.

For example:

> The default `SD5.0_Adaptor_Config.properties` file has the line entry:
>
> ```
>  INCIDENT_VIEW             = v_incident
> ```
>
> This is saying that the property `INCIDENT_VIEW` needs to be set to the name of the incident view in your OVSD database. On your installation it might be named `v1033_incident`, or it could be something else. You need to find the name for the incident view in your database and set the property in the `SD5.0_Adaptor_Config.properties` file to that name.

2. OVBPI host name details
3. OVSD database connection details

   These are details such as the name of the OVSD Database, the JDBC driver, JDBC connection URL details, the user name and password.

   There are two blocks of OVSD database properties in the configuration file - one for MSSQL and one for Oracle. You comment in/out the necessary lines.

➤ Note that the OVSD database password needs to be specified in encoded form. To produce an encoded password, run the command:

```
java -classpath Work-dir\java\bia-event.jar
               org.openadaptor.adaptor.util.Encoder password
```

(all on one command line)

where:

— You supply the full path for the `bia-event.jar` file

— You supply your password in plain text

— The encoded version of the password is output to the command window display. You then paste this result into your adaptor configuration file.

4. The OVSD database time zone

   The `DB_TIME_ZONE` property needs to be set to the time zone that you chose when you (re)generated the OVSD Database Views. (See Creating the OVSD Views on page 56.)

   Refer to Valid Time Zones on page 118 to see the list of possible values that you can set for the `DB_TIME_ZONE` property.

5. Adaptor Remote Control Ports

   These values just need to be numbers of ports available on your system.

6. Adaptor Polling Period

   Specified in milliseconds. Defaults to 300000 (5 minutes.)

7. Adaptor Commit SQL command

   This property lets you set the behavior for the adaptors once they have processed records from the `ovbpi_item_change_hist` table.

If you do not set all properties in `SD5.0_Adaptor_Config.properties` correctly for your installation, you may not realize that there is a problem until you actually have everything up and running; see Run Time Adaptor Errors on page 113 for an example of errors that might occur.

➤ It is essential to take your time setting the `SD5.0_Adaptor_Config.properties` file and get all the values correct for your installation.

Generate the Adaptors and Triggers

Once you have set up the `SD5.0_Adaptor_Config.properties` file, you are ready to generate the adaptor configuration files and the SQL trigger scripts.

On the machine where you intend to run the adaptors, you run the script `sd5.0_createadaptors(.bat/.sh)` from the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors` directory, as follows:

- Open a Console window and change directory to the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors` directory.

  For Unix, you need to be logged on as the **root** user.

- Within this console window, set the following environment variables:
  - `OVBPI_ROOT`

    Although the variable is called `OVBPI_ROOT`, it needs to point to the mini-environment that you installed when you unzipped the `ovbpi-sd-adaptors.zip` file; that is, the `Work-dir` directory.

    Do not point this at your OVBPI installation, if you have one on the machine, as this can cause the `sd_createadaptors` script to fail.
  - `OA_ROOT`

    Set this to the installation directory for openadaptor on your machine.

    The default is: *`OVBPI_ROOT`*`\nonOV\openadaptor\1_6_5`
  - `JAVA_HOME`

    Set this to the installation directory for Java on your machine.

    The default is: `C:\java`
- Then run the script: `sd5.0_createadaptors(.bat/.sh)`

  The script reads the template adaptor configuration files, and SQL trigger files (found under the `SD5.0\template` subdirectory), and substitutes the property values as specified in your `SD5.0_Adaptor_Config.properties` file.

The `sd5.0_createadaptors` script creates a number of files in the `*Work-dir*\examples\bia\ServiceDeskAdaptors` directory, including:

- The adaptor configuration files:

  ```
  SD5.0_Poll_ServiceCall.props
  SD5.0_Poll_Incident.props
  SD5.0_Poll_Change.props
  SD5.0_Poll_Problem.props
  SD5.0_Poll_WorkOrder.props
  ```

  ...and the adaptor log4j logging configuration files:

  ```
  ServiceCallAdaptor_log4j.props
  IncidentAdaptor_log4j.props
  ChangeAdaptor_log4j.props
  ProblemAdaptor_log4j.props
  WorkOrderAdaptor_log4j.props
  ```

- The database trigger files:

  ```
  CreateServiceDeskTriggers.sql
  DropServiceDeskTriggers.sql
  ```

- The Windows-Service/Daemon configuration files:

  ```
  OVBPISDServiceCallAdaptor.cfg
  OVBPISDIncidentAdaptor.cfg
  OVBPISDChangeAdaptor.cfg
  OVBPISDProblemAdaptor.cfg
  OVBPISDWorkOrderAdaptor.cfg
  ```

  ...and the following additional files on a Unix system:

  ```
  OVBPISDServiceCallAdaptorWrapper.sh
  OVBPISDIncidentAdaptorWrapper.sh
  OVBPISDChangeAdaptorWrapper.sh
  OVBPISDProblemAdaptorWrapper.sh
  OVBPISDWorkOrderAdaptorWrapper.sh
  ovbpisd.sh
  ovbpisd.cfgsh
  ```

The `sd5.0_createadaptors` script also installs the five resulting adaptors as windows services, if running on a PC, or Unix daemons, if running on a Unix machine; see Restartable Services/Daemons on page 120 for more information about how to enable these services to be restartable.

Regenerating the Adaptors and Triggers

If you need to make alterations to any of the properties in your `SD5.0_Adaptor_Config.properties` file, simply re-run the `sd5.0_createadaptors` script, and the changes are reflected throughout all the regenerated files.

Install the Triggers

The SQL triggers need to be installed on the OVSD server's database.

After running the `sd5.0_createadaptors` script, you have two new SQL files in the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors` directory:

- `CreateServiceDeskTriggers.sql`

  The script to create the necessary triggers for the OVSD Process Insight integration.

- `DropServiceDeskTriggers.sql`

  A script for removing the OVSD Process Insight triggers.

To install the triggers you need to logon to the OVSD database as an administrator, and execute the following script:

```
CreateServiceDeskTriggers.sql
```

The `CreateServiceDeskTriggers.sql` script can be run from within your favorite SQL utility.

For example: (for MSSQL)

```
isql -U sduser -P sd -i CreateServiceDeskTriggers.sql
```

where:

- The user name `sduser` is the Service Desk administrative user.
- The password in this example is `sd`.

Verify OVSD with the Triggers

Having installed the OVSD Process Insight triggers into the OVSD database, it is important to verify that the triggers work and that the SD Client is still able to create and/or modify items such as Service Calls, Incidents, Changes, Problems and Work Orders.

Enter Some Dummy Data

In a development installation you could simply create a new item of each type - Service Call, Incident, Change, Problem and Work Order.

If the triggers are installed correctly there should be no database errors when creating any of these new items.

After creating each new item you should see new data records written to the `ovbpi_item_change_hist` table. You also see one or more records written to this database table for each new item you create.

If you are able to create new items within your SD Client and you see data being written to the `ovbpi_item_change_hist` table, then the triggers are installed correctly and your OVSD installation is functioning correctly.

If you are working on a production OVSD system you probably do not wish to create “dummy” items. That’s fine. As Calls/Incidents/etc. come in, your triggers are actioned and this verifies that everything is working.

Removing Process Insight Triggers

If you encounter problems and you are unable to create items from within your SD Client, you need to drop all the OVSD Process Insight triggers whilst you try and work out what is wrong with them.

To help you drop the triggers you can use the script: `DropServiceDeskTriggers.sql`.

Start the Adaptors

Now that the triggers are in place and working, you are able to start up the adaptors. The adaptors are all configured to run as either Windows-Services, or Daemons, depending on the operating system.

To start the adaptors:

1. Open a console window
2. Change directory to `Work-dir\examples\bia\ServiceDeskAdaptors`
3. Run: `sd_startadaptors(.bat/.sh)`

   This starts all adaptors.

Each adaptor is configured to log trace output to files within the `Work-dir\data\log` directory. For example, the Service Call adaptor outputs trace information to the log file called `OVBPISDServiceCallAdaptor.log`.

➤ Note that the adaptors append to their respective trace file. So if you stop and restart an adaptor, the trace output is **appended** to the end of the existing log file.

Each adaptor is configured to auto-start when the machine reboots. For more details about configuring the behavior of these adaptors see Restartable Services/Daemons on page 120.

Stop All Adaptors

The script `sd_stopadaptors(.bat/.sh)` is used to stop all the adaptors.

The Custom Dashboard

With the triggers installed and the adaptors running, you can run the standard OVBPI Business Process Dashboard and you are able to see your five ITIL flows.

Indeed, if you have entered any new Service Calls (as suggested in the verification step Verify OVSD with the Triggers on page 66) then you should be able to see them listed in the OVBPI Business Process Dashboard.

The OVBPI Business Process Dashboard helps you to test that things are working and that changes in OVSD are indeed making it over to OVBPI. However, there is a custom dashboard provided as part of the OVSD Process Insight module. The OVSD Process Insight custom dashboard presents a tailored view of the five ITIL flows.

What the Custom Dashboard Displays?

The adaptors monitor the following changes:

- Status changes.
- Deadline changes.
- Assigned Person changes.
- Priority changes.

As these types of changes occur in OVSD, you should be able to see them reflected in the OVBPI ITIL flows within your OVSD Process Insight custom dashboard...**however**...a flow instance does not occur until an item has changed its state! That is, an OVBPI flow instance is not created for an item until a state change has occurred. Once an item **has changed state,** a flow instance is created and OVBPI then continues to monitor all changes for this item.

So, if you are testing out that changes in the SD Client do indeed cause changes to appear in the OVSD Process Insight custom dashboard, don't forget that until an item has changed its state (for example, changed from `In Progress` to `Pending` - for a service call), it will not appear in the custom dashboard. It will not appear in either the OVBPI Business Process dashboard or the OVSD Process Insight custom dashboard.

Run the Custom Dashboard

The OVSD Process Insight custom dashboard is already installed under the `webapps` directory and is ready for use...

To run the OVSD Process Insight custom dashboard, point a Web browser at the URL:

```
http://hostname:44080/ovbpi_sd_dashboard2-10
```

where:

- *`hostname`* is the host name of your OVBPI machine
- *`44080`* is the default port for the Servlet Engine

For example:

```
http://localhost:44080/ovbpi_sd_dashboard2-10
```

This works if OVBPI is installed on the same machine as your Web browser.

Logging On

When OVBPI is first installed, no logon is required to use the OVSD Process Insight custom dashboard.

The OVBPI Administration Console allows the administrator to specify whether you require people to give a username and password when running the OVSD Process Insight custom dashboard. You then have a choice of authorization mechanisms. See the *OVBPI System Administration* guide for more details.

Set the SD Version Flag

The OVSD Process Insight custom dashboard needs to know whether it is working against a set of flows that are monitoring data from OVSD 4.5 or OVSD 5.x.

There is a property setting, called `SDVersion`, within the OVSD Process Insight custom dashboard's configuration file, `SD_DashboardConfig.properties`. This property needs to be set to `5.0` as follows:

```
SDVersion=5.0
```

At installation time, the `SDVersion` property defaults to the value `4.5`.

Refer to The Configuration Properties File on page 74 for details of how to locate and edit the OVSD Process Insight custom dashboard's configuration file.

OVBPI Engine Instance Cleaner

The main page of the OVSD Process Insight custom dashboard allows you to select the tab labelled: `Closed Instances Over Last Week`. However, this `Closed Instances Over Last Week` tab requires that your OVBPI Business Impact Engine instance cleaner is configured to leave at least one week's (seven days) worth of completed flow instances in the OVBPI database.

For example, if your OVBPI Business Impact Engine instance cleaner is cleaning out completed flow instances after only 3 days, then clicking on the `Closed Instances Over Last Week` tab is only able to present the last 3 days worth of data.

You should set the OVBPI Business Impact Engine instance cleaner to clean out completed instances after `7` days or more.

Summary of Main Steps

Here is a checklist summarizing the main steps, outlined above, to set up the OVBPI Process Insight module:

On the OVBPI Machine

- Locate the OVBPI Process Insight files.
- Deploy the ITIL flows. (Using the correct version of the flows.)
- Use `sd_exportadaptors.bat` to export the adaptor environment/files (creating the `ovbpi-sd-adaptors.zip` file).

On the OVSD Machine

- Enable auditing for the five objects.
- Generate the OVSD Database Reporting Views.

On the Machine running the Adaptors

- Unzip the `ovbpi-sd-adaptors.zip` file.
- Install openadaptor.
- Set up all installation specific properties by editing `SD5.0_Adaptor_Config.properties`
- Create the adaptors by running the script `sd5.0_createadaptors`

On the OVSD Machine

- Install the triggers into the OVSD database.

On the Machine running the Adaptors

- Run the adaptors by using the script `sd_startadaptors`.

On the OVBPI Machine

- Edit the OVSD Process Insight custom dashboard’s property file and set `SDVersion=5.0`
- Use the OVSD Process Insight custom dashboard to view your ITIL flows.

5 Basic Customization

The main page of the OVSD Process Insight custom dashboard (the `Service Scorecard` page) has three main sections:

- Operational Summary
- Delivery Summary
- Service Summary

These three sections do allow for some user customization, such as:

- Changing the time interval period (minutes, hours, or days)
- Changing the column headings
- Changing the intervals
- Changing the green/yellow/red ranges for the slider graphics
- Changing the list of services to display in the service summary graphs

It is all done by editing a simple configuration file.

This chapter explains how to make these customizations.

The Configuration Properties File

To modify the OVSD Process Insight custom dashboard settings, you can directly edit the configuration file:

```
OVBPI-install-dir\webapps\ovbpi_sd_dashboard2-10\WEB-INF\
 classes\SD_DashboardConfig.properties
```

...however... any modifications you make are temporary. If someone uses the OVBPI Administration Console to make any changes to OVBPI configuration this removes the changes you have made.

So you have two options available:

1. Make the changes to the `SD_DashboardConfig.properties` file and be aware that they may be lost the next time someone uses the OVBPI Administration Console to reconfigure things
2. Make the changes to the base template files, within the directory `newconfig\DataDir\conf\bia`, and then use the OVBPI Administration Console to apply the changes.

   The base templates files are named:

   — `SD_DashboardConfig.mssql.properties`

   — `SD_DashboardConfig.oracle.properties`

   If your OVBPI installation is using the MSSQL database then you edit the `SD_DashboardConfig.mssql.properties` file. If your OVBPI installation is using the Oracle database then you edit the `SD_DashboardConfig.oracle.properties` file.

Time Units

You can alter the unit of time used to calculate the deadline lists.

The properties are:

- `OperationSummaryTableUnits`
- `DeliverySummaryTableUnits`

These set the unit of time for the appropriate summary section of the page.

Values can be:

```
days
hours
minutes
```

Example

```
OperationSummaryTableUnits=hours
```

This means that the values given for the operational measurement intervals are read as hour values.

Measurement Intervals

You can alter both the size of each measurement interval and the number of intervals displayed.

The properties are:

- `OperationSummaryTableInterval`
- `DeliverySummaryTableInterval`

These set the measurement intervals for the appropriate summary section of the page.

Example 1

```
OperationSummaryTableInterval=-10,-5,-1,0,1,5,10
```

This is the default setting.

The operational section of the page shows calls and incidents divided into these 7 divisions of time. The time units are those set by the `OperationSummaryTableUnits` property. The intervals listed above are relative to the time the page is displayed.

If `OperationSummaryTableUnits` is set to days, then these 7 divisions list calls and incidents with deadlines as follows:

```
deadline >= 10 days from now
deadline >= 5 days from now but < 10 days
deadline >= 1 day from now but < 5  days
deadline <  1 day from now, today, or < 1 day in the past
deadline >= 1 day in the past but < 5 days
deadline >= 5 days in the past but < 10 days
deadline >= 10 days in the past
```

Example 2

```
OperationSummaryTableInterval=0,1,5,10
```

The operational section only displays calls and incidents that are due today or overdue.

If `OperationSummaryTableUnits` is set to days, then these 4 divisions list calls and incidents with deadlines as follows:

```
deadline = today, or < 1 day in the past
deadline >= 1 day in the past but < 5 days
deadline >= 5 days in the past but < 10 days
deadline >= 10 days in the past
```

Example 3

```
OperationSummaryTableInterval=-50,0,20
```

The operational section of the page shows calls and incidents divided into these 3 divisions of time.

If `OperationSummaryTableUnits` is set to days, then these 3 divisions list calls and incidents with deadlines as follows:

```
deadline >= 50 days from now
deadline <  50 days from now, today, or < 20 day in the past
deadline >= 20 days in the past
```

Column Headings

You can alter the column headings displayed for each corresponding measurement interval.

The properties are:

- `OperationalColumnHeaders`
- `DeliveryColumnHeaders`

These set the column headings for the appropriate summary section of the page.

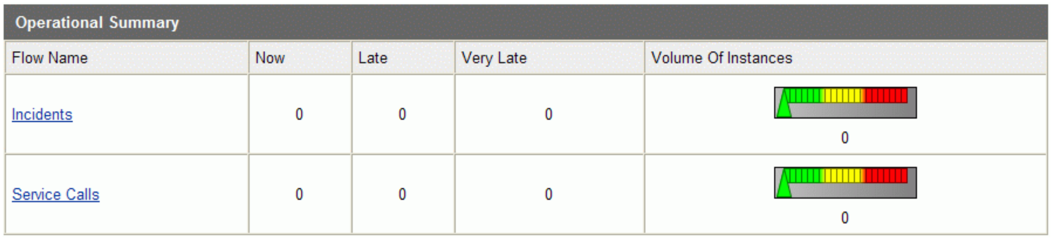
It is up to you to make sure that you column headings match the measurement intervals.

Example

```
OperationSummaryTableInterval=0,1,10
OperationalColumnHeaders=Now,Late,Very Late,Volume Of Instances
```

Would produce an operation section that looks as follows:

Figure 12 Custom Column Headings



Operational Summary

Flow Name	Now	Late	Very Late	Volume Of Instances
Incidents	0	0	0	0
Service Calls	0	0	0	0

Slider Ranges

For each flow, the main page shows a slider graphic. This slider displays the total number of instances that fall within the time intervals you have specified.

For each slider you can specify the min/mid/high/max ranges. That is, you can specify the size of the green/yellow/red areas.

The properties are:

- `OperationalSliderRangesAll`
  `OperationalSliderRangesOperator`
- `DeliverySliderRangesAll`
  `DeliverySliderRangesOperator`

You enter two sets of ranges. The order of the ranges signals which flow they are for within the appropriate summary section. The flows are listed in alphabetical order on the Web page so you need to specify the ranges for the flows assuming that order of display.

For each summary slider there are two ranges:

- `...RangesAll`

  Specifies the ranges to be used when the page is displaying for all operators.

- `...RangesOperator`

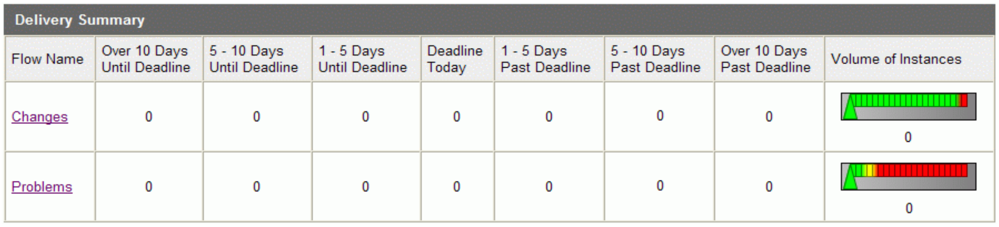
  Specifies the ranges to be used when the page is displaying for an individual operator.

Example

```
DeliverySliderRangesAll=0,1000,1000,1050|0,100,200,900
```

Sets the sliders to look as follows:

Figure 13 Custom Slider Ranges



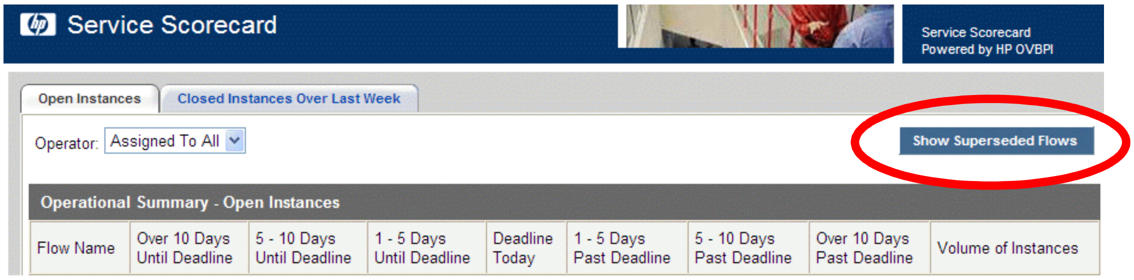
Delivery Summary

Flow Name	Over 10 Days Until Deadline	5 - 10 Days Until Deadline	1 - 5 Days Until Deadline	Deadline Today	1 - 5 Days Past Deadline	5 - 10 Days Past Deadline	Over 10 Days Past Deadline	Volume of Instances
Changes	0	0	0	0	0	0	0	0
Problems	0	0	0	0	0	0	0	0

Superseded Flows

The main page of the custom dashboard shows a button that allows you to view information for superseded flows. See Figure 14.

Figure 14 Showing Superseded Flows


Service Scorecard
Service Scorecard Powered by HP OVBPI
Open Instances
Closed Instances Over Last Week
Operator: Assigned To All
Show Superseded Flows
Operational Summary - Open Instances

Flow Name	Over 10 Days Until Deadline	5 - 10 Days Until Deadline	1 - 5 Days Until Deadline	Deadline Today	1 - 5 Days Past Deadline	5 - 10 Days Past Deadline	Over 10 Days Past Deadline	Volume of Instances

You can remove this button from the dashboard by setting the `EnableSupersededFlowButton` property as follows:

```
    EnableSupersededFlowButton = false
```

SDVersion

The OVSD Process Insight custom dashboard needs to know whether it is working against a set of flows that are monitoring data from OVSD 4.5 or OVSD 5.x.

The property setting `SDVersion` needs to be set to either `4.5` or `5.0`.

At installation time, the `SDVersion` property defaults to the value `4.5`.

For example:

> The following line tells the OVSD Process Insight custom dashboard that it is reporting against OVSD 5.x data:
>
> ```
> SDVersion=5.0
> ```

Service List

The main scorecard page of the OVSD Process Insight custom dashboard displays two graphs in the Services Summary section. These graphs list the number of Service Calls and Incidents against a particular service, and the average time to complete these calls and incidents.

You can customize which OVSD services are displayed in these graphs.

There are two sets of properties that determine the services listed, depending on the value of the `SDVersion` property.

If `SDVersion=4.5`, the properties used are:

- `ActiveTotalServiceNames`
- `AverageTimeToCompleteServiceNames`

If `SDVersion=5.0`, the properties used are:

- `SD5_ActiveTotalServiceNames`
- `SD5_AverageTimeToCompleteServiceNames`

You provide a comma-separated list of service names. You can chose any service names you require, and the OVSD Process Insight custom dashboard lists the number and average times of calls and incidents against those services.

6 Advanced Customization

The ITIL flows provided with the OVSD Process Insight module are provided as working examples. It is expected that individual OVSD installations may implement slightly different flows and/or flow states.

This chapter details how to extend the flow definitions to accommodate your individual installation.

Flow Level Changes

The most obvious change that may be required is a change to one or more of the flows. Flow level changes do not require any changes to the OVBPI adaptors or SQL triggers.

Each flow represents the various states that an item can be in during its life cycle, and these are based on the settings of a default OVSD installation. For example, an OVSD 4.5 Service Call goes through the states:

```
Registered
In progress
Waiting
Completed
Informed
Closed
```

and within the flow, there is a node for each of these states.

The flow progression rules tend to be quite straightforward. In the case of the `Service Call` flow, the progression rules tend to be of the form:

```
For NodeA:
Start Condition:  statusValue.after()  == "NodeA"
Stop  Condition:  statusValue.before() == "NodeA"
```

Representing Additional Nodes/States

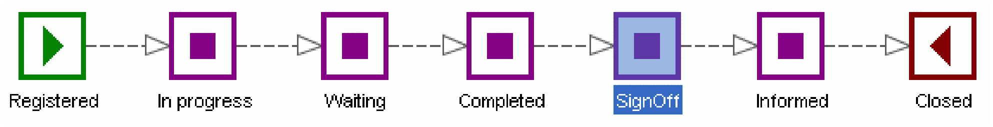
Suppose the Service Call states in your OVSD installation include the additional state `SignOff`.

The standard triggers and adaptors that come with the OVSD Process Insight module are able to handle this (and any) additional state, and the state information is sent into the OVBPI Business Impact Engine. All you need to do is edit the flow definition and add in a new node to represent this additional state.

For example:

You might edit the standard OVSD 4.5 `Service Call` flow to be as shown in Figure 15:

Figure 15 Service Call - Additional Node



where you have added the `SignOff` node after the `Completed` node, because that is where it makes sense for your installation.

You then need to give it progression rules that simply say when the state changes to `SignOff` - you are in the `SignOff` node, and when the state changes from `SignOff` - you are no longer in the `SignOff` node.

For example:

```
Start transition:
   Property: this.data.statusValue
   From:
   To:       "SignOff"

Complete transition:
   Property: this.data.statusValue
   From:     "SignOff"
   To:
```

This assumes that the text string `SignOff` is the new additional state value that is returned from your OVSD installation.

Renaming Nodes/States

The name of the nodes is totally up to you. You can name a node whatever name you like.

As for the states coming into the flow, the actual states values are used within the progression rules. So if your OVSD installation has the same number of states but the values (names) of these states are different, then you need to open up the flow and edit the progression rules to use the correct state values.

For example:

The default states for an OVSD 4.5 Service Call are:

```
Registered
In progress
Waiting
Completed
Informed
Closed
```

Suppose on your OVSD installation, the states were named:

```
Registered
Work In Progress
Waiting
Completed
Informed
Closed
```

You need to modify the `Service Call` OVBPI flow and change any progression rules that uses the state value `In progress`, to use the state value `Work In Progress` instead.

You do not have to change the name of the node...but you can if you want to.

Removing Nodes

If there are states in the default flows that you do not use, then you can remove the corresponding nodes from the flows. You should then check all progressions rules to make sure that nothing else is using that state value associated with the node you have just removed.

Event/Data Level Changes

There may be situations where there is additional OVSD data that you would like to monitor within the provided flows. To add additional data to the flows requires significant effort and customization.

There are two types of changes that you might consider:

1. Monitoring when something new changes its value

   The current triggers pick up on changes to the item's `Status`, `Priority`, `Deadline` and `Assigned Operator`. Suppose you also wanted to be told when (for example) the `Assigned Workgroup` changes for any items. This requires you modifying the existing database triggers to add an additional test condition. You then need to modify the adaptors and flow definition(s) to cope with this additional data.

2. Bringing in additional data values on existing item changes

   Suppose you want to bring in additional data values when (for example) any item changes its `Status`. This does not require any changes to the database triggers. This just requires the adaptors and the flow definition(s) to be modified.

Of course, once this additional data is in the OVBPI system, you need to think about whether you want to make any changes to the OVSD Process Insight custom dashboard. Maybe the data is just additional data with each flow instance. In this case, the standard OVSD Process Insight custom dashboard is able to display that data without any problem. However, maybe the data you are bringing in is something that you would like to highlight within the OVSD Process Insight custom dashboard. In this case, you would need to modify the OVSD Process Insight custom dashboard accordingly.

So, in general, the steps are:

1. Modify the triggers

   This step is required if you wish the triggers to pick up on changes to the values within an additional data field.

2. Modify OVSD (Version 5.x only)

   If you are wanting the triggers to pick up on changes to a new OVSD data field then you need to enable OVSD auditing for this field, and then regenerate the OVSD reporting views.

3. Modify the adaptors

   This step is always required. You must edit the adaptor configuration files so that they pick up the new data and send this into OVBPI. This data requires a change to your event definitions - either a new event or extension to an existing event definition.

4. Modify the flow definition (and redeploy)

   For new data to come into OVBPI you need to modify the event definition and the associated data definition (including the event subscriptions).

   You then need to redeploy the flow.

5. Modify the OVSD Process Insight custom dashboard

   The existing OVSD Process Insight custom dashboard shows the additional data by default. However, you might like to modify the OVSD Process Insight custom dashboard to highlight this additional data in some way. For example, you might wish to offer the user the ability to filter their screens based on the new data.

OVSD 4.5 Example: Filtering On Workgroups

Let's consider an example that extends the OVSD Process Insight module, where the integration is with OVSD 4.5.

The main screen in the custom dashboard for OVSD Process Insight allows you to filter the items to be shown according to the assigned operator. Suppose that you would prefer to have this filtering based, not on operator, but by assigned workgroup.

How do you go about extending the OVSD Process Insight module to provide for this?

The basic steps are as follows:

1. Extend the SQL in the template database triggers.
2. Extend the template adaptor files.
3. Set up any new tokens required in the `SD_Adaptor_Config.properties` file.
4. Recreate the adaptor configuration files and trigger files.
5. Remodel the flows:
   - Define the new data property in the associated data definition.
   - Define the new events
   - Define the new event subscriptions
   - Redeploy the flows
6. Install the modified database triggers.
7. Install and run the new adaptors.
8. Customize the OVSD Process Insight custom dashboard to allow users to filter by workgroup.

Let's go through and discuss each of these showing code segments and screen shots...

Extend the SQL in the Template Database Triggers

Because you want to receive an event whenever there is a change to the workgroup assignment for an item, you need to extend the SQL triggers to pick up on workgroup changes.

For each trigger, the extension is similar. The only difference is the name of the string you specify in the `ItemType` column.

For example, in the Service Call trigger (`servicecall_status_change`) you simply need to add another `if` statement that tests when the search code value is `To workgroup`. This is the value that is contained in the OVSD history record when this trigger is executed and the change is a workgroup assignment change. Of course, for your OVSD installation this might be different. You would obviously need to check this value for your installation.

The code segment looks like this:

```
if @SearchCode = 'To workgroup'
BEGIN
  INSERT INTO OVBPI_ITEM_CHANGE_HIST
  (Object_id, EventTime, EventStatus, ItemType)
  VALUES (@Object_id, @CurrentTime, 'New', 'SC_Toworkgroup')
END
```

where the string `SC_Toworkgroup` is written in the `ItemType` column. The `ItemType` forms the last part of the OVBPI event name when the adaptor sends this event. By default, all the adaptors prefix the `ItemType` value with the string `ServiceDesk`. So the code segment above is effectively saying that the event to be generated is going to be `ServiceDesk/SC_Toworkgroup`.

You extend the other four triggers with similar SQL code segments each specifying different `ItemType` values.

In this worked example, the values chosen are:

`SC_Toworkgroup`	for the Service Calls flow
`I_Toworkgroup`	for the Incidents flow
`C_Toworkgroup`	for the Changes flow
`P_Toworkgroup`	for the Problems flow
`WO_Toworkgroup`	for the Work Orders flow

You should apply these SQL code extensions to the template trigger files. The idea is to make your updates to the template files and use the `sd_createadaptors` script to generate the final set of files when everything is ready.

Extend the Template Adaptor Files

With the trigger writing additional records into the `ovbpi_item_change_hist` table, you need to extend the adaptor to handle these records.

You update the template adaptor configuration file and use `sd_createadaptors` to generate the final adaptor files when everything is ready. You choose the template file appropriate for the OVSD database your adaptor is interrogating. The adaptor template files are located under the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors\templates` directory.

All adaptor configuration files have a similar format. They consist of sections for each type of change they are monitoring. For example, in the Service Call adaptor there are sections for the components:

```
ServiceCallAdaptor.Component3.Name = StatusChange
ServiceCallAdaptor.Component4.Name = DeadlineChange
ServiceCallAdaptor.Component5.Name = PersonChange
ServiceCallAdaptor.Component6.Name = PriorityChange
```

Within each of these sections are all the configuration details for that component. So, to add a new component to monitor a new record type (`ItemType`) you essentially need to copy one of these component blocks and then modify this to handle the new record type.

The steps are basically the same for each adaptor. Let's consider how this is done for the Service Call adaptor:

1. Go to the end of the template adaptor configuration file.
2. Make a copy of the previous component's configuration section.
3. Increment the component number by 1.
4. Give this new component a unique name, and change this name throughout the whole component.
5. Change the primary key Select (`NextPrimaryKeySQL`) to look for the new `ItemType` value `SC_Toworkgroup`.
6. Work out which OVSD view you can use to get the Workgroup information and then use this in the Select statement (`SelectSQL`).

   You need to create tokens for any OVSD names and then define these within the `SD_Adaptor_Config.properties` file.
7. Save this adaptor configuration template file.

The configuration segment for the Service Call adaptor looks as follows:

```
ServiceCallAdaptor.Component7.Name            = WorkgroupChange
ServiceCallAdaptor.WorkgroupChange.LinkTo1    = ServiceCallAdornment

ServiceCallAdaptor.WorkgroupChange.ClassName  = org...jdbc.PollingSQLSource
ServiceCallAdaptor.WorkgroupChange.JdbcDriver = {{DB_DRIVER}}
ServiceCallAdaptor.WorkgroupChange.JdbcUrl    = {{DB_JDBC_URL}}
ServiceCallAdaptor.WorkgroupChange.Database   = {{DB_NAME}}
ServiceCallAdaptor.WorkgroupChange.UserName   = {{DB_USER_NAME}}
ServiceCallAdaptor.WorkgroupChange.Password   = {{DB_PASSWORD}}
ServiceCallAdaptor.WorkgroupChange.PasswordEncoding  = true
ServiceCallAdaptor.WorkgroupChange.ExitOnError       = false
ServiceCallAdaptor.WorkgroupChange.PollPeriod       = {{ADAPTOR_POLL_PERIOD}}
ServiceCallAdaptor.WorkgroupChange.QuoteMultipleKeys = true
ServiceCallAdaptor.WorkgroupChange.BatchSize         = 100
ServiceCallAdaptor.WorkgroupChange.UsingInClauses    = true
ServiceCallAdaptor.WorkgroupChange.PrimaryKeyRegExp  = PK

#
# get list of next primary keys to process
#
# Select the events of the new type "SC_Toworkgroup"
# ---------------------------------===============
ServiceCallAdaptor.WorkgroupChange.NextPrimaryKeySQL = SELECT buf.Object_id \
FROM OVBPI_ITEM_CHANGE_HIST buf \
WHERE \
buf.EventStatus = 'New' AND \
buf.ItemType = 'SC_Toworkgroup' \
ORDER BY buf.EventTime

#
# Process primary keys
#
ServiceCallAdaptor.WorkgroupChange.SelectSQL = SELECT \
buf.EventTime GeneratedDate, \
buf.ItemType EventName, \
sch.{{SCH_SERVICE_CALL_ID}} service_call_id, \
sch.{{SCH_CREATED_BY_DISPLAY_NAME}} changedBy, \
sch.{{SCH_DATE_CREATED}} changeTime, \
sc.{{SC_DEADLINE}} deadline, \
wkg.{{WORKGROUP_NAME}} workgroup \
FROM \
OVBPI_ITEM_CHANGE_HIST buf, \
{{HISTORY_LINE_SERVICE_CALL_VIEW}} sch, \
{{WORKGROUP_VIEW}} wkg, \
{{SERVICE_CALL_VIEW}} sc \
WHERE sch.{{SCH_OBJECT_ID}} IN ('PK') \
AND wkg.{{WORKGROUP_OBJECT_ID}} = sch.{{SCH_VALUE_TO}} \
AND sch.{{SCH_OBJECT_ID}} = buf.Object_id \
AND sc.{{SC_OBJECT_ID}} = sch.{{SCH_SERVICE_CALL_OBJECT_ID}} \
ORDER BY buf.EventTime

ServiceCallAdaptor.WorkgroupChange.CommitSQL  = {{COMMIT_SQL}}
```

where:

- This section was a copy of component6.
- The component is then renamed to component**7**.
- The name is changed to **WorkgroupChange**.
- The `NextPrimaryKeySQL` is picking up the new `SC_Toworkgroup` entries.
- The `SelectSQL` has been altered to pull out the workgroup details.

The difficult part of putting together the adaptor configuration is the `SelectSQL`. This requires you to investigate the OVSD database and determine where the workgroup information is located. In this case, it is located in the single view `v_workgroup`. You tokenize this view name as it may vary depending on the way the OVSD database reporting views were set up.

Set Up New Tokens in the SD_Adaptor_Config File

Having worked out the name of the view your adaptor needs to access, and the column names it needs, you tokenize these names within the adaptor configuration template file, and set up these tokens in the `SD_Adaptor_Config.propertes` file.

```
WORKGROUP_VIEW       = v_workgroup
WORKGROUP_OBJECT_ID  = object_id
WORKGROUP_NAME       = name
```

Recreate the Adaptor Configuration Files and Trigger Files

You run `sd_createadaptors` to process your template files and reproduce a new trigger file and adaptor configuration files.

It is always worth while checking the resultant adaptor configuration files to ensure that there are no {{ }} items remaining. If there are then this means that you have either not defined them in your `SD_Adaptor_Config.properties` file, or mis-spelt them within your adaptor configuration files.

Remodel the flows

You need to remodel your flows to accept the new events.

1. Define the new data property in the associated data definition.

   The adaptors are sending in new data item(s). In this example, they are sending in a new data item called `workgroup` - as a String[80].

2. Define the new events

   The adaptors are sending in the following new events:

`ServiceDesk/SC_Toworkgroup`	for the Service Calls flow
`ServiceDesk/I_Toworkgroup`	for the Incidents flow
`ServiceDesk/C_Toworkgroup`	for the Changes flow
`ServiceDesk/P_Toworkgroup`	for the Problems flow
`ServiceDesk/WO_Toworkgroup`	for the Work Orders flow

   Each event contains the data:

   ```
   changedBy        (String[80])
   changeTime       (Date)
   deadline         (Date)
   workgroup        (String[80])
   ```

   ...and an appropriate ID field.

3. Define the new event subscriptions

   You must set up new subscriptions for each of the associated data definitions. They need to subscribe to the appropriate event, pulling out all the data fields.

4. Redeploy the flows

   The flows need to be redeployed.

Install the Modified Database Triggers

The extended SQL triggers need to be installed.

For Oracle, this is simply a case of re-installing the triggers (using the `CreateServiceDeskTriggers.sql` script) - this is because Oracle has the concept of “create or replace”.

For an MSSQL database you need to first drop the triggers (using the `DropServiceDeskTriggers.sql` script) before then re-installing them (using the `CreateServiceDeskTriggers.sql` script).

Install and Run the New Adaptors

With the triggers in place, and the updated flows deployed, you are all set.

Start up the new adaptors and then try creating a new item within the SD Client. Assign this new item to a workgroup. You should see the adaptor sending the information across to OVBPI.

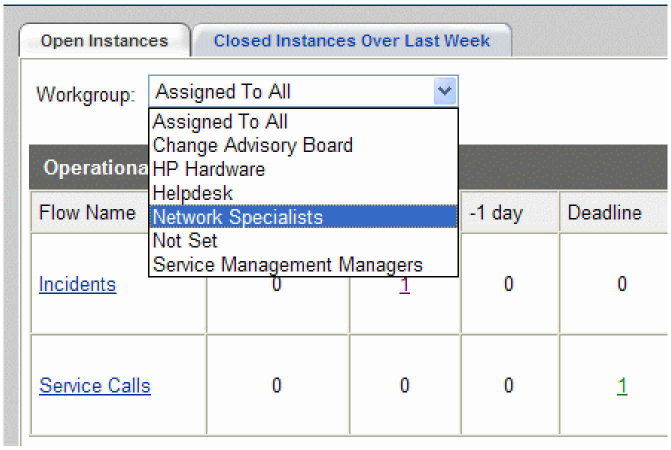
You can use the existing OVSD Process Insight custom dashboard to see if the new workgroup information is getting across to OVBPI. By drilling into a particular instance you should be able to see all the associated data - including the workgroup.

➤ Remember that the OVBPI flows only track an item once it has received a state change. That is, if you modify an existing item and alter the workgroup, you do **not** see this item reflected in OVBPI. You must first change the state of the item (for example, change it from `Registered` to `In progress`). Then all workgroup assignment changes for this item is tracked.

Customize the Dashboard to Allow Users to Filter by Workgroup.

Once you have workgroup information coming across to OVBPI, you can customize the OVSD Process Insight custom dashboard further such that users can filter the items on the main screen not by operator....but by workgroup (or both)!

For example, you could customize the main screen as follows:



Workgroup: Assigned To All

Flow Name			-1 day	Deadline
Incidents	0	1	0	0
Service Calls	0	0	0	1

where:

- Instead of selecting by assigned operator, the user can select by assigned workgroup.
- When the user has selected a workgroup, the main page then only shows items currently assigned to that workgroup.

To customize the OVSD Process Insight custom dashboard requires that you are comfortable reading and modifying JSP and Java code.

OVSD 5.x Example: Filtering On Workgroups

Let’s consider an example that extends the OVSD Process Insight module, where the integration is with OVSD 5.x.

The main screen in the custom dashboard for OVSD Process Insight allows you to filter the items to be shown according to the assigned operator. Suppose that you would prefer to have this filtering based, not on operator, but by assigned workgroup.

How do you go about extending the OVSD Process Insight module to provide for this?

The basic steps are as follows:

1. Extend the OVSD Auditing to include Workgroup information
2. Regenerate the OVSD DB Reporting Views
3. Extend the SQL in the template database triggers.
4. Extend the template adaptor files.
5. Set up any new tokens required in the `SD5.0_Adaptor_Config.properties` file.
6. Recreate the adaptor configuration files and trigger files.
7. Remodel the flows:
   - Define the new data property in the associated data definition.
   - Define the new events
   - Define the new event subscriptions
   - Redeploy the flows
8. Install the modified database triggers.
9. Install and run the new adaptors.
10. Customize the OVSD Process Insight custom dashboard to allow users to filter by workgroup.

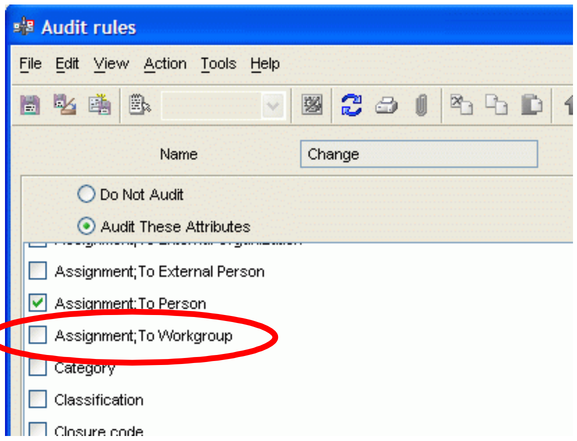
Let’s go through and discuss each of these showing code segments and screen shots...

Extend the OVSD Auditing to include Workgroup Information

Because you want to receive an event whenever there is a change to the workgroup assignment for an item, you need to tell OVSD to audit this data attribute for each of the five objects: `Service Call`, `Change`, `Problem`, `Work Order`, and `Incident`.

Refer to Enable OVSD DB Auditing on page 52 for details on how to do this. For each of the five objects, you need to select (tick) the attribute `Assignment; To Workgroup`, as shown in Figure 16:

Figure 16 Auditing Workgroup Attribute



Name: Change

Do Not Audit [✓ Audit These Attributes]

- [ ] Assignment;To External Person
- [x] Assignment;To Person
- [ ] Assignment;To Workgroup
- [ ] Category
- [ ] Classification
- [ ] Closure code

Regenerate the OVSD DB Reporting Views

Once you have enabled auditing on the `Assignment; To Workgroup` attribute for each of the five objects, you need to regenerate the OVSD Database Reporting Views.

Refer to OVSD Database Reporting Views on page 56 for details of how to do this.

Extend the SQL in the Template Database Triggers

You need to extend the SQL triggers to pick up on workgroup changes.

For each trigger, the extension is similar. The only difference is the name of the string you specify in the `ItemType` column.

For example, in the Service Call trigger (`servicecall_status_change`) you simply need to add another `if` statement that tests when the search code value is `To Workgroup`. This is the value that is contained in the OVSD history record when this trigger is executed and the change is a workgroup assignment change. Of course, for your OVSD installation this might be different. You would obviously need to check this value for your installation.

The code segment looks like this:

```
if @SearchCode = 'To Workgroup'
BEGIN
  INSERT INTO ovbpi_item_change_hist
  (Object_id, EventTime, EventStatus, ItemType)
  VALUES (@Object_id, @CurrentTime, 'New', 'SC_Toworkgroup')
END
```

where the string `SC_Toworkgroup` is written in the `ItemType` column. The `ItemType` forms the last part of the OVBPI event name when the adaptor sends this event. By default, all the adaptors prefix the `ItemType` value with the string `ServiceDesk`. So the code segment above is effectively saying that the event to be generated is going to be `ServiceDesk/SC_Toworkgroup`.

You extend the other four triggers with similar SQL code segments each specifying different `ItemType` values.

In this worked example, the values chosen are:

`SC_Toworkgroup`	for the Service Calls flow
`I_Toworkgroup`	for the Incidents flow
`C_Toworkgroup`	for the Changes flow
`P_Toworkgroup`	for the Problems flow
`WO_Toworkgroup`	for the Work Orders flow

You should apply these SQL code extensions to the template trigger files. The idea is to make your updates to the template files and use the `sd5.0_createadaptors` script to generate the final set of files when everything is ready.

Extend the Template Adaptor Files

With the trigger writing additional records into the `ovbpi_item_change_hist` table, you need to extend the adaptor to handle these records.

You update the template adaptor configuration file and use `sd5.0_createadaptors` to generate the final adaptor files when everything is ready. You choose the template file appropriate for the OVSD database your adaptor is interrogating. The adaptor template files are located under the *`Work-dir`*`\examples\bia\ServiceDeskAdaptors\SD5.0\templates` directory.

All adaptor configuration files have a similar format. They consist of sections for each type of change they are monitoring. For example, in the Service Call adaptor there are sections for the components:

```
ServiceCallAdaptor.Component3.Name = StatusChange
ServiceCallAdaptor.Component4.Name = DeadlineChange
ServiceCallAdaptor.Component5.Name = PersonChange
ServiceCallAdaptor.Component6.Name = PriorityChange
```

Within each of these sections are all the configuration details for that component. So, to add a new component to monitor a new record type (`ItemType`) you essentially need to copy one of these component blocks and then modify this to handle the new record type.

The steps are basically the same for each adaptor. Let’s consider how this is done for the Service Call adaptor:

1. Go to the end of the template adaptor configuration file.
2. Make a copy of the previous component’s configuration section.
3. Increment the component number by 1.
4. Give this new component a unique name, and change this name throughout the whole component.
5. Change the primary key Select (`NextPrimaryKeySQL`) to look for the new `ItemType` value `SC_Toworkgroup`.
6. Work out which OVSD view you can use to get the Workgroup information and then use this in the Select statement (`SelectSQL`).

   You need to create tokens for any OVSD names and then define these within the `SD5.0_Adaptor_Config.properties` file.
7. Save this adaptor configuration template file.

The configuration segment for the Service Call adaptor looks as follows:

```
ServiceCallAdaptor.Component7.Name              = WorkgroupChange
ServiceCallAdaptor.WorkgroupChange.LinkTo1      = ServiceCallAdornment

ServiceCallAdaptor.WorkgroupChange.ClassName  = org...jdbc.PollingSQLSource
ServiceCallAdaptor.WorkgroupChange.JdbcDriver = {{DB_DRIVER}}
ServiceCallAdaptor.WorkgroupChange.JdbcUrl    = {{DB_JDBC_URL}}
ServiceCallAdaptor.WorkgroupChange.Database   = {{DB_NAME}}
ServiceCallAdaptor.WorkgroupChange.UserName   = {{DB_USER_NAME}}
ServiceCallAdaptor.WorkgroupChange.Password   = {{DB_PASSWORD}}
ServiceCallAdaptor.WorkgroupChange.PasswordEncoding  = true
ServiceCallAdaptor.WorkgroupChange.ExitOnError       = false
ServiceCallAdaptor.WorkgroupChange.PollPeriod       = {{ADAPTOR_POLL_PERIOD}}
ServiceCallAdaptor.WorkgroupChange.QuoteMultipleKeys = true
ServiceCallAdaptor.WorkgroupChange.BatchSize         = 100
ServiceCallAdaptor.WorkgroupChange.UsingInClauses    = true
ServiceCallAdaptor.WorkgroupChange.PrimaryKeyRegExp  = PK

#
# Get list of next primary keys to process
#
# Select the events of the new type "SC_Toworkgroup"
# ---------------------------------------================
ServiceCallAdaptor.WorkgroupChange.NextPrimaryKeySQL = SELECT buf.Object_id \
FROM ovbpi_item_change_hist buf \
WHERE \
buf.EventStatus = 'New' AND \
buf.ItemType = 'SC_Toworkgroup' \
ORDER BY buf.EventTime

#
# Process primary keys
#
ServiceCallAdaptor.WorkgroupChange.SelectSQL = SELECT \
buf.EventTime GeneratedDate, \
buf.ItemType EventName, \
sch.{{SCH_SERVICE_CALL_ID}} service_call_id, \
sch.{{SCH_CREATED_BY_DISPLAY_NAME}} changedBy, \
sch.{{SCH_DATE_CREATED}} changeTime, \
sc.{{SC_DEADLINE}} deadline, \
wkg.{{WORKGROUP_NAME}} workgroup \
FROM \
ovbpi_item_change_hist buf, \
{{HISTORY_LINE_SERVICE_CALL_VIEW}} sch, \
{{WORKGROUP_VIEW}} wkg, \
{{SERVICE_CALL_VIEW}} sc \
WHERE sch.{{SCH_OBJECT_ID}} IN ('PK') \
AND wkg.{{WORKGROUP_OBJECT_ID}} = sch.{{SCH_VALUE_TO}} \
AND sch.{{SCH_OBJECT_ID}} = buf.Object_id \
AND sc.{{SC_OBJECT_ID}} = sch.{{SCH_SERVICE_CALL_OBJECT_ID}} \
ORDER BY buf.EventTime

ServiceCallAdaptor.WorkgroupChange.CommitSQL  = {{COMMIT_SQL}}
```

where:

- This section was a copy of component6.
- The component is then renamed to component**7**.
- The name is changed to **WorkgroupChange**.
- The `NextPrimaryKeySQL` is picking up the new `SC_Toworkgroup` entries.
- The `SelectSQL` has been altered to pull out the workgroup details.

The difficult part of putting together the adaptor configuration is the `SelectSQL`. This requires you to investigate the OVSD database and determine where the workgroup information is located. In this case, it is located in the single view `v_workgroup`. You tokenize this view name as it may vary depending on the way the OVSD database reporting views were set up.

Note that when you come to extend the adaptor properties file for the **Incident** adaptor, there are additional adaptor components numbered 7 and 8. So you need to copy component 6 (the component that handles a priority change) and number this copy as component 9.

Set Up New Tokens in the SD5.0_Adaptor_Config File

Having worked out the name of the view your adaptor needs to access, and the column names it needs, you tokenize these names within the adaptor configuration template file, and set up these tokens in the `SD5.0_Adaptor_Config.propertes` file.

```
WORKGROUP_VIEW        = v_workgroup
WORKGROUP_OBJECT_ID   = object_id
WORKGROUP_NAME        = name
```

Recreate the Adaptor Configuration Files and Trigger Files

You run `sd5.0_createadaptors` to process your template files and reproduce a new trigger file and adaptor configuration files.

It is always worth while checking the resultant adaptor configuration files to ensure that there are no {{ }} items remaining. If there are then this means that you have either not defined them in your `SD5.0_Adaptor_Config.properties` file, or mis-spelt them within your adaptor configuration files.

Remodel the flows

You need to remodel your flows to accept the new events.

1. Define the new data property in the associated data definition.

   The adaptors are sending in new data item(s). In this example, they are sending in a new data item called `workgroup` - as a String[80].

2. Define the new events

   The adaptors are sending in the following new events:

`ServiceDesk/SC_Toworkgroup`	for the Service Calls flow
`ServiceDesk/I_Toworkgroup`	for the Incidents flow
`ServiceDesk/C_Toworkgroup`	for the Changes flow
`ServiceDesk/P_Toworkgroup`	for the Problems flow
`ServiceDesk/WO_Toworkgroup`	for the Work Orders flow

   Each event contains the data:

   ```
   changedBy       (String[80])
   changeTime      (Date)
   deadline        (Date)
   workgroup       (String[80])
   ```

   ...and an appropriate ID field.

3. Define the new event subscriptions

   You must set up new subscriptions for each of the associated data definitions. They need to subscribe to the appropriate event, pulling out all the data fields.

4. Redeploy the flows

   The flows need to be redeployed.

Install the Modified Database Triggers

The extended SQL triggers need to be installed.

For Oracle, this is simply a case of re-installing the triggers (using the `CreateServiceDeskTriggers.sql` script) - this is because Oracle has the concept of “create or replace”.

For an MSSQL database you need to first drop the triggers (using the `DropServiceDeskTriggers.sql` script) before then re-installing them (using the `CreateServiceDeskTriggers.sql` script).

Install and Run the New Adaptors

With the triggers in place, and the flows deployed, you are all set.

Start up the new adaptors and then try creating a new item within the SD Client. Assign this new item to a workgroup. You should see the adaptor sending the information across to OVBPI.

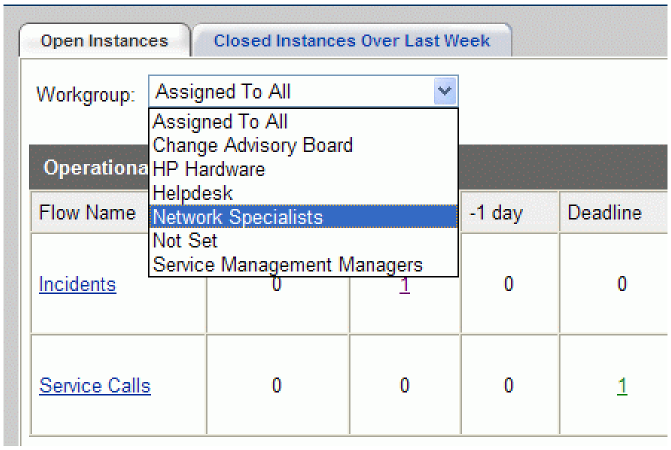
You can use the existing OVSD Process Insight custom dashboard to see if the new workgroup information is getting across to OVBPI. By drilling into a particular instance you should be able to see all the associated data - including the workgroup.

➤ Remember that the OVBPI flows only track an item once it has received a state change. That is, if you modify an existing item and alter the workgroup, you do **not** see this item reflected in OVBPI. You must first change the state of the item (for example, change it from `In Progress` to `Pending`). Then all workgroup assignment changes for this item are tracked.

Customize the Dashboard to Allow Users to Filter by Workgroup.

Once you have workgroup information coming across to OVBPI, you can customize the OVSD Process Insight custom dashboard further such that users can filter the items on the main screen not by operator....but by workgroup (or both)!

For example, you could customize the main screen as follows:



Flow Name			-1 day	Deadline
Incidents	0	1	0	0
Service Calls	0	0	0	1

where:

- Instead of selecting by assigned operator, the user can select by assigned workgroup.
- When the user has selected a workgroup, the main page then only shows items currently assigned to that workgroup.

To customize the OVSD Process Insight custom dashboard requires that you are comfortable reading and modifying JSP and Java code.

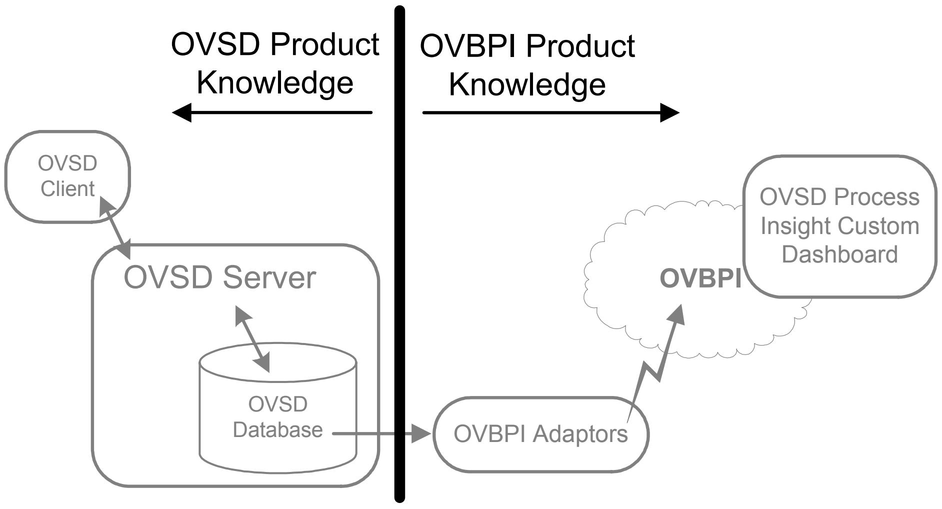
7 Further Topics

This chapter includes sections on troubleshooting the OVSD Process Insight module, and other miscellaneous features.

Troubleshooting

If you think about the division of product knowledge required to support the OVSD Process Insight module it can be essentially divided as follows:

Figure 17 Product Knowledge Division



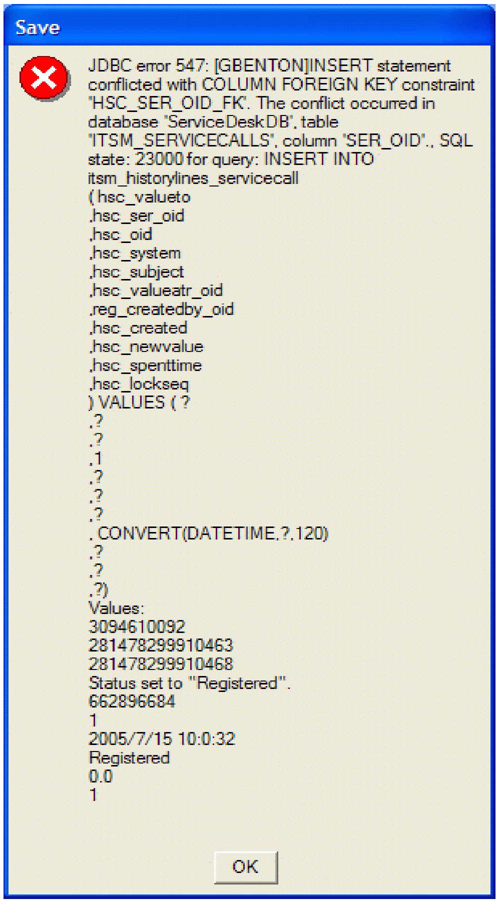
Certainly if someone has a problem using the OVSD Process Insight custom dashboard then it is an OVBPI issue. However the situation may not be that straightforward if someone using the SD Client complains that they cannot add a new Service Call.

Let's now consider some example problems and their solutions.

JDBC Error in the SD Client

Because the OVSD Process Insight module adds triggers to the history line tables, if there is a problem with a trigger on one of the history lines tables this may prevent the entering or updating of the corresponding item (Service Call, Incident, Change, Problem or Work Order).

For example, if there is a problem with the logic in the trigger placed on the Service Call history table, when the SD Client tries to add a new Service Call, or update an existing Service Call, they receive a JDBC Error. The message may look something like this:



```
JDBC error 547: [GBENTON]INSERT statement
conflicted with COLUMN FOREIGN KEY constraint
'HSC_SER_OID_FK'. The conflict occurred in
database 'ServiceDeskDB', table
'ITSM_SERVICECALLS', column 'SER_OID'., SQL
state: 23000 for query: INSERT INTO
itsm_historylines_servicecall
( hsc_valueto
,hsc_ser_oid
,hsc_oid
,hsc_system
,hsc_subject
,hsc_valueatr_oid
,reg_createdby_oid
,hsc_created
,hsc_newvalue
,hsc_spenttime
,hsc_lockseq
) VALUES ( ?
,?
,?
,1
,?
,?
,?
, CONVERT(DATETIME,?,120)
,?
,?
,?)
Values:
3094610092
281478299910463
281478299910468
Status set to "Registered".
662896684
1
2005/7/15 10:0:32
Registered
0.0
1
```

Unfortunately, there is nothing obvious in this message to help you decide whether the problem is due to the OVBPI SQL trigger. The message is basically saying that “Something went wrong when doing the SQL Insert”.

Do not simply assume that this error message means that there is a problem with the SQL trigger. There may be a valid OVSD problem totally unrelated to the trigger. But certainly the trigger can cause the SD user to be unable to add or update items (Service Calls, Incidents, etc.).

Install Testing

Clearly you can avoid any such JDBC problems by ensuring that the person who installs the SQL Triggers tests that an SD Client is able to add and update Service Calls, Incidents, etc.. once the triggers in place.

Access to the OVBPI Table

If your triggers are causing your JDBC problems, make sure you also check that the `ovbpi_item_change_hist` table is correctly created and that your triggers are able to access this table. The default installation sets this up correctly, however it is always worth checking if your triggers are having a problem.

Dropping the Triggers

This is **not** something that you really want to do on a production system as it causes OVBPI to miss activity within OVSD. However, if you are getting these JDBC problems when adding or updating items within OVSD, you could choose to drop all the OVSD Process Insight module triggers. This must only be considered when all other options have been exhausted, as it causes an interruption to the OVBPI monitoring.

See Removing Process Insight Triggers on page 42.

Obviously, once the problem has been resolved you would want to reinstate the corrected triggers so as to allow OVBPI monitoring to continue.

Exception at Adaptor Startup

If you run `sd_startadaptors` you might see the following error message in the adaptor log files:

```
[06/05/02 12:14:08.833] [ERROR]: Caught exception [Decode failure : missing
version], whilst password decoding.
```

This error message is output if you have not set the `DB_PASSWORD` property to a correctly encoded password value. You should also check that you have correctly set all the other OVBPI database properties in the adaptor configuration properties file. This adaptor configuration file is either:

- `SD_Adaptor_Config.properties` for OVSD 4.5
- `SD5.0_Adaptor_Config.properties` for OVSD 5.x

Run Time Adaptor Errors

A very important step in setting up the adaptors and triggers is the step where you set up the adaptor configuration file:

- `SD_Adaptor_Config.properties` for OVSD 4.5, or
- `SD5.0_Adaptor_Config.properties` for OVSD 5.x

If you do not get every name in this file correct for your installation, then your adaptors will have trouble...but they are most likely to have this trouble when they come to process their first data record. That is, the adaptors may well start up successfully, but it is not until they have each processed a record that you can be sure you have everything configured correctly.

Setting the adaptor configuration file is one of the most important steps in the whole process. So take your time when doing it. Make sure every name corresponds to the correct name in your OVSD database.

If you do have a mis-naming problem then your adaptors throw an exception **at run time**. Here is an example exception thrown by the Service Call adaptor:

```
[05/10/22 14:22:46.719] INFO: StatusChange executing query [SELECT
 buf.EventTime GeneratedDate, buf.ItemType EventName, sch.service_call_id
 service_call_id, sch.service_call_description description,
 sch.created_by_display_name changedBy, sch.created changeTime, rct.rct_name
 valueTo, sc.service_name serviceName, sc.service_id serviceId FROM
 OVBPI_ITEM_CHANGE_HIST buf, v_history_line_incorrectViewName sch,
 rep_codes_text rct, v_service_call sc WHERE sch.object_id IN
 ('281478296240215') AND rct.rct_rcd_oid = sch.value_to AND
 sch.object_id = buf.Object_id AND
 sc.object_id = sch.service_call_object_id ORDER BY buf.EventTime]
[05/10/22 14:22:46.719] WARN: SQL threw exception, com.inet.tds.SQLException:
 Msg 208, Level 16, State 1, Line 1, Sqlstate S0002[GBENTON]Invalid object
 name 'v_history_line_incorrectViewName'.
[05/10/22 14:22:46.719] FATAL: Unexpected exception, Source terminating,
 java.lang.NullPointerException
[05/10/22 14:22:46.719] INFO: ServiceCallAdaptor.Controller - Source
 StatusChange is exiting
[05/10/22 14:22:46.729] INFO: Adaptor terminating with error code 1
```

The main cause of the run-time error is that the name of one of the OVSD Database Reporting Views is incorrect for this installation.

To fix this problem:

- Fix the naming error in the appropriate adaptor configuration properties file.
- Re create the adaptors, using either:
  - `sd_createadaptors` for OVSD 4.5
  - `sd5.0_createadaptors` for OVSD 5.x
- Re start the adaptors.

See also Oracle Exception - Table or View Does Not Exist on page 115 for another possible run-time exception.

Oracle Exception - Table or View Does Not Exist

You may encounter an error when your openadaptor adaptors are running against an OVSD database that is installed on Oracle. This should only be an issue when running OVSD 4.5.

The problem is that one (or more) of your adaptors terminates when it tries to process its first data record. The exception thrown is an Oracle database exception:

```
Table or view does not exist.
```

This error can occur if you specified more than one database user name when you installed the OVSD database.

When you install OVSD 4.5, the database install wizard gives you a screen where you configure the name of a default database user. When you are installing on Oracle, this database wizard screen also gives you the option to specify a second database user name, who has access to the OVSD Repository tables. By default you just specify a single “default” user. However, you might have decided to specify this second database user to have access to the OVSD Repository tables. If you have configured this second database user, then the OVBPI openadaptor adaptors will fail when they come to process the first data record.

When you configure the OVBPI adaptor property file you specify a `DB_USER_NAME` property. This `DB_USER_NAME` specifies the database user that the OVBPI adaptor uses to connect to the OVSD database. If you have configured your OVSD database to have two users (a default user and an OVSD Repository user) then the OVBPI adaptor is unable to access all the tables and views that it needs, and hence it terminates with an exception.

If you have set up two OVSD database users then the “default” user has access to all the OVSD Reporting Views. Hence you should set the `DB_USER_NAME` property in your adaptor configuration property file to this “default” user. However, the OVBPI adaptor also needs access to the `rep_codes_text` table within the OVSD database, and this table is only viewable to the OVSD Repository user.

To get around this problem, you need to grant the OVSD “default” user access to the `rep_codes_text` table within the OVSD database. For example, the Oracle command to grant the required access might be as follows:

```
grant select on rep_codes_text to default-user-name
```

Oracle 10 Performance Issues

There can sometimes be an issue where the OVSD Process Insight adaptors run very slowly. That is, the adaptors work fine, however it takes them many seconds (or even minutes) to process each data record and send them to OVBPI.

The problem is caused by a database performance issue when the standard OVSD Process Insight adaptors run against an Oracle 10 database. The way the adaptors join the OVSD Reporting Views can be very slow with Oracle 10.

If you encounter this problem, and you are running OVSD 5.x, then there is a way to fix this. You need to use a different set of adaptor templates and then use the `sd5.0_createadaptors` script to create a new set of adaptors.

The steps are as follows:

1. Stop all OVSD Process Insight adaptors.
2. Locate the new set of templates in the directory:

`Work-dir``\examples\bia\ServiceDeskAdaptors\SD5.0\templates\Oracle10`

3. Copy these `SD5.0_Poll_*.template` files up into the directory:

`Work-dir``\examples\bia\ServiceDeskAdaptors\SD5.0\templates`

   In other words, copy these Oracle 10 template files up one level so that they overwrite the old template files.

4. Run the `sd5.0_createadaptors` script to create a new set of adaptor files based on these new templates.
5. Start up the adaptors.

Dates and Time Zones

When you first set up OVBPI to monitor your OVSD installation, you should check that the dates and times being sent by OVBPI are correctly shown within the OVSD Process Insight custom dashboard.

The most obvious date and time field to check is the `deadline` attribute. That is, choose an item within OVSD (for example, choose a Service Call) and take note of the `deadline` date and time. Also take note of the time zone that the OVSD Client is running in. Then locate this same item within the OVSD Process Insight custom dashboard and see what date, time and time zone is shown for the `deadline` attribute. The `deadline` times shown should equate to the same time. For example: 2/May/2006 16:02 ECT (European Central Time) is the same as 2/May/2006 15:02 BST (British Summer Time).

The date/time values are stored within the OVSD Database Views according to the time zone that you selected when you (re)generated the OVSD Reporting Views. Unfortunately, these date/time values are held in the OVSD database **without** the actual time zone information. That is, when your openadaptor reads the date/time value it does not know what time zone this value represents. This is why you set the property `DB_TIME_ZONE` within the adaptor configuration properties file.

The value you specify for the `DB_TIME_ZONE` property is then used within the adaptor configuration files to make sure that the adaptors read the data/time values in the correct time zone. You need to set the `DB_TIME_ZONE` property to match the time zone you used when you (re)generated the OVSD Reporting Views.

Refer to Valid Time Zones on page 118 for a list of valid time zones that you can specify for the `DB_TIME_ZONE` property.

Valid Time Zones

When setting the `DB_TIME_ZONE` property you need to specify a time zone string that is understood by openadaptor adaptors.

The actual time zone can be any string token that represents a valid time zone. For example:

```
UTC
GMT
Europe/London
America/Phoenix
```

You can not specify an offset, or relative, time zone. For example you can not specify any of the following:

```
GMT+1
GMT+02:00
UTC+1
```

If you do specify an offset, or relative, time zone, as in the above examples, the adaptor reads times in the base time zone and disregards the offset.

You can also have issues where certain time zone strings do not seem to work as you would hope. The most consistent style of time zone string is the long-format time zones such as:

```
Europe/London
America/Phoenix
```

If you need to find a time zone, use an internet search engine and there are many pages that list possible time zone strings.

Where Are My Flow Instances?

If you have made changes to an item in OVSD (using the SD Client) but nothing is appearing in the OVSD Process Insight custom dashboard then it could be because of a number of different reasons:

- Check that the adaptors are up and running
- Check that OVBPI is up and running and has not reported any errors
- Check that the SQL triggers are installed in the OVSD database

It may be that you know everything is up and working because when you make a change through the SD Client you can see the adaptor sending information to OVBPI...yet no flow instance gets created!

If everything looks alright and you are still unable to understand why a certain item does not appear in the OVSD Process Insight custom dashboard, it might be because the item has **not yet changed its state**. Until an item has changed its state, that item does not appear in OVBPI. See What the Custom Dashboard Displays? on page 44 for more details.

Restartable Services/Daemons

When you run the script to create your adaptors, the adaptors are all installed as windows services or Unix daemons - depending on your operating system.

Windows PC

On a windows PC, the adaptors are installed and registered as windows services with the following names:

```
OVBPISDChangeAdaptor
OVBPISDIncidentAdaptor
OVBPISDProblemAdaptor
OVBPISDServiceCallAdaptor
OVBPISDWorkOrderAdaptor
```

All these services are configured to start automatically when the system boots.

You can use the `Services` panel within the PC’s `Control Panel` to configure these services to auto-restart in the event of any failures.

Un-Registering Windows Services

If you wish to unregister all the OVSD Process Insight module windows services, you can run the script `sd_unregisteradaptors.bat`, which is located in the directory *`Work-dir`*`\examples\bia\ServiceDeskAdaptors`.

Unix Machine

On a Unix machine, the adaptors are wrapped as restartable daemons.

The scripts to start each adaptor are created in the `Work-dir/examples/bia/ServiceDeskAdaptors` directory. These scripts are named as follows:

```
OVBPISDServiceCallAdaptorWrapper.sh
OVBPISDIncidentAdaptorWrapper.sh
OVBPISDChangeAdaptorWrapper.sh
OVBPISDProblemAdaptorWrapper.sh
OVBPISDWorkOrderAdaptorWrapper.sh
```

The script `sd_startadaptors.sh` simply invokes each of the above listed scripts in turn, to start up all the adaptors.

Start/Stop at Reboot

By default, the adaptors are configured to auto-stop and auto-start when the system shuts down or reboots. This is achieved by creating the files:

```
/etc/rc.config.d/ovbpisd
/sbin/init.d/ovbpisd
/sbin/rc2.d/K010ovbpisd
/sbin/rc3.d/S999ovbpisd
```

Configuring Auto-Restart on Failure

By default, the adaptors are **not** configured to restart themselves if any one of them encounters an error and closes itself down.

You can configure each adaptor to be able to auto-restart itself if it encounters an error and subsequently closes down. To do so, you need to edit each of the adaptor wrapper configuration files:

```
OVBPISDServiceCallAdaptorWrapper.sh
OVBPISDIncidentAdaptorWrapper.sh
OVBPISDChangeAdaptorWrapper.sh
OVBPISDProblemAdaptorWrapper.sh
OVBPISDWorkOrderAdaptorWrapper.sh
```

You need to edit each of the above listed files and set the following configuration options:

```
RESTART_ON_FAILURE
RETRY_PERIOD
```

where:

- `RESTART_ON_FAILURE`

  Set this to `true` to enable the script to restart the adaptor in the event of a failure.

- `RETRY_PERIOD`

  Specify the amount of time (in seconds) to wait before attempting to restart the adaptor after a failure.

For example, the following configuration sets the script to restart after failure, with a delay of 60 seconds (1 minute) between retries.

```
RESTART_ON_FAILURE=true
RETRY_PERIOD=60
```

Un-Registering Daemons

If you do not want the adaptors started/stopped automatically at system boot/shutdown time, then you can run the script `sd_unregisteradaptors.sh` which is located in the directory *`Work-dir`*`/examples/bia/ServiceDeskAdaptors`. This script removes the `ovbpisd` files from the `/etc/rc.config.d` and `/sbin/init.d` directories.