HP Service Manager

For the supported Windows® and UNIX® operating systems

Software Version: 9.31

Upgrade Guide (from HP Service Manager 7.0x)

Document Release Date: October 2012

Software Release Date: October 2012

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Java is a registered trademark of Oracle and/or its affiliates.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle® is a registered US trademark of Oracle Corporation, Redwood City, California.

UNIX® is a registered trademark of The Open Group.

For a complete list of open source and third party acknowledgements, visit the HP Software Support Online web site and search for the product manual called *HP Service Manager Open Source and Third Party License Agreements*.

Documentation Updates

The title page of this document 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://h20230.www2.hp.com/selfsolve/manuals

This site requires that you register for an HP Passport and log on. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the **New users - please register** link on the HP Passport log on page.

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

Visit the HP Software Support Online web site at:

http://www.hp.com/go/hpsoftwaresupport

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

HP Software online 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 valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into 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 sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

Contents

Chapter 1

Upgrade overview

The HP Service Manager Upgrade Utility upgrades the HP Service Manager 7.0x applications to HP Service Manager 9.31 applications. If you are running an HP ServiceCenter application version prior to 6.2, you should contact HP Customer Support for information about the best upgrade strategy for your version. You can also obtain an Upgrade Guide for previous versions of ServiceCenter at:

http://h20230.www2.hp.com/selfsolve/manuals.

You will need an HP Passport to access this site. For additional information, contact HP Customer Support or your HP sales representative.

The purpose of this Upgrade Guide is to describe how to upgrade your HP Service Manager 7.0x applications to Service Manager 9.31 applications.

Before you begin an upgrade

Before you begin an upgrade, ensure that you:

- Read through the *Upgrade Guide* to familiarize yourself with the upgrade process and all of the upgrade requirements.
- Are an experienced HP Administrator who is familiar with Service Manager.

If you do not have the administrative experience necessary to manage the upgrade, you may need assistance from your local application developers and database administrators. You can also contact HP Service Manager Customer Support for help with troubleshooting upgrade errors. For additional information and support, contact your HP sales representative.

Server and client upgrade

Make sure that you have upgraded your server and the client to the latest version before you attempt to run an application upgrade. This allows the upgrade utility to call the new functions in the latest server and client and to take full advantage of all of the application features following an upgrade. If you have deployed Knowledge Management, you must also upgrade the Knowledge Management Search Engine to the new version.

To upgrade your applications by using this Upgrade Patch, you must first perform a server and client upgrade to 9.31 platform (server and client).

- To obtain the latest client, install the client from the installation CD-ROM and follow the instructions in the *HP Service Manager Installation Guide*.
- To obtain the latest server, install the HP Service Manager 9.31 server from the installation CD-ROM.

Application upgrade

You can upgrade your existing Service Manager applications to Service Manager 9.31 applications using the Upgrade Utility and resolving the differences between the two versions.

What are applications?

Applications are the Service Manager modules and their related configuration files. For example, Incident Management, Change Management, and Inventory Management are Service Manager applications.

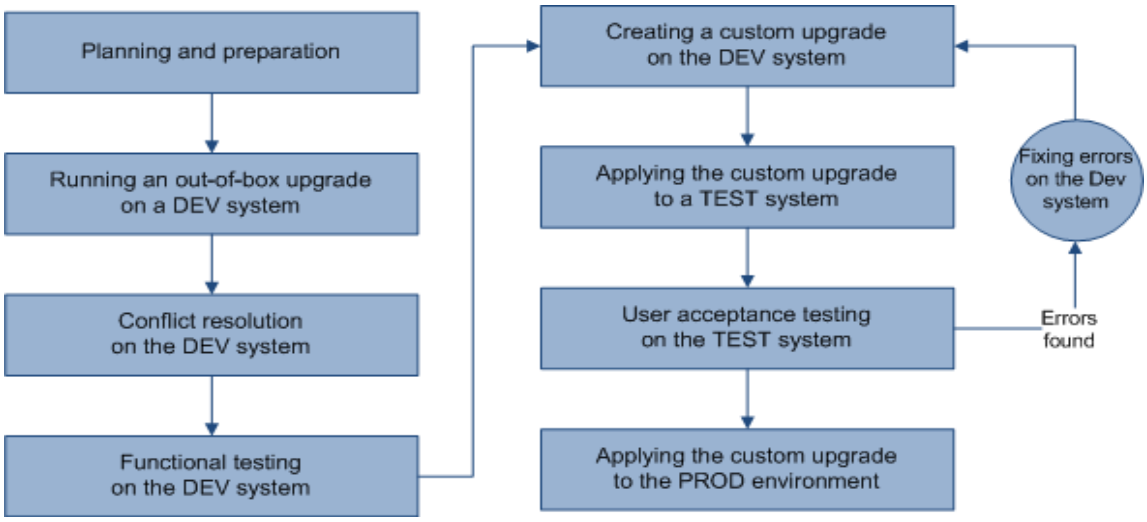
New features that require an application upgrade

Some new features provided by the release of Service Manager 9.31 require an application upgrade. These new features include the following:

- Process Designer
- Knowledge Management SOLR search engine
- Mobility
- Service Manager 9.30 Service Request Catalog (SRC)

Application upgrade lifecycle

The following flow chart illustrates the lifecycle of a typical upgrade of Service Manager applications.



How does customization affect the upgrade process?

The following explains how customization affects the upgrade process:

Conflicts

Object changes: The Upgrade Utility compares the objects in your database with their out-of-box versions and the corresponding objects provided from the upgrade package. The Upgrade Utility compares objects by their signatures. Each data record in Service Manager has a unique signature, which changes once that data record is updated.

- If an object in your database that has been tailored does not match the corresponding object that is provided by the upgrade package, the following behaviors apply:
  - If this is your first upgrade, the Upgrade Utility tries to merge the object from the upgrade package with your tailored object. If the merge is successful, the Upgrade Utility marks the

new merged object as "Auto Merged." If the merge fails, Upgrade Utility marks the object as "Renamed." In both cases, the Upgrade Utility prefixes the object from the upgrade package with "NEW931," and then copies the object in your datebase and prefixes that object as "PRE<*version_number*>". For example, an object from application version 7.11.000 would be prefixed with "PRE7.11.000."

  - If this is a custom upgrade, the Upgrade Utility marks the object from the upgrade package as “Forced,” and then copies the object in your database, prefixing it with “OLD<*version_number*>." For example, an object from application version 7.11.000, would be prefixed with "OLD711."

- If an object in your database has been tailored, but the out-of-box version matches exactly the corresponding object from the upgrade package, the following behaviors apply:
  - If this is your first upgrade, the Upgrade Utility keeps your local version, even if your version has been tailored and marks your local version as “Kept Customer”.
  - If this is a custom upgrade, the Upgrade Utility marks your local version object as “Already Current.”
- If an object in your database has not been tailored, but does not match the corresponding object provided from the upgrade package, the following behaviors apply:
  - If this is your first upgrade, the Upgrade Utility overwrites the object in your database with the object from the upgrade package, and marks the object as “Upgraded.”
  - If this is a custom upgrade, the Upgrade Utility marks the object from the upgrade package as “Forced,” and then copies the object from your database, prefixing it with “OLD<*version_number*>." For example, an object from application version 7.11.000, would be prefixed with "OLD711."
- If an object in your database matches exactly the object provided from the upgrade package, the Upgrade Utility marks the object as "Already Current" regardless of whether this is the first upgrade or a custom upgrade.
- If an object from the upgrade package does not have a corresponding object from your database, the Upgrade Utility adds the object to your database and marks the object as "Added" regardless of whether this the first upgrade or a custom upgrade.
- If an object from your database does not have a corresponding object from the upgrade package, the Upgrade Utility ignores it.

Schema changes: The Upgrade Utility merges new fields to existing schemas, without deleting any existing fields.

Field mapping changes: The Upgrade Utility applies field mapping changes automatically. For example, when a length change is required, the utility automatically expands the length mapping.

Key changes: The Upgrade Utility applies key changes automatically. It does not delete existing keys, except when Unique keys must be overwritten to support required changes.

Customization during upgrade

If any tailoring changes are made to your production system, for example, by applying an application "hot fix," after you have initiated the upgrade process, it is highly recommended that you apply those same changes to the development system that is being used for conflict resolution before you create the final custom upgrade package. Or, these tailoring changes may be lost after

the custom upgrade has been applied to production, and any conflict resolution that needs to be done in a production environment may slow down the production upgrade.

Upgrade Utility contents

The following table lists the files that are included in the Service Manager Upgrade Utility.

List of upgrade utility files

File	Contents
AppUpgVersion.txt	Contains Upgrade Utility version and build number information to help you identify which application upgrade version you have available. For example: A version of "SC62-9.31.*xxx* v9.31 *xxx* Upgrade Build *xxx*" indicates the following: • The Upgrade Utility upgrades ServiceCenter 6.2 and later releases to Service Manager 9.31. • The Upgrade Utility version number is *9.31.xxx*. • The Upgrade Utility build number for this version is *xxx*.
preupg.bin	Files that allow you to access the various features of the Upgrade Utility.
transfer.bin	Files that allow for the execution of the upgrade.
sqlupgrade.unl	Files that allow you to run SQL Compare, a feature of the Upgrade Utility.
upgrade.inf	Signature information for the upgrade objects.
upgrade.str	Database dictionaries to be upgraded.
upgrade.ver	Version stamp for this upgrade.
*.dta (in the Data folder)	The data files for each table that needs to be upgraded. For example, upgradeactivityactions.dta and upgradeactivitytype.dta.
upgrade.mak	Signature definitions for the upgrade objects.
upgdbdct.dta	Temporary dbdicts needed for the SQL Compare process.

Chapter 2

Planning an upgrade

Good planning allows your upgrade to run as smoothly and quickly as possible, and helps you to avoid retracing your steps. When preparing for your upgrade, you will need to consider how long each step will take and when users need to be logged off the system so that you can schedule each phase of your upgrade.

Topics in this section include:

Step 1: Identify the upgrade resources

Make sure that you have access to the following resources

- *Service Manager tools*: The utilities you will use most during the upgrade process include Database Manager and Forms Designer.
- *Documentation resources*: For client/server installation instructions, see the *HP Service Manager Installation Guide* that is shipped with the installation media. Additionally, you can obtain most Service Manager knowledge from the Service Manager 9.31 online Help.
- *HP Customer Support Web site*: The HP Customer Support Web site has operating system and compatibility information, product documentation, and release notes. If you do not have an account for this Web site, contact HP Customer Support at: *www.hp.com/go/hpsoftwaresupport*

Step 2: Meet the software requirements

Before you start your upgrade, make sure that you meet the following Service Manager system requirements:

- Your RDBMS version, operating system, and client/server environment must meet all criteria listed in the Compatibility Matrix for the target version. See the HP Customer Support web site to review the Compatibility Matrix.
- Your existing Service Manager application release level must be HP Service Manager 7.0x.
- The Service Manager server process (sm) must have read-write access to the database.

Backups

It is highly recommended, at a minimum, that you back up the database at the following strategic points in the upgrade lifecycle:

- After applying an upgrade
- After resolving conflicts

NFS-mounted partitions

Do not install either Service Manager or the Service Manager Upgrade Utility on an NFS-mounted remote partition. This can cause serious performance degradation. The performance of an NFS-mounted partition drops significantly if it reads data in many small pieces instead of one large chunk. Service Manager generates a lot of database read/write activity. An NFS-mounted partition is significantly slower than a local drive when running the Upgrade Utility process.

Step 3: Perform a system health check

A well-maintained production system is the easiest to upgrade. Before starting the upgrade process, perform all regular maintenance on your production system. If necessary, contact HP Customer Support for recommended actions. Suspend all customization activity on the production system.

Step 4: Create development and test environments

Plan to have at least two copies of your existing production environment:

- A development system that mirrors your current production environment. Use the development system to run the Upgrade Utility and build a custom upgrade.
- A test system that mirrors your current production environment. Apply the custom upgrade on the test system and verify it there.

Step 5: Develop an upgrade strategy

In standard Service Manager terminology:

- *Customization* refers to changes to RAD applications;
- *Tailoring* refers to changes made by using Service Manager tailoring tools, such as Forms Designer and Format Control;
- *Configuration* refers to local settings (for example, in your environment records and the system information record).

The upgrade process affects different parts of the Service Manager system. Besides upgrading the standard Service Manager applications, an upgrade affects the RDBMS where Service Manager is running and any customized files or RAD applications. For more information, see "How does customization affect the upgrade process?" (on page 10).

Tailored systems

A list of tailored files can help you resolve differences quickly between your existing files and new files. You can also use the SQL Compare utility to determine how files differ.

RDBMS-mapped systems

Because Service Manager tables (data files) must be mapped to an RDBMS, you must choose one of the following options before beginning the upgrade:

- Allow the Upgrade Utility to modify your RDBMS tables for you.
- Use SQL Compare to update the RDBMS databases before beginning the upgrade process.

The upgrade can affect certain mappings and tables. Contact your database administrator for assistance and to discuss the impact on the RDBMS.

Localized systems

You can upgrade a localized system with the Upgrade Utility. Before you begin to upgrade a localized system, ensure that you have the correct language pack available to which you will be upgrading. For more information and instructions on how to install the language pack, refer to the *HP Service Manager 9.30 Language Pack Installation Guide*. The Upgrade Utility detects the presence of a localized system and runs the upgrade just as it would for an English system. You will have to make any customization and tailoring changes, based on the requirements described in this document for each of your system configurations.

Customized RAD applications and ScriptLibrary records

A list of customized RAD applications and ScriptLibrary records and the extent of the customization is useful. If it is not available, the programmer who made the changes may be able to supply information. Or, you may need to run a comparison between the existing application or script and the new version.

Chapter 3

Preparing for an upgrade

Before you can develop or test an upgrade, you must create a mirror image of your current Service Manager production environment and prepare the system data for the upgrade process.

Topics in this section include:

Step 1: Meet database requirements

Before upgrading your system, you must review and perform the applicable RDBMS conversion tasks.

Note: For Oracle users, you must have a granted role that includes “connect” and “resource” along with a granted system privilege of “select any dictionary” as a minimum in order to avoid errors generated by Oracle.

Convert all tables and fields from lowercase to uppercase

Service Manager does not generate lowercase table names or field names. Therefore, if your database is case sensitive, you must convert all tables and fields from lowercase to uppercase before you can upgrade the server and client or applications.

Enlarge the page size for DB2 RDBMS table spaces

If you are running on a DB2 RDBMS, make sure that the buffer size for the following tables is at least 32K:

- inbox
- kmquery
- svcCatalog
- systemperform

To enlarge user space for the affected tables:

1. For each table that requires a larger user space, run the following SQL statements (replace the placeholders with appropriate values) and press Enter after each statement:

```
create table INBOXM1_TEMP like INBOXM1 in USER32K

commit

insert into INBOXM1_TEMP (select * from INBOXM1)

commit

drop table INBOXM1

commit

rename table INBOXM1_TEMP to INBOXM1

commit
```

2. Repeat step 1 for each table, making the applicable command substitutions. Note that the command sequence shown in Step 1 applies to inbox1.

Notes:

- USER32K represents the name of the larger-sized user space, one that has a buffer size larger than 4K.
- If the **kmquery** table is empty, you can skip the insert statement.

See your DB2 documentation for more information.

Remove indexes and constraints in the RDBMS

Note: This step is required only if you previously upgraded from a version prior to Service Manager 7.0.

The current or a previous upgrade process may fail to remove certain mapped indexes and constraints after removing a key from Service Manager applications. These indexes or constraints may prevent the upgrade process from adding records to those specific tables. To remove these indexes and constraints manually:

1. Open the dbdict for the **svcCatInterface** table, and remove the **service.request.id** unique key.
2. Log in to the RDBMS, and remove the mapped index for the **svcCatInterface** table.

   Example: For an Oracle RDBMS, run the following SQL statements to determine the key name and then remove the index if it exists:

```
select i.index_name, c.column_name from user_indexes i, user_ind_
columns c where i.table_name='SVCCATINTERFACEM1' and i.index_
name=c.index_name

drop index SVCCATINTERFACEM1_2
```

3. Log in to the RDBMS, and remove the mapped constraint and index for the **notification** table.

   Example: For an Oracle RDBMS, run the following SQL statements to remove the constraint and index (if they exist):

```
alter table notificationm1 drop constraint NOTIFICATIONM1_P

drop index NOTIFICATIONM1_P
```

Step 2: Purge existing upgrade files

If you have run an applications upgrade in the past, there may be some artifacts left over from upgrade processing that need to be removed.

To purge existing upgrade files:

1. Type `*aapm.upgrade.purge` in the Service Manager client command box. Press **Enter**.
2. Select **I'm done, and I want to remove the upgrade files completely**.
3. Click **OK** to proceed.

Step 3: Upgrade the server and client

Verify that the server and client has been upgraded to the latest version. For more information, see the Service Manager documentation and the "Server and client upgrade" (on page 9) section.

Step 4: Duplicate the production environment

To achieve the best results, develop and test the custom upgrade on a system that resembles your production environment as closely as possible.

To duplicate the production environment:

1. Identify a server to use for the development and test environments.
   - Unix: You can copy the files to a new location on your production machine.
   - Windows: You must create the development system on a different machine from your production system.
2. Ensure that adequate memory and disk space is available and accessible. Frequent backups are necessary.
3. Ensure that your development and test systems meet all upgrade requirements. For more information, see "Step 2: Meet the software requirements" (on page 13).
   - Upgrade your RDBMS to a version compatible with HP Service Manager 9.31. See the Service Manager 9.31 compatibility matrix.
   - Convert your RDBMS code page to Unicode. See your RDBMS vendor documentation.
4. Set up the environment of your development and test machines to resemble your production server as closely as possible. The operating system version and service pack level should match.
5. Copy your existing production system data onto your development system.
   HP recommends you use the native RDBMS backup utilities to back up your data. Refer to your RDBMS documentation for backup instructions.
6. Install a Service Manager 9.31 run time environment on the duplicated system. Do not load the Service Manager 9.31 demonstration data files.
7. Install a Service Manager 9.31 client on the duplicated system.

Step 5: Update Service Manager configuration files

The following tables list the changes that you need to make to the Service Manager configuration files before running the Upgrade Utility. Record all changes that you have made so that you can

revert them to the original status after the upgrade.

Stop the Service Manager server, apply the required changes to the configuration files, and then restart the Service Manager server.

sm.cfg

Parameter	Changes	Description
sm system.start	If this parameter exists, comment it out by changing it to: `#sm system.start`	Commenting this parameter out disables the background processes.
sm -sync	Add this parameter to the end of the file if it does not exist yet.	This parameter starts the sync process, which identifies and releases locks owned by inactive processes and shared memory that is not in use.
sm -httpPort	If there is more than one instance of the "sm -httpPort" parameter, keep only one instance.	Each "sm -httpPort" parameter starts a Service Manager server process that can handle a certain number of client sessions (see the Service Manager Help Server documentation for more information). Keeping one process alive will be enough for the upgrade process.
Other parameters	Comment out all other parameters except the ones mentioned in this table.	Commenting out those parameters disables all the other Service Manager processes that are not required during an upgrade.

sm.ini

Parameter	Changes	Description
ir_disable:1	Add this parameter to the end of the file if it does not exist.	This parameter disables all IR keys on your existing Service Manager system. This will make the upgrade process run faster.
sessiontimeout:1200	Add this parameter to the end of the file if it does not exist. If this parameter already exists, update it to an appropriate value.	This parameter defines the number of minutes that the server waits for a client heartbeat signal before the server assumes that the client session has timed out and closes the connection. A value of 1200 sets the timeout to 20 hours (1200 minutes), a period that should be enough for an

Parameter	Changes	Description
		upgrade phase to complete in a typical scenario.
JVMOption(#):-Xss6M	Required only for HP-UX systems: Add this parameter to the end of the file if it does not exist. **Note:** When adding the parameter, replace the hash symbol (#) with an option number that is not used in the sm.ini file. For example, if the sm.ini file already contains a **JVMOption(0)** and **JVMOption(1)**, add `JVMOption(2):-Xss6M` to the file.	This parameter increases the Java virtual machine stack size to 6MB.
shared_ memory:96000000	Replace the default **shared_ memory:32000000** with **shared_memory:96000000**.	This sets the shared memory size to 96MB. However, if you have a large database, you may need to allocate more shared memory to accommodate the upgrade processing.
jsgctrigger:67108864	Add this parameter to the end of the file if it does not exist.	This enables JavaScript garbage collection at a memory threshold of 64MB to avoid issues that may potentially occur.

Chapter 4

Running the Upgrade Utility

Now that you have a functional environment, you are ready to run the Upgrade Utility. Follow the steps in this chapter to run the out-of-box upgrade against the data in your development system and to run your custom upgrade against your test and production systems. You must perform these steps in a Service Manager Windows client, instead of a Web client.

Caution: If the upgrade fails while the Upgrade Utility is running, fix possible issues and rerun the Upgrade Utility, and you should be able to resume the upgrade from the failure point; if the upgrade process cannot be resumed, you must restore the database to the last backup point and fix possible issues before you can rerun the Upgrade Utility.

Topics in this section include:

Step 1: Load the application upgrade files

You must load the **preupg.bin** file and the **transfer.bin** file into Service Manager before you can use the Upgrade Utility.

Note: If you are performing a custom upgrade on a test or production system, use the **preupg.bin** file and the **transfer.bin** file included in your custom upgrade instead.

To load the application upgrade files:

1. On the Service Manager server, create a folder (referred to as the *Upgrade* folder later in this document).

   Notes:

   Make sure that the Service Manager server process (sm) has write and execute privileges for this folder.

   If you are connecting to the Service Manager server from a client that is installed on a remote client computer, make sure that the folder is created on the Service Manager server instead of the client computer.

2. Extract the Service Manager application Upgrade Utility files to the *Upgrade* folder.
3. Log in to the Service Manager Windows client as a system administrator.
4. Click **Window** > **Preferences** > **HP Service Manager**, and uncheck the **Client side load/unload** option.

   Important Note: Failure to disable this option will cause the upgrade process to fail.

5. Load the **preupg.bin** file using Service Manager's Database Manager.

6. Type **smupgrade** in the Service Manager client command box, and press **ENTER**. This starts the Upgrade Utility.
7. In the **UPGRADE UTILITY** section, click **load transfer**.
8. In the text box, type the fully qualified path to the folder that hosts **transfer.bin**, and then click **Next**.

   Note: When typing the path do not include the file name (**transfer.bin**) in the path.

   Example:

   Windows: `c:\temp\upgrade\`

   Unix: `/tmp/upgrade/`
9. Wait until the file is loaded and the system displays the message "Transfer files loaded."

   Note: The loading process may take a long time.
10. Log off Service Manager and log back on.

Step 2: Run the SQL compare utility

The SQL compare utility is an informational tool that you can use to compare DBDICT changes in the new release with the DBDICTS in your current release. It lists the fields and tables that the upgrade will change and gives you the opportunity to determine whether or not you want to accept the changes.

Note: If you are going to accept the new DBDICTS and the changes made to the DBDICTS in the upgrade, you do not need to run this utility.

The SQL compare utility compares your existing table and field information with the tables and fields of the Service Manager version you are upgrading to. It will report new fields that are to be merged into the existing tables. You can use the list of the fields produced by SQL Compare to determine whether any fields in your current system differ from those in the new version. This may be useful if you choose to update the RDBMS manually.

Running SQL Compare

The following SQL Compare files are included when you install the Service Manager Upgrade Utility:

- sqlupgrade.unl
- upgdbdct.dta

SQL Compare returns messages for dbdict mappings that contain new fields. You can update the dbdicts to contain the fields specified by the SQL Compare applications before you begin the application upgrade.

Note: Run SQL Compare on the development system.

To run the SQL Compare utility:

1. Load **sqlupgrade.unl** into your development system from Database Manager.
2. Type **smupgrade** in the Service Manager client command box, and press ENTER.
3. Click **Run SQL Compare Utility**. A dialog box opens.

4. Type the fully qualified path to upgdbdct.dta including the final back slash (\) or forward slash (/), depending on your operating system. For example, if you copied the files to a temporary directory, the path might be:

   Windows: `c:\temp\upgrade\`

   Unix: `/tmp/upgrade/`

   Do not include the file name (**upgdbdct.dta**) in this path.

5. Click the Load button.

   SQL Compare returns this message:

   Process Complete. Please check for any additional messages.

   The results of the SQL Compare process are stored in the **sqlupgrade** table. This table resets each time you run SQL Compare.

To view the SQL compare results:

1. Type **smupgrade** in the Service Manager client command box, and press ENTER.
2. Click **View SQL Compare Results**.
3. Click **Search**. The results are displayed in a record list.

Each database dictionary that requires changes appears as a separate record in the **sqlupgrade** file. This record also lists the new fields that you must add to the database dictionary, if you are updating your RDBMS mapped system manually.

The sqlupgrade record provides the following information for each field you must add, if you are updating your RDBMS mapped system manually.

List of sqlupgrade fields

Field	Description
Field Name	The exact field name to add to the associated database dictionary.
Type	The data type of the field.
Level	The level where this field resides.
Structure	The structure and array name that you should add to this field.
Alias of	If this is an alias field, it contains the name of the primary field that it is an alias of. Otherwise this field is blank.

Add new fields

For the new fields to perform correctly, they must exist in the database dictionary and the SQL database. If you are updating your RDBMS mapped system manually, you must add them to the database and update the existing Service Manager SQL mapping. When you update a table in sqlsystemtables, add fields only through the database dictionary. Modifying the SQL mapping damages the file structure of the table.

Determine the correct structure

In most cases, you should add the new field to the descriptor structure. However, sometimes the Structure field contains something other than the word descriptor. When this occurs, add the new field to the appropriate location.

Action to take with non-descriptor fields:

In this instance	Add the field here
The field resides in another structure	If the field is not an array field, you must add the field to the structure listed in the Structure field. For example, if the Structure field reads middle, add the field to the middle structure of the dbdict.
The field is an array	If the field is an array, the field name appears twice in the new field list. The first entry has a data type of array; the second is the data type of the array, such as character or logical. Use the first entry to determine the structure where you should add the array. The Structure field in the second entry reflects both the structure for the array (unless it uses the descriptor structure) and the name of the array itself.
The field is part of an array of structures	If the Structure field lists multiple fields exclusive of an array name, you must add the field to a structured array. To determine the placement in the structured array, follow the list of field names in the Structure/Array from left to right. The first name is the array name and the second is the structure name.

Note: When adding fields to an array of structures, add them in the same order as they appear in the sqlupgrade record.

Step 3: Run the Upgrade Utility

The running of the Upgrade Utility involves the following three primary phases:

1. In the first phase, the Upgrade Utility guides you through several questions and collects information needed for the upgrade.
2. The second phase is called the "dbdict update phase," where the utility updates dbdicts.
3. The third phase is called the "data update phase," where the utility updates application data.

To run the Service Manager Upgrade Utility:

1. Type **smupgrade** in the Service Manager client command box, and press **ENTER**. This opens the Upgrade Utility.
2. In the **UPGRADE UTILITY** section, click **SERVICE PACK**.
3. In the **Upgrade Processing** section, click **Apply an Upgrade**.
4. On the Welcome screen, verify that the **Applications version upgrading from** field displays

your current application version (**SM7**), and then click **Next** to continue.

Note: If this screen does not display the correct version, do not continue with the upgrade. Instead contact HP Software Customer Support.

5. **Message: Please select the language(s) in addition to English to be upgraded.**

   You see a list that displays all the supported non-English languages. By default, the languages that are installed on your Service Manager system are set to **true**. Select **false** for the languages that you do not want to upgrade, and click **Next**.

   Note: You can choose to update more languages later if you choose only part of the installed languages in this step. For more information, see "Updating languages at a later time" (on page 61).

6. **Message: Are you going to use this system to create a custom upgrade for another system?**

   We are preparing a custom upgrade on a development system, so leave the selection as **Yes**, and then click **Next**.

7. **Message: What is the fully qualified path to the HP Service Manager Upgrade patch files?**

   By default, the text box displays the fully qualified path to the *Upgrade* folder on the Service Manager server. Keep the path unless it does not point to the correct folder, and then click **Next**.

8. **Message: When HP Service Manager Upgrade doesn't recognize an object it should**

   We are preparing a custom upgrade on a development system, so select **Install HP's Version of the Object Alongside Your Own**, and then click **Next**.

   Note: The **Replace your version of the object with the HP Service Manager's version of the object** option is for applying a custom upgrade. Do not choose that option at this point unless you are sure to replace your own versions of objects and do not need to perform conflict resolution.

9. **Message: Do you want to enable automatic conflicts merge and which base version do you like to use?**

   For the Upgrade Utility to merge conflicting objects automatically, select the **Enable** check box, select a **Base Version** from the list, and then click **Next**.

   Note: The Base Version you select here will be considered as a clean version on which all changes are based. The earlier the version is, the less the chance your tailoring will be discarded. However, the earlier the version is, the less likely the utility will successfully merge objects. Typically, it is safe to keep the default value, which is set to the earliest version that your applications were previously upgraded from.

10. The upgrade is now ready to start. Click **Next**.
11. When you are asked whether you want to proceed, click **Yes**.
12. The Upgrade Utility displays the status when the upgrade is being processed.
13. When you receive an "UPGRADE IS COMPLETE" message, the Upgrade Utility has finished the data processing.

Caution: If new schedulers are displayed on the status window after the upgrade completes, do not stop the system! Let the schedulers finish the background data upgrade before stopping the system.

14. Exit the client and restart the server.
15. Open the **scversion** table in the Database Manager, and verify that the **Application Version** field is **9.31.0022**. If this field displays a value other than**9.31.0022**, check the log files to identify the issue that occurred.

Upgrade Utility logs and error messages

The Upgrade Utility creates a set of log files during the upgrade process. These files reside in the same directory as the upgrade files.

List of upgrade log files

Log file	Contents
detail.log	Specific information about the upgrade, such as which files are being signed at any time.
except.log	Information about any exceptions reported by the upgrade. The **except.log** file may contain messages about data type mismatches that failed to be resolved, or database dictionaries failed to be upgraded. See "Data type mismatches" (on page 30). If there are exceptions logged in this file, you will have to resolve them in the "Resolving exceptions and conflicts" phase.
upgrade.log	Information about where the upgrade is at any point. This file contains only the main steps of the upgrade.

Chapter 5

Resolving exceptions and conflicts

After you complete an application upgrade on your development system, you are ready to resolve the exceptions and conflicts that originate from tailoring on an upgraded object. Before resolving conflicts, Service Manager features may not function as expected. This chapter will guide you through a process called "conflict resolution."

Topics in this section include:

Upgrade results

While you are applying an out-of-box upgrade on the development system, the Upgrade Utility stores information regarding the upgrade result of each object. You can access this information in the Upgrade Utility through **View/Merge Upgrade Results**.

View the upgrade results

To view the upgrade results:

1. Type **smupgrade** in the Service Manager client command box, and press ENTER. This opens the Upgrade Utility.
2. In the UPGRADE UTILITY section, click **View/Merge Upgrade Results**.
3. In the **Result** drop-down list, select the type of results you want to search for.

   Example: Renamed

   When you select a result type from the drop-down list, the description of that result type appears under the drop-down list.
4. Click **Search**.
5. A list is returned that displays all the result records of the specified type.

   Note: Some types of results are only informational and do not require any follow-up action.

Manage the upgrade result data

To manage the upgrade result data more easily, you can either:

- Open the Upgrade Results list, click **More** or the More Actions icon, and click **Export to Excel** to manage the data in a Microsoft Excel document, or
- Open the Upgrade Results list, click **File** → **Print** → **List View** to print the list of records. (See the Help Server documentation for more information.)

Step 1: Resolve exceptions

Exceptions are logged if the Upgrade Utility cannot update an object. After running an upgrade, you can identify exceptions by viewing error messages in the except.log or sm.log file. These exceptions are reported in the Upgrade Results list as "Error."

Data type mismatches

If the data type of a field in your dbdict does not match the data type of the like-named field defined in the dbdict provided by the upgrade package, the Upgrade Utility cannot merge these dbdicts. For example, if an existing dbdict has a scalar field and the Upgrade Utility attempts to add a structure field with the same name, this discrepancy prevents the dbdict from being updated.

To fix this issue, you can change the data type and the SQL type of the field, and use Complex Update to migrate existing data on that field to the target data type. The following is an example that shows the process of fixing a typical data type mismatch.

Note: The error messages in the except.log file identify each data type by an index number:

Index number	Data type
1	number
2	character
3	date/time
4	logical
8	array
9	structure
11	expression

The following table shows a list of data type mismatches that may appear in the except.log file. The data type mismatches are listed for an Oracle database. If you are using an MSSQL or a DB2 database, the actual error message may vary slightly. For example, the following bullet points highlight the different errors in the different databases:

- Oracle Error : dbdict:ApprovalDef, field:appr.condition, SQL type is CHAR(1) -- expected to be:RAW(255)
- MSSQL Error: dbdict:ApprovalDef, field:appr.condition, SQL type is CHAR(1) -- expected to be:VARBINARY(255)
- DB2 Error: dbdict:ApprovalDef, field:appr.condition, SQL type is CHAR(1) -- expected to be:VARCHAR(255) FOR BIT DATA

Error message in Oracle:	Solution:
dbdict:ApprovalDef, field:appr.condition, SQL type is CHAR(1) -- expected to be:RAW(255) dbdict:cm3r, field:svc.options, SQL type is VARCHAR2(30) -- expected to be:BLOB dbdict:incidents, field:svc.options, SQL type is VARCHAR2(90) -- expected to be:BLOB	To fix these issues, change the SQL type to RAW (255) or BLOB by using the Dbdict utility. Additionally, you will need to set the “SQL RC” to true to allow the field to store RAD expressions. Note that the stored value of the field in the database is encoded by Service Manager.
dbdict:cm3eventack, field:number, field type is number -- expected to be:character	To fix this issue, change the field type to character by using the Dbdict utility.
dbdict:eventin, field:evnumber, field type is number -- expected to be:character dbdict:eventout, field:evnumber, field type is number -- expected to be:character	These fields reside in the descriptor structure field of BLOB SQL type. To fix this issue, change the field type to character by using the Dbdict utility.
dbdict:licenseinfo, field:id, field type is character -- expected to be:number	The licenseinfo table is used to track license information by Service Manager server. This issue should be ignored.
dbdict:svcCatalog, field:id.attach, field type is character -- expected to be:number	This id.attach field is an alias of id field in svcCatalog table. To fix the issue, change the field type to by using the Dbdict utility.
dbdict:FolderRights, field:close, field type is logical -- expected to be:character	The close field is an alias of delete field in FolderRights table. To fix the issue, change the field type to character by using the Dbdict utility.
dbdict:FolderRights, field:delete, field type is logical -- expected to be:character	This issue can be fixed by following the steps in the "Fixing the FolderRights delete field" (on page 31) section.

Fixing the FolderRights delete field

Example: The dbdict for the **FolderRights** table has a **delete** field with the "logical" data type. The Upgrade Utility tries to update the **delete** field with the "character" data type, which has possible values of "always," "never," "workgroup," and "assigned."

1. In the dbdict for the **FolderRights** table, add a field named **delete.tmp** with a data type of **character**, and update the dbdict.
2. Log out of the system and log back in.
3. Make sure that the Complex Update feature is enabled for the **FolderRights** table.
4. In the Database Manager, search for all records in the **FolderRights** table.

5. From the More Actions menu, click **Mass Update**.
6. When you are asked whether you want to update all records, click **Yes**.
7. Click **Complex Update** on the toolbar.
8. In the statements area under **Instructions for action on EACH RECORD**, add statements using standard RAD expressions to migrate data from the **delete** field to the **delete.tmp** field.

   Example:

   ```
   if delete in $file=true then delete.tmp in $file="always" else
   delete.tmp in $file="never"
   ```

9. In the dbdict for the **FolderRights** table, edit the **delete** field and add the **Type** (character) and **SQL Type** (same as the SQL Type automatically assigned for **delete.tmp**).
10. Log out of the system and log back in.
11. In the Database Manager, search for all records in the **FolderRights** table.
12. From the More Actions menu, click **Mass Update**.
13. Click **Complex Update** on the toolbar.
14. In the statements area under **Instructions for action on EACH RECORD**, add statements using standard RAD expressions to migrate data from the **delete.tmp** field to the **delete** field and empty the **delete.tmp** field.

    Example:

    ```
    delete in $file=delete.tmp in $file; delete.tmp in $file=NULL
    ```

15. In the dbdict for the **FolderRights** table, delete the **delete.tmp** field.
16. Log out of the system and log back in.
17. Test the change by updating records in the **FolderRights** table and populating the **delete** field with "always," "never," "workgroup," or "assigned."

Handle key change failure

If the Upgrade Utility fails to apply certain key changes, error information is logged into the except.log file. Review the log file and make appropriate operations:

Error message 1: Failed to add <*key_type*> key: <*field_name*> to table <*table_name*>. You must add it manually.

To handle this error:

1. Type **dbdict** in the Service Manager command line, and press Enter.
2. In the **File Name** field, type the table name indicated by the error message, and click **Search**.
3. Click the **Keys** tab.
4. Position the mouse point in the key name part of an empty key structure, and click **New Field/Key**.

5. Select the appropriate key type from the combo list, and add the appropriate field to the key, as indicated by the error message.
6. Click **Add**, and click **Yes** to confirm.
7. Click the **Keys** tab, and click **OK** to save the change.
8. Repeat steps 1 through 7 for each key that failed to be added.

Error message 2: Failed to update <*key_type*> key: <*old_field_name*> to <*new_field_name*> in table <*table_name*>. You must update it manually.

To handle this error:

1. Type **dbdict** in the Service Manager command line, and press Enter.
2. In the **File Name** field, type the table name indicated by the error message, and click **Search**.
3. Click the **Keys** tab.
4. From the key list, select the key name that is indicated by the error message, and click **Edit Field/Key**.
5. Update the fields in the key according to the field names indicated by the error message.
6. Click **OK**, and click **Yes** to confirm.
7. Click the **Keys** tab, and click **OK** to save the change.
8. Repeat steps 1 through 7 for each key that failed to be updated.

Error message 3: Failed to remove <*key_type*> key: <*field_name*> from table <*table_name*>. You must remove it manually.

To handle this error:

1. Type **dbdict** in the Service Manager command line, and press Enter.
2. In the **File Name** field, type the table name indicated by the error message, and click **Search**.
3. Click the **Keys** tab.
4. From the key list, select the key name that is indicated by the error message, and click **Edit Field/Key**.
5. Click **Delete**, and click **Yes** to confirm.
6. Click the **Keys** tab, and click **OK** to save the change.
7. Repeat steps 1 through 6 for each key that failed to be removed.

In addition, you may encounter the following Unique Key errors:

Error: dbdict:Approval, Unique Key is {"unique.key", "file.name", "name"} -- expected to be: {"unique.key", "file.name", "name", "component"}

Error: dbdict:ApprovalLog, Unique Key is {"counter", "file.name", "unique.key"} -- expected to be: {"counter", "file.name", "unique.key", "component"}

Error: dbdict:displayeventrev, Unique Key is {"screen.id", "language", "event", "event.sig", "sc.revision"} -- expected to be:{"id", "sc.revision"}

Error: dbdict:displayoptionrev, Unique Key is {"screen.id", "language", "txt.bank", "txt.option", "text.sig", "sc.revision"} -- expected to be:{"id", "sc.revision"}

Unexpected errors

If the except.log file or the Upgrade Results list reports any errors other than data type mismatches, review the sm.log file for more information, and if needed, contact Customer Support for assistance.

Step 2: Resolve conflicts

In this step, you are going to resolve the conflicts between your tailored objects and the objects provided in the upgrade package.

Standard conflict resolution process

The following is the standard process of resolving conflicts:

1. To view the conflicts, click **View/Merge Upgrade Results** and search for records with a status of "Renamed."
2. For each renamed object, you can choose one of the following options.
   - **Option 1:** Use your customized object instead of the new object.

     In this case, delete the new object that is prefixed with NEW931 and the copied object that is prefixed with PRE<*version_number*>.
   - **Option 2:** Use the new object instead of your customized object.

     In this case, delete your customized object and the copied object that is prefixed with PRE<*version_number*> and then rename the new object by removing the NEW931 prefix.

     Note: The Mass Choose Upgrade feature can help you replace multiple customized objects with new objects that are prefixed with NEW931. For more information, see the "Using the Mass Choose Upgrade feature" (on page 42).
   - **Option 3:** Merge the changes shipped with the new object into your customized object.

     In this case, find out what changes the new object includes, manually apply those changes to your customized object, and then delete the new object that is prefixed with NEW931 and the copied object that is prefixed with PRE<*version_number*>.

   Note: The Merge tool, Auto-Merge and Revert options, and third-party three-way compare and merge tools can assist you in comparing the objects and merging the code during the out-of-box upgrade. See "Using the Merge tool" (on page 38) and "Using the Auto Merge and Revert options" (on page 42) for more information.
3. After resolving each conflict, you must mark the object as "Reconciled." Marking an object as reconciled will remove it from the current conflict list and provides the version with which the object was last reconciled.
   Note: The Mark as Reconciled feature can assist you to mark the object as Reconciled. For more information, see "Using the Mark as Reconciled feature" (on page 42).

Display components

The Upgrade Results list also includes conflicts in display components. You can follow the standard conflict resolution process to resolve these conflicts. However, you must pay special

attention to following types of display components:

- The Display RAD application (RAD=display)
- displayscreen records
- displayoption records
- displayevent records

Display application

The Display RAD application (RAD=display) is a Service Manager RAD application that provides access to RAD features without requiring RAD programming skills or RAD licensing.

If the Display RAD application appears in your upgrade results list, perform conflict resolution on that application as part of the standard conflict resolution process. It is highly recommended that you use the new version of the application.

Display screen records

Display screens are individual records identified by a unique screen ID. The displayscreen records define the attributes of a screen and provide access to the individual records for options and events. A display screen is not the same as a form.

Caution: There are triggers attached to the displayscreen file. Changes to the records in this file affect their associated display options and events.

If any displayscreen objects appear in your upgrade results list, perform conflict resolution on those records by following the standard conflict resolution process.

Display options and display events

The Upgrade Utility upgrades the display components in the same manner as all other components. The **displayoption** table and the **displayevent** table have a unique identifier, stored in the ID field. The upgrade process assigns an ID to every display option following the pattern: **<screen id>_ <action>_<number>**, where the screen ID and action are from the display option (or event), and the number is an optional field added when multiple options have the same screen ID and action.

If options that have been added to your system have the same action as others in the same display screen, the upgrade process assigns a <number> in the order of the options' GUI option number.

If an added option was not the last option in terms of GUI option number, the upgrade process does not add the additional numbers in the ID field in the same order as they would have for an out-of-box system. The upgrade process renames the added option and any option after it (in GUI option order), it does not upgrade them automatically.

To ensure that this type of renaming does not happen in future upgrades, when performing conflict resolution on these options, use the ID of the renamed option, **NEW931<screen id>_<action>_ <number>** and manually change the identifier of the added options. Rename all other options to match the ID of the renamed ones.

When renaming an option, use an identifier to specify that this is a customized option, added for your installation. For example, an ID might look like: "**apm.edit.problem_do nothing_ACME1**".

This table gives an example of part of the display screen conflict resolution for **apm.edit.problem**.

Example conflict resolution for the apm.edit.problem display option

Screen ID	GUI Action	Upgrade Action	
300	do nothing	update	Name: apm.edit.problem_do_nothing_1 Result: This item was updated correctly. User Action: No action necessary.
400	do nothing	update	Name: apm.edit.problem_do_nothing_2 Result: This item was updated correctly. User Action: No action necessary.
450 This is an option you added.	do nothing	rename	Name: apm.edit.problem_do_nothing_3 Result: This item was renamed. It is your customized option. User Action: Rename this object to give it a unique new name, such as: apm.edit.problem_do_nothing_ACME1 Name: NEW931apm.edit.problem_do_nothing_3 Result: This is the new SM931 option. User Action: Perform conflict resolution. To perform conflict resolution, open apm.edit.problem and look at the options. Compare this option with apm.edit.problem_do_nothing_3 and NEW931apm.edit.problem_do_nothing_3
500	do nothing	The upgrade ignores this option.	Result: This option does not appear in the reports. User Action: Perform conflict resolution. To perform conflict resolution, open apm.edit.problem and look at the options. Compare this option with apm.edit.problem_do_nothing_3 and NEW931apm.edit.problem_do_nothing_3

RAD applications

You can follow the standard conflict resolution process to resolve conflicts for RAD applications. In the Upgrade Results list, conflicts for RAD applications are displayed with an object type of "Application Cluster." These types of objects are different from the other object types because a RAD application is made up of records from several different tables.

Options for resolving RAD application conflicts

Records with the "Application Cluster" object type appear in the Upgrade Results list only in either of the following scenarios.

Note: The codes for RAD applications for which "Current Release Level" is not marked as "SM 9.31" (for example "7.1"), are already current with the ones in Service Manager9.31. Therefore, you should not change the "Current Release Level" field to "SM 9.31". There are two cases in which a RAD application's "Current Release Level" may be marked as "SM 9.31":

1. The RAD applications which are new in Service Manager 9.31.
2. The RAD applications whose code has been changed in Service Manager 9.31.

Scenario	Action
Your organization has a RAD license and has tailored the RAD application in question	Similar to the "Standard conflict resolution process" (on page 34), you can choose one of the following options when resolving a RAD application conflict: • **Option 1:** Use your customized object instead of the new object. In this case, delete the new RAD application that is prefixed with NEW931 and the copied RAD application that is prefixed with PRE<*version_number*>. • **Option 2:** Use the new object instead of your customized object. In this case, delete your customized RAD application and the copied RAD application that is prefixed with PRE<*version_number*> and rename the new RAD application by removing the NEW931 prefix. • **Option 3:** Merge the changes shipped with the new object into your customized object. In this case, find out what differences exist between the RAD applications, manually update the customized RAD application, and compile the code. Then, delete the new RAD application that is prefixed with NEW931 and the copied RAD application that is prefixed with PRE<*version_number*>. For example, you can use the 'Compare All' feature in the RAD Editor to assist you in identifying which panels have changed and then manually update panels as necessary in that RAD application.
Your organization has applied a hotfix or patch that included a RAD application, which changed the existing RAD application when it was loaded into the system.	Option 3 is not applicable to you. You can choose to either keep the new version of the RAD app that came with the upgrade package, or keep the RAD application that was loaded with the hotfix or patch. In most cases, you need the newest version of the RAD application that came with the upgrade package. In this scenario, resolving RAD application conflicts involves two operations: deleting a RAD application, and renaming a RAD application. To delete a RAD application:

Scenario	Action
	1. Open the RAD application in the RAD Editor. 2. Click **Delete**. 3. Click **Delete All**. To rename a RAD application: 1. Open the RAD application in the RAD Editor. 2. From the More Actions menu, click **Copy/Rename**. 3. Enter the name of the new RAD application. 4. Click **Rename**.

Using the Merge tool

For each object marked as "Renamed" in the Upgrade Results list, the Upgrade Utility generates XML objects for the three versions of the object: base, customer, and upgrade.

Version	Location	Description
base	*Upgrade*\3waymerge\work\base	An XML representation of every object that has been signatured in the pre-upgrade out-of-box version.
customer	*Upgrade*\3waymerge\work\customer	An XML representation of all objects that were tailored in the customer version and resulted in a conflict during the upgrade.
upgrade	*Upgrade*\3waymerge\work\upgrade	An XML representation of the object provided by the upgrade package of all objects that resulted in a conflict.

Each of the three folders described above contains a sub-folder for each signatured table. You can find the XML representations of the objects in the table within these sub-folders.

The built-in, two-way/three-way Merge tool allows you to examine the upgrade and customer versions of a record in a side-by-side view as well as the base, upgrade, and customer versions of a record in a three-way view. This will help you to determine which changes to include in the final record.

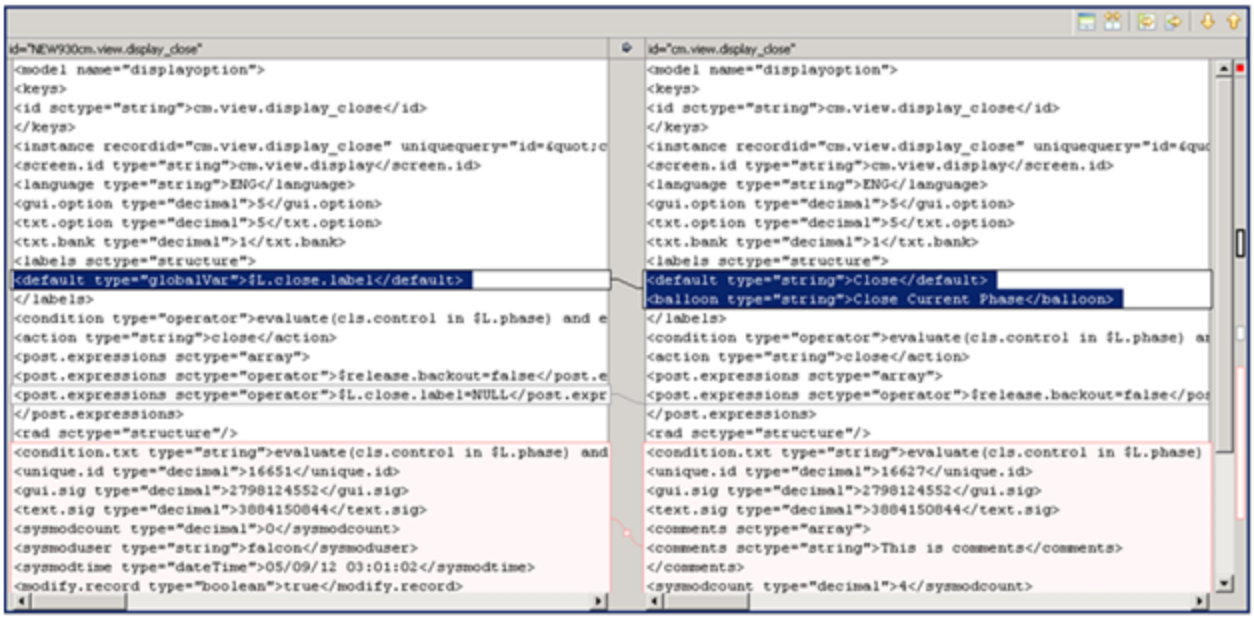
Note: The tool does not work with RAD applications.

This tool assists the conflict resolution process in these two ways:

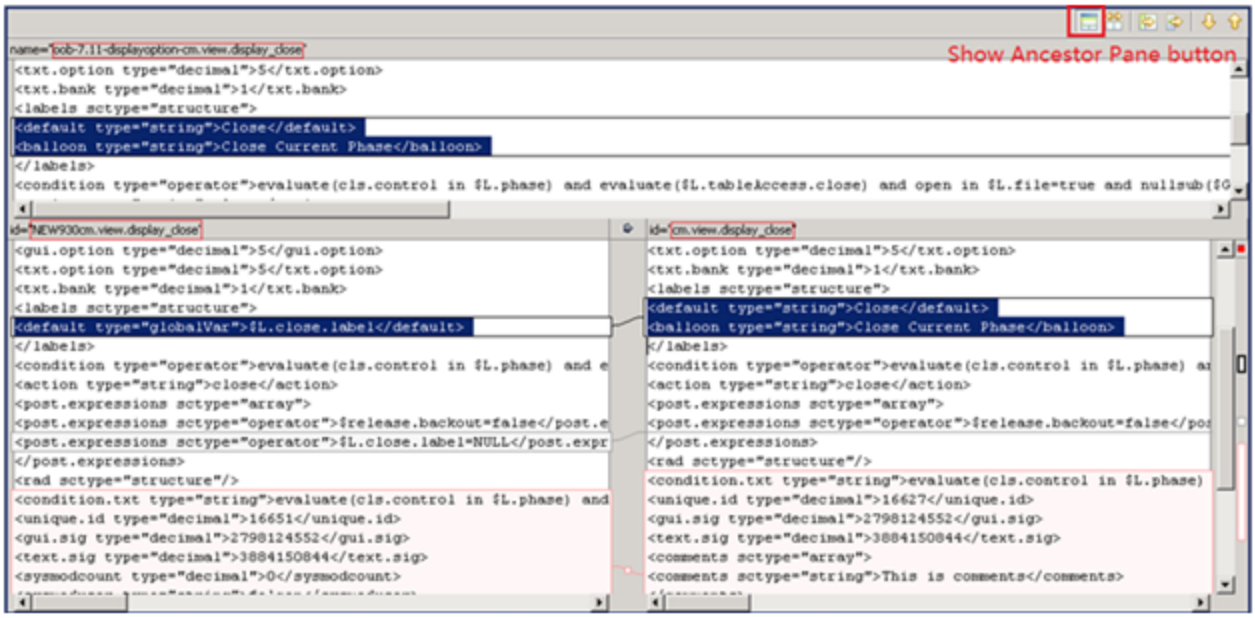
- It allows you to identify where changes are located before you can visually compare the objects and to make changes manually, such as in format records.
- It allows you to identify and merge changes directly between objects, such as ScriptLibrary records.

To use the Merge tool, follow these steps:

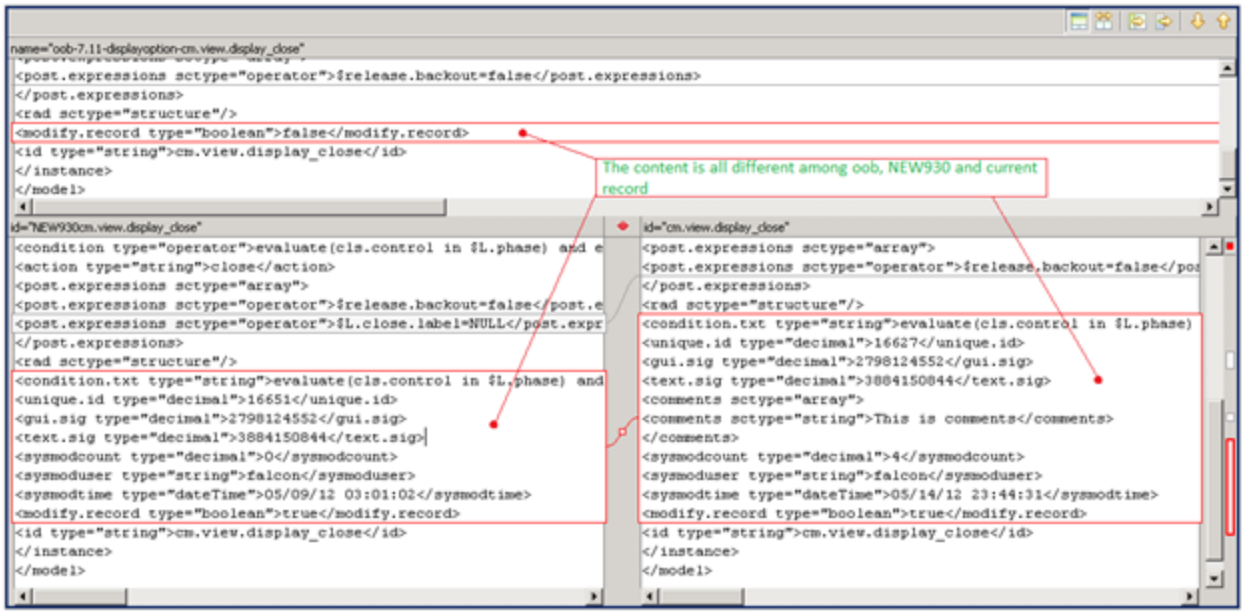
1. In the UPGRADE UTILITY section, click **View/Merge Upgrade Results**.
2. In the **Result** drop-down list, select **Renamed**.
   Note: The Two-way/Three-way Merge tool is available only for “Renamed” records.
3. Click **Search**.
4. A list is returned that displays all the result records of the “Renamed” type. Select a record that you want to examine, click **More** or the **More Actions** icon, and then click **Merge**.
5. The current default merge mode is Three-Way Merge mode. The merge option is not available for format records. Instead, you can click **Compare** option from the **More Actions** menu to start the merge tool in read-only mode. You can use this mode to identify the differences in the side-by-side view or three-way view and then merge records manually in Forms Designer.



   a. In Three-Way Merge mode, click the **Show Ancestor Pane** button to show the base record reference. Click the button again to hide the base record reference.



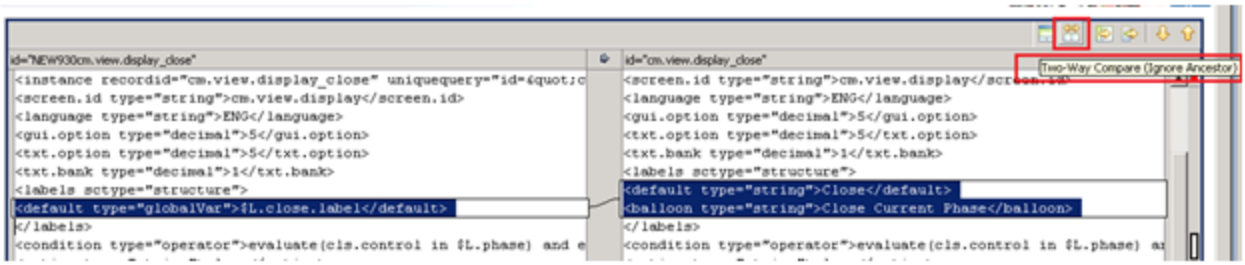
b. In Three-Way Merge mode, if the records of base, upgrade, and customer versions are all different, the background color is red. If you put the cursor into the records with the red background and then click the **Copy Current Change from Left to Right** button, the selected records from the left pane will be appended to selected records in the right pane.



c. In Three-Way Merge mode, if the records of the customer version are identical to the base version, but different from the upgrade version, the background color is blue. If you put the cursor into the records with the blue background, and then click the **Copy Current Change from Left to Right** button the selected records from the right pane will be replaced by the selected records from the left pane.

d. In Three-Way Merge mode, if the records on the left contain red background records, and you then click the **Copy All from Left to Right** button, the records in the right pane with blue background will be replaced with selected records from the left pane.
Note: The right records with red background will not be changed. To do this, you will need to switch to Two-Way Merge mode to use Copy All from Left to Right feature.

6. To enter Two-Way Merge mode, click the Two-Way Compare (Ignore Ancestor) button. In this mode, the Show Ancestor Pane button is disabled.
Note: When you do this, the button label changes to Three-Way Compare. Click this again to revert to Three-Way Compare mode.



a. In Two-Way-Merge mode, if one record differs upgrade and customer versions, you can click **Copy Current Change from Left to Right** button to replace a selected record from the left pane to the right pane.

b. In Two-Way-Merge mode, click **Copy All from Left to Right** button to replace all records in the right pane with the records from the left pane.

Using a third party tool to visually compare objects

You may also visually compare the three versions of each object using a three-way compare and merge tool outside Service Manager, and then merge them manually in Service Manager. For example, you can use a tool, such as KDiff3 for Windows, to compare and merge objects.

To download and learn about KDiff3, visit the KDiff3 Web site.

A brief example of using KDiff3 to compare the three versions of an object is provided in the following steps:

1. Install KDiff3 on the Service Manager server host.
2. Open KDiff3. For more information, refer to the KDiff3 documentation.
3. For the A(Base) parameter, specify the path to the Upgrade\3waymerge\work\base folder.
4. For the B parameter, specify the path to the Upgrade\3waymerge\work\customer folder.
5. For the C parameter, specify the path to the Upgrade\3waymerge\work\upgrade folder.
6. Click **OK**.
7. Navigate the folder structure and double-click the file that is named after the object you want to merge.
8. Compare the three versions of the object in the 3-way compare view.
9. Manually apply the changes you have identified to the object in Service Manager.

Using the Auto Merge and Revert options

During upgrade processing, the Upgrade Utility attempts to merge your objects with the corresponding objects provided with the upgrade. You can search for records with a result type of "Auto Merged" in the Upgrade Results list to review these objects.

Note: An auto-merged object needs to be thoroughly tested for syntax and functional integrity.

If you encounter issues with objects that have been auto-merged or reconciled, you can use the Revert feature from the More Actions menu to restore "Auto Merged" or "Renamed" objects to their previous states. Use **Revert** to restore one object and **Mass Revert** to restore multiple objects. After restoring an object, you can also re-attempt to auto-merge that object using the **Auto Merge** or **Mass Auto Merge** option from the More Actions menu.

Using the Mass Choose Upgrade feature

During the Upgrade process, you can use the Mass Choose Upgrade feature to overwrite your systems old objects with the newer versions from the upgrade utility. You can use this feature to quickly update the objects of the following statuses, which are generated during the upgrade:

- Auto Merged
- Renamed
- Previously Reconciled
- Reconciled

To use the Mass Choose Upgrade feature, follow these steps:

1. In the UPGRADE UTILITY section, click **View/Merge Upgrade Results**.
2. In the Result drop-down list, filter the set of objects (Auto Merged; Renamed; Previously Reconciled; Reconciled) on which you wish to use the Mass Choose Upgrade feature and then click **Search**.
3. If more than two objects exist in the resulting search, click the **Mass Choose Upgrade** button from **More Actions** menu in the returned list and then click **Yes**.

After you click Yes, the objects that you selected will be updated with the contents of the newer versions from the upgrade utility.

Using the Mark as Reconciled feature

During the Upgrade process, you must mark conflicting objects as “Reconciled” after resolving each conflict. To help with this process, you can use the Mass Mark as Reconciled feature to mark multiple objects as “Reconciled.” You can use this feature on objects with the following statuses:

- Auto Merged
- Renamed
- Previously Reconciled

To use the Mass Mark as Reconciled feature, follow these steps:

1. In the UPGRADE UTILITY section, click **View/Merge Upgrade Results**.
2. In the Result drop-down list, filter the set of objects (Auto Merged; Renamed; Previously Reconciled) on which you wish to use the Mass Mark as Reconciled feature and then click **Search**.
3. If more than two objects exist in the resulting search, click the **Mass Mark as Reconciled** button from the **More Actions** menu and then click **Yes**. After you click Yes, all objects that you selected will be marked as “Reconciled” and removed from current conflict list.

   Note: If only one object exits in the resulting search, or if you want to resolve conflicts for the selected objects individually, you can use the **Mark as Reconciled** button from the **More Actions** menu on each object instead of the **Mass Mark as Reconciled** button.

Step 3: Perform additional manual tasks

This section lists changes that cannot be automated by the Upgrade Utility and changes that are required only for certain customers. Make these changes before testing and backing up your system.

The Upgrade utility does not automatically clean up artifacts that were left over by the upgrade, such as objects that were prefixed with NEW931 or PRE<version_number> which are copied and renamed from pre-upgrade objects. You must manually delete those objects against the exported list. Otherwise, the system may not work as expected. To find those objects, search the Upgrade Results list for records with a result type of "Auto Merged" "Previously Reconciled" "Reconciled" or "Renamed" and then export the list to an Excel file.

To make added or upgraded display options work as expected, you must manually re-save all the related display screens. To find those display screens, you can search the Upgrade Results list for display options records with a result type of "Added" "Auto Merged" "Previously Reconciled" "Renamed" "Reconciled" or "Upgraded" and export the list to an Excel file.

Required application changes

Review the following list and make changes manually as appropriate.

Required changes

Change	Introduced in version	Action required
Problem Management: Profile records now contain settings to control access to Known Error Tasks.	7.10	After you update your applications, the settings in the Known Error Tasks section of the profile records are not populated. Therefore, no users can access this feature. You must update your profile records to enable users to view and modify records within Known Error Tasks.
Service Desk: The Service Desk application uses these fields to list contacts for new calls:	7.10	Specify the contact for the **callback.contact** and **contact.name** fields for each existing Service Desk ticket in your system.

Change	Introduced in version	Action required
• Contact for this call (callback.contact) • This call is for (contact.name) Once you select a contact (callback.contact), the other field, if left blank, defaults to the callback.contact.		You can use the Complex Update feature to fill these fields automatically if there are a huge number of records.
Change Management with Web Services: If your Change Management Web Services is tailored, the following fields in the cm3r table must be exposed following an upgrade of the server and client: header,assign.dept affected.item closureComments These are now mandatory fields when opening or closing a Change record, so they must be accounted for in any Web services definitions. HP also recommends that the emergency field in the cm3r table be exposed as well, to support emergency Changes.	9.20	To expose the fields, add them to the appropriate extaccess records for the cm3r table published in the Web services API. header,assign.dept (suggested Caption=AssignmentGroup) affected.item (suggested Caption=Service) closureComments (suggested Caption=ClosureComments) emergency (suggested Caption=Emergency)
Incident Management with Web Services: If your Indicent Management Web Services is tailored, the affected.item field in the cm3r table must be exposed following an upgrade of the server and client. This is now a mandatory field when opening or closing an Incident record, so it must be accounted for in any Web services definitions.	9.20	To expose the field, add it to the appropriate extaccess records for the probsummary table published in the Web services API. affected.item (suggested Caption=Service)

Localize Service Catalog items (optional)

Starting with Service Manager 7.11, the format of the Service Catalog data changed to allow for localization. For each Service Catalog item, the Upgrade Utility creates several records in the **svcDisplay** table for localized versions, each for a language available on your system. The contents in each record are copied from the original English record, but with a specific language code.

To localize your Service Catalog items:

1. Type **db** in the Service Manager client command box, and then press ENTER.
2. In the **Table** field, type **svcDisplay** and click Search.
3. From the **Language** drop-down list, select the language into which you want to localize your catalog items, and then click **Search**.
4. Select each of the records, and replace the English text in the fields with the appropriate translation.
5. Click **Save**.

Step 4 (optional): Modify automatically fixed data

The Upgrade Utility scans the existing records for duplicates when imposing a "Unique" or "No Duplicates" key on a field. If duplicates are found, the utility fixes the duplicates and continues the upgrade process. To do this, it differentiates duplicate items by appending suffixes such as -dup1, -dup2...-dup*x*.

To correct the auto-modified values to your own values:

1. Open the sm.log file.
2. Search for **-dup** and locate a message that resembles the following:

   `RTE I scan: fix duplicates, changed column` *`COLUMN_NAME`* `in TABLE_NAME to new value:` *`ORIGINAL`*`-dup1`

   This message indicates that duplicates were found on the ***COLUMN_NAME*** field of the ***TABLE_NAME*** table, and all duplicates other than the first instance were changed from ***ORIGINAL*** to ***ORIGINAL*****-dup1**, ***ORIGINAL*****-dup2**, and so on.
3. Change the auto-modified value to your own value, either by using the Service Manager client or by directly updating data in the RDBMS.
4. Search for the next occurrence of **-dup** to fix the next duplicate item.
5. Repeat steps 2 through 4 to change all the values that were updated by the upgrade utility.

Step 5: Return the system to normal operation

After the upgrade, the system may exhibit abnormal behavior until you return it to its normal operating environment.

Important: All upgrade-related files should be removed from the system after successful completion of the upgrade. For information on removing these files, see "Step 6: Clean up upgrade objects" (on page 52).

To return to a normal operating environment:

1. Log out.
2. Stop the server.
3. Remove the comment from the system.start entry from the `sm.cfg` file.
4. Restore all the parameters that you updated in **sm.ini** and **sm.cfg** to their original state.

5. Add all parameters that are documented as necessary for your upgraded system to run properly.
6. Restart the Service Manager server.
7. Log on.
8. Wait for the background processes to finish.
9. Regenerate the IR keys on the **incidents** table. Regenerating an IR key may take a long time, so you can schedule this regeneration to occur during scheduled maintenance. For more information, see the instructions on how to regenerate IR keys in the Service Manager Help Server documentation.

Note: Service Manager does not recompile indexes in your RDBMS. If your RDBMS is not configured to recompile indexes automatically after index changes, you must recompile your indexes manually.

Step 6: Test the system (functional testing)

After you resolve all conflicts, test the upgraded system and verify that it functions properly. If there are problems that you cannot resolve, contact HP Customer Support.

Step 7: Back up the system

Make a checkpoint backup of the data files to enable you to restore from this point, if necessary. Refer to the documentation for your RDBMS for backup instructions.

Chapter 6

Creating and applying the custom upgrade

Now you have resolved all the exceptions and conflicts. It is necessary to package all the reconciled objects together into a "custom upgrade" so that they can be moved into the production environment automatically.

Topics in this section include:

Step 1: Build a custom upgrade

If you have followed all the steps to this point, you have already run the upgrade against the duplicate system you created of your production system, and you have performed reconciliation for that system. This section describes how to build a custom upgrade, which is then applied to your test system and production system.

To build the custom upgrade:

1. Log on to the reconciled development system.
2. Before you begin to build the custom upgrade, you need to back up (or duplicate) the production system. For information on duplicating your system, see "Step 4: Duplicate the production environment" (on page 19).
3. Create a folder for the custom upgrade on the Service Manager server. You can assign any name you like to this folder. However, for documentation purposes, this folder will be referred to as the *CustomUpgrade* folder. Ensure that the *CustomUpgrade* folder is empty.

   Note: If you are connecting to the Service Manager server from a client that is installed on a remote client computer, make sure that the folder is created on the Service Manager server instead of the client computer.
4. Type **smupgrade** in the Service Manager client command box, and press **ENTER**. This opens the Upgrade Utility.
5. In the **UPGRADE UTILITY** section, click **SERVICE PACK**.
6. In the **Upgrade Processing** section, click **Create an Upgrade**.
7. On the Welcome screen, click **Next**.
8. **Message: What is the name of this release (e.g. SM7).**

   Specify a name for the custom upgrade package, and click **Next**.

   Example: SM931.

9. **Message: Please select which currently supported language(s) in addition to English are to be included in this build**

   On this screen, a list displays all the non-English languages installed on the system. By default, these languages are set to **true**. Select **false** for the languages that you do not want to package into the custom upgrade, and click **Next**.

   Note: You **cannot** go back and rerun the upgrade to select additional languages. You must ensure that all applicable languages have true selected.

10. **Message: Where do you want the upgrade files to be exported?**

    Type the fully-qualified path to the folder where the Upgrade Utility should create files, and then click **Next**. This should be the path to the empty *CustomUpgrade* folder that you created in step 3.

11. **Message: Which patch file should be used to build the upgrade?**

    Select the most recent patch record from the menu (**SM93**), and click **Next**.

12. **Message: Take which action?**

    Keep the default setting **Complete Upgrade Build**, and click **Next**.

    Note: The other options are not available for use at this time.

13. **Message: Warning. This process will destroy any existing upgrade definitions on file. Proceed?**

    Click **Yes** to continue.

14. When you receive a "Finished creating the transfer files for the upgrade" message, the Upgrade Utility has finished the data packaging.

Step 2: Apply the custom upgrade to the test system

You need to apply the newly-created custom upgrade to your test system for user acceptance testing.

Note: If you experience problems, such as a power failure or a network connection error while upgrading the system, you need to restore the database before attempting to run the upgrade again.

To upgrade the custom upgrade:

1. On the test system, which is a copy of your production system, complete all the preparation tasks in "Preparing for an upgrade" (on page 17).
2. Load **preupg.bin** and **transfer.bin**.

   Note: You can load these two files with the steps that you followed in "Step 1: Load the application upgrade files" (on page 23) except that, instead of using the files that you extracted from the product DVD, you must use the files in your *CustomUpgrade* folder.

   Important: Before loading these files, you also need to disable the **Client side load/unload** option from **Window** > **Preferences** > **HP Service Manager**.

3. Type **smupgrade** in the Service Manager client command box, and press **ENTER**. This opens the Upgrade Utility.
4. In the **UPGRADE UTILITY** section, click **SERVICE PACK**.

5. In the **Upgrade Processing** section, click **Apply an Upgrade**.
6. On the Welcome screen, verify that the **Applications version upgrading from** field displays your current application version (**SM7**), and then click **Next** to continue.

   Note: If this screen does not display the correct version, do not continue with the upgrade. Instead contact HP Software Customer Support.
7. **Message: Please select the language(s) in addition to English to be upgraded.**

   You see a list that displays all the supported non-English languages. By default, the languages that are installed on your Service Manager system are set to **true**. Select **false** for the languages that you do not want to upgrade, and click **Next**.
8. **Message: Are you going to use this system to create a custom upgrade for another system?**

   We are applying a custom upgrade that includes objects that you consider final, so select **No**, and then click **Next**.
9. **Message: What is the fully qualified path to the HP Service Manager Upgrade patch files?**

   By default, the text box displays the fully qualified path to the *CustomUpgrade* folder on the Service Manager server. Keep the path unless it does not point to the correct folder, and then click **Next**.
10. **Message: When HP Service Manager Upgrade doesn't recognize an object it should**

    We are applying a custom upgrade that includes objects that you consider final, so select **Replace your version of the object with the HP Service Manager's version of the object**, and then click **Next**.
11. The upgrade is now ready to start. Click **Next**.
12. When you are asked whether you want to proceed, click **Yes**.
13. The Upgrade Utility displays the status when the upgrade is being processed.
14. When you receive an "UPGRADE IS COMPLETE" message, the Upgrade Utility has finished the data processing.

    Caution: If new schedulers are displayed on the status window after the upgrade completes, do not stop the system! Let the schedulers finish the background data upgrade before stopping the system.
15. Exit the client and restart the server.
16. Open the **scversion** table in the Database Manager, and verify that the **Application Version** field is **9.31.0022**. If this field displays a value other than **9.31.0022**, check the log files to identify the issue that occurred.

Note how long it takes to apply the custom upgrade, so you will know how long the production system will be unavailable during the production upgrade. You can check the log file for an estimate.

Tables and records that are not upgraded by the Upgrade Utility

The Upgrade Utility does not automatically upgrade all tables and records. The patches record lists the tables and records that are packaged into the custom upgrade. Customizations made to any other tables or records will not be part of the custom upgrade. To make sure that the objects that you have reconciled are moved to the production system, verify that these objects are in the patches record. If not, you can do one of the following:

- Create an unload file containing those objects by adding them to an unload script or using the standard Service Manager Unload/Export Facility,
- Make the same changes manually by directly modifying the objects on the production system. For records that you might have deleted, you can either build a purge script for those records or delete the records manually on the production system.

Step 3: Perform additional manual tasks

This section lists changes that cannot be automated by the Upgrade Utility and changes that are required only for certain customers. Make these changes before testing and backing up your system.

The Upgrade utility does not automatically clean up artifacts that were left over by the upgrade, such as objects that were prefixed with OLD<version_number> which are copied and renamed from pre-upgrade objects. You must manually delete those objects against the exported list. Otherwise, the system may not work as expected. To find those objects, search the Upgrade Results list for records with a result type of "Forced" and then export the list to an Excel file.

To make added or upgraded display options work as expected, you must manually re-save all the related display screens. To find those display screens, search the Upgrade Results list for display options records with a result type of "Added" or "Forced" and export the list to an Excel file.

Step 4: Test the custom upgrade

After you apply the custom upgrade on the test system, perform user acceptance testing for all features, especially customized applications.Test the upgraded system with the new Service Manager client to verify any changes you have made in reconciliation. If the upgrade process has any problems, you need to contact HP Customer Support. After you complete testing of the upgraded system, you can use it to upgrade your production system.

To test the custom upgrade:

1. Return the system to a normal operating environment.
2. Install and configure the Service Manager client for the target version (see instructions in the *HP Service Manager Installation Guide*).
3. Use the new Service Manager client to log on.
4. Review the features described in the Service Manager 9.31 online Help.
5. Use the new Service Manager client to thoroughly test the upgraded system. Test all features that your users will access. Pay particular attention to areas that were modified on your system.

Step 5: Apply the custom upgrade to the production system

After you test the custom upgrade on the test system, you need to apply the newly-created custom upgrade to your production system. This process is identical to the one you followed when applying your upgrade to your test system.

Warning: Do not apply an upgrade to your production system if it has not been thoroughly tested.

When you upgrade the production system:

- The production system should not be available to users while you are applying the custom upgrade.
- Ensure the upgrade files you created are accessible to the production system (the files are located on the same server).
- If you transfer the files to your production system by FTP, set FTP to binary mode.

- If you experience problems, such as a power failure or a network connection error while upgrading the system, you need to restore the database before attempting to run the upgrade again.

Step 6: Clean up upgrade objects

After you successfully apply the custom upgrade to your production system, you can run the purge tool to remove temporary objects that were generated by the Upgrade Utility.

To run the purge tool:

1. Type ***aapm.upgrade.purge** in the Service Manager client command text box. Press **Enter**.
2. Click **I'm done, and I want to remove the upgrade files completely**.
3. Click **OK**.

Chapter 7

Troubleshooting

Before you contact HP Service Manager Customer Support, try the following troubleshooting instructions to identify and resolve your issues.

Troubleshooting: The Upgrade Utility appears to stop responding

Symptoms

The Upgrade Utility may appear unresponsive during the upgrade process. This issue may exhibit the following symptoms:

- The Windows Task Manager indicates that Service Manager client is not responding.
- The Upgrade Utility does not show the progress after you click **Next** to start the upgrade execution.
- The Upgrade Utility appears to stop at a percentage of completion.

Resolution

This is normal and does not indicate a problem with the upgrade. Additionally, the percentage on the status screen does not accurately indicate the actual progress.

Troubleshooting: The client session was terminated during an upgrade

Symptoms

The upgrade failed with "Session no longer valid" error when running in foreground mode. This issue is most likely to occur when you are creating or applying a custom upgrade.

The client session was terminated during an upgrade. The Service Manager client may temporarily lose heartbeat when the Upgrade Utility is running. If the server cannot detect the heartbeat for a certain amount of time, it disconnects the client session.

Resolution

To resolve this issue, restore the database to the latest pre-upgrade state, add a `sessiontimeout:1200` parameter to the sm.ini file, and then restart the upgrade. See also "Step 5: Update Service Manager configuration files" (on page 19).

Note: We recommend that you set the timeout period to a length of time that is long enough for an upgrade phase to complete.

Troubleshooting: Unexpected errors during an upgrade

Symptoms

An unexpected error stopped the upgrade, such as the Service Manager server running out of memory, stack overflow, power outage, and network failure.

Resolution

Fix the issue that stopped the upgrade, restore the database to the latest pre-upgrade state, and then restart the upgrade.

Troubleshooting: Upgrade failed with a "Not enough shared memory available" error

Symptoms

The upgrade process ended abruptly when you were applying an upgrade, and the following error message was logged into the sm.log file:

```
E big_alloc: Not enough shared memory available to allocate <number>
bytes
```

Resolution

Restore the database to the latest pre-upgrade state, increase the size of the shared memory (see "Step 5: Update Service Manager configuration files" (on page 19)), and then restart the upgrade.

Troubleshooting: Upgrade failed with a "signal 11" error

Symptoms

The upgrade process ended abruptly and you received the following error message when you were applying an upgrade:

```
SOAP Fault occurred: A signal 11 was raised in native code. Client
terminated.
```

Resolution

To resolve this issue:

1. Stop the Service Manager server and restore the database to the latest pre-upgrade state.
2. Add a **jsgctrigger:67108864** parameter to the sm.ini file.
3. Start the Service Manager server and restart the upgrade.

Troubleshooting: Database transaction log full

Symptoms

The process ended abruptly when you were loading data or applying an upgrade. The sm.log file contains error messages that resemble the following:

```
[Microsoft][ODBC SQL Server Driver][SQL Server] The log file for
database 'DB_Name' is full. Back up the transaction log for the
database to free up some log space. (message,add.schedule)

An error occurred while attemping to add a record (file.load,
add.record.0)
```

Resolution

Running an upgrade generates a huge number of transactions and this is likely to make the transaction log full. Restore the database to the latest pre-upgrade state, refer to the documentation

for your RDBMS to increase the database transaction log size or enable auto-growth, and then restart the upgrade.

Troubleshooting: Automatic merge fails

Symptoms

The Automatic Merge task fails during the upgrade and you receive the following error message:

```
Failed to unzip <zip file path>. Auto Merge skipped.
```

This error message indicates that automatic merge was skipped because of a failure to unzip the required zip file. The Upgrade Utility skips the Automatic Merge task and continues the upgrade process.

Resolution

To rerun the Automatic Merge task after running the Upgrade Utility:

1. In the **Upgrade\3waymerge\oob** folder, find the zip file named after the version that you selected for the **Base Version** when running the Upgrade Utility.
2. Extract the folder named after the **Base Version** that you previously selected, from inside the zip file to the **Upgrade\3waymerge\oob** folder. For example, if you selected **SC6.2** for the **Base Version** previously, extract the **SC6.2** folder from inside **SC6.2.zip**, and place the extracted folder (**SC6.2**) in the **Upgrade\3waymerge\oob** folder.
3. Open the Upgrade Results list and search for records with a type of "Renamed."
4. Select the objects that you want to merge, and click **Mass Auto Merge** from the More Actions list.

Chapter 8

Data scan option

Before you run an upgrade, you can scan a portion of your dataset for certain types of data that may cause issues following the upgrade. The data scan logs information into the sm.log file, including how many data records were scanned, and which records were automatically updated.

Currently, this option allows you to scan for two types of problematic data:

- Null values in fields applied with certain keys
- Mismatches between the data type defined on the field and the data type of the field value

The option available in the Upgrade Utility does not allow you to specify tables that are scanned. The tables that are scanned include the list of tables as noted in the latest patches record. To view the patches record, open the **patches** table by clicking **SERVICE PACK → Update Patch Definitions** in the Upgrade Utility.

For more information about the rtecall functions used during the data scans, and about how you can implement these functions to scan for additional tables, such as **probsummary**, see the latest Service Manager Programming Guide.

Run the data scan option

To run the data scan option:

1. Make sure you have already loaded the application upgrade files into the system. See "Step 1: Load the application upgrade files" (on page 23).
2. Type **smupgrade** in the Service Manager client command box, and then press ENTER.
3. In the **DATA SCAN UTILITY** section, click **Scan for and Fix Incorrect Data**.
4. Click **Next** and follow the instructions.
5. When the process is complete, you receive a report of the tables that were scanned and the incorrect data that was found and corrected. Results information is logged into the datascan.log file, which resides in the folder where you put the transfer.bin file.

View the data scan results

Search the sm.log file for messages relating to the data scan.

Null values disallowed by keys

The data scan option scans for null values that violate the restrictions imposed by a "No Nulls" or "Unique" key. Then the utility:

- Removes the record if null values violate a "Unique" key, or
- Replaces one of the null values with a default value if null values violate a "No Nulls" key.

To modify the auto-modified values to your own values:

1. Open the sm.log file.
2. Search for **deleted a record** or **updated a record** to locate a message that resembles the following:

Message	Description
`scan: scan for nulls, deleted a record that contains null values on a "Unique" key field, table = TABLE_NAME, record = RECORD_CONTENTS`	This message indicates that the utility has removed a record. *TABLE_NAME* shows the name of the table, and *RECORD_CONTENTS* shows all the fields and values of the removed record.
`scan: scan for nulls, updated a record that contains null values on a "No Nulls" key field, table = TABLE_NAME, key = RECORD_IDENTIFIER`	This message indicates that the utility has replaced a null value with a default value. *TABLE_NAME* shows the name of the table, and *RECORD_IDENTIFIER* shows the primary key and its value of the updated record.

3. Change the auto-modified value to your own value or restore the removed record with valid field values, either by using the Service Manager client or by directly updating data in the RDBMS.
4. Search for the next occurrence of **deleted a record** or **updated a record** to fix the next auto-modified record.
5. Repeat steps 2 through 4 to change all the values that were updated by the upgrade utility.

Data type mismatches

The data scan option scans for mismatches between the data type defined on the field and the data type of the field value.

To modify the auto-modified values to your own values:

1. Open the sm.log file.
2. Search for **scan for inconsistent data types, table** and locate a message that resembles the following:

Message	Description
`scan: scan for inconsistent data types, table: "TABLE_NAME", key: "RECORD_IDENTIFIER", field: "FIELD_NAME", value "ORIGINAL_VALUE", type: ORIGINAL_TYPE, value changed to: "NEW_VALUE", type: NEW_TYPE.`	This message indicates that the utility has changed the value of a field in a record. ▪ *TABLE_NAME* shows the name of the table. ▪ *RECORD_IDENTIFIER* shows the primary key and its value of the updated record. ▪ *FIELD_NAME* shows the name of the

Message	Description
	field that was updated. ■ *ORIGINAL_VALUE* shows the original value of the field that was updated. ■ *ORIGINAL_TYPE* shows the original type of the field that was updated. ■ *NEW_VALUE* shows the new value of the field that was updated. ■ *NEW_TYPE* shows the corrected type of the field that was updated.
`scan: scan for inconsistent data types, table: "`*`TABLE_NAME`*`", key: "`*`RECORD_IDENTIFIER`*`", field: "`*`FIELD_NAME`*`", value "`*`ORIGINAL_VALUE`*`", type: `*`ORIGINAL_TYPE`*`, type changed to `*`NEW_TYPE`*`.`	This message indicates that the utility has changed the type of a field in a record. ■ *TABLE_NAME* shows the name of the table. ■ *RECORD_IDENTIFIER* shows the primary key and its value of the updated record. ■ *FIELD_NAME* shows the name of the field that was updated. ■ *ORIGINAL_VALUE* shows the original value of the field that was updated. ■ *ORIGINAL_TYPE* shows the original type of the field that was updated. ■ *NEW_TYPE* shows the new type of the field that was updated.

3. Change the auto-modified value to your own value, either by using the Service Manager client or by directly updating data in the RDBMS.
4. Search for the next occurrence of **scan for inconsistent data types, table** to fix the next auto-modified record.
5. Repeat steps 2 through 4 to change all the values that were updated by the upgrade utility.

Chapter 9

Updating languages at a later time

After you initially run the Upgrade Utility with only part of the languages selected, there are two points later in the upgrade lifecycle where you can specify additional languages to be updated:

- **Before you create a custom upgrade** (then the updates to the selected additional languages will be included in the custom upgrade package).
- **After you apply a custom upgrade** (to perform an upgrade that updates only the languages).

  In this case, first create copies of the production system to be used as development systems and test systems. Then, run the Upgrade Utility, perform conflict resolution, and create a custom upgrade on the development system, which includes only updates to the languages. Eventually, apply the custom upgrade to your production system.

Update additional languages before creating a custom upgrade

To update additional languages before creating a custom upgrade:

1. Finish conflict reconciliation before you continue, because the Upgrade Results list will be cleaned up when you update additional languages.
2. Load the preupg.bin and transfer.bin files into Service Manager. See "Step 1: Load the application upgrade files" (on page 23).
3. Type **smupgrade** in the Service Manager client command box, and press ENTER. The Upgrade Utility opens.
4. In the **UPGRADE UTILITY** section, click **SERVICE PACK**.
5. In the **Upgrade Processing** section, click **Apply an Upgrade**.
6. On the Welcome screen, click **Next** to continue.
7. **Message: Please select the language(s) in addition to English to be upgraded.**

   You see a list that displays all the supported non-English languages. Some languages are already set to true. Make sure that all languages that you want to update are set to true, and then click **Next**.
8. You must complete the rest of the wizard. However, all the subsequent settings will not take effect because you are only updating the languages.
9. If you have already performed conflict resolution, you must continue to resolve additional conflicts for the language updates.

Update additional languages after applying a custom upgrade

To update additional languages after applying a custom upgrade:

1. Load the preupg.bin and transfer.bin files into Service Manager. See "Step 1: Load the application upgrade files" (on page 23).
2. Type **smupgrade** in the Service Manager client command box, and press ENTER. This opens the Upgrade Utility.
3. In the **UPGRADE UTILITY** section, click **SERVICE PACK**.
4. In the **Upgrade Processing** section, click **Apply an Upgrade**.
5. On the Welcome screen, click **Next** to continue.
6. **Message: Please select the language(s) in addition to English to be upgraded.**

   You see a list that displays all the supported non-English languages. Some languages are already set to true. Make sure that all languages that you want to update are set to true, and then click **Next**.
7. Follow the wizard to complete all selections. You can select the same options as you did the first time you ran the Upgrade Utility on a development system.
8. Perform conflict resolution, create a custom upgrade, and apply the custom upgrade, as you do in a standard application upgrade.

Glossary

A

application

Applications are the Service Manager modules and their related configuration files. For example, Incident Management, Change Management, and Inventory Management are Service Manager applications.

B

BLOB/Image

BLOB is a data type for binary large objects in a database system. In certain RDBMS systems like Oracle, this binary data type is called BLOB. In other RDBMS systems like Microsoft SQL Server, this binary data type is called Image.

C

command box

The Service Manager command box refers to the command-line box on the top-left corner of the Service Manager client, which provides quick access to RAD applications.

compatibility matrix

A compatibility matrix defines the software, system, and platform environments that are supported by an HP software product.

conflict

A conflict refers to a situation where the Upgrade Utility identifies an existing object as "changed" (different from its out-of-box version) when updating that object. To avoid overwriting your customization, the utility adds the new version of the object along with your customized version. Until you resolve this situation, Service Manager features may not behave as expected, or may not function at all.

conflict resolution

Conflicts may occur when you are applying the new Service Manager changes to your existing Service Manager installation, which is likely to include customizations, tailoring, and patches. The confilict resolution phase of the upgrade is to reconcile the differences between the customized objects and the objects provided by the upgrade package.

custom upgrade

A custom upgrade is the upgrade build that is created on the development system after applying the upgrade files and resolving conflicts. This custom upgrade is eventually exported from the development system and applied on the production system. A custom upgrade consists of new Service Manager application files that replaced old application files, customized application files that you retained, and merged files that combine prior customization with new application functionality.

D

Data Policy

Data Policy enables System Administrators to apply default values, mandatory fields, and lookup validations to a specific table. These policies, once

set, are enforced across the entire system, regardless of what form is being used to display the data.

data type mismatch

A data type mismatch refers to a situation where the data type of a field in your dbdict does not match the data type of the like-named field defined in the dbdict provided by the upgrade package.

database dictionary

Service Manager maintains a logical view of your RDBMS tables and columns in the database dictionary. The database dictionary describes each table and column in your system and how they are mapped to logical entities within Service Manager.

DDL

In restricted-access RDBMS environments, Service Manager can create database definition language (DDL) describing the changes proposed by your database dictionary records. The RDBMS administrator can then create the necessary tables and columns for Service Manager manually. After the RDBMS has the necessary tables and columns, the Service Manager administrator can then update the database dictionary records to map to the actual RDBMS objects.

detail.log

The detail.log file includes specific information about the upgrade, such as which files are being signed at any time.

development system (environment)

A development system (environment) refers to a Service Manager system that mirrors your current production environment. Use this development system to run the Upgrade Utility and build a custom upgrade. This system should not be on the same machine as the production server.

display screen

Display screens are individual records identified by a unique screen ID. The displayscreen records define the attributes of a screen and provide access to the individual records for options and events. A display screen is different from a form.

E

except.log

The except.log file includes information about any exceptions reported by the upgrade. The except.log file may have important messages about data type mismatches that you should resolve, or database dictionaries that it cannot upgrade.

exception

An exception refers to an error reported by the upgrade. It could be a data type mismatch that failed to be resolved, a dbdict change that failed to be applied, or an unexpected error (for example, a unique key violation) that stops the upgrade. You must resolve all exceptions before you continue to resolve conflicts.

I

index regeneration

Index regeneration occurs when a database administrator recompiles indexes following changes that have been applied to Service Manager keys.

IR key

IR (information retrieval) key is a key type where the fields in the key are indexed by IR Expert. Only one IR key can be used per dbdict record or IR searches on that file do not work. You can concatenate several fields in an IR key.

K

key

Keys are abstract entities that provide a logical view of the indexes in your RDBMS. When you create Service Manager logical keys, the server creates corresponding indexes in the back-end RDBMS.

N

No Duplicates

No Duplicates is a key type where the value of the complete key must be unique in the index or the values of all fields must be null.

Nulls & Duplicates

Nulls & Duplicates is a key type where all fields can be null and the complete key value can be in the index more than once.

O

out-of-box version

The out-of-box version of Service Manager refers to a Service Manager installation that is not customized, tailored, localized, or patched.

P

Peregrine Four (P4)

The legacy Database Management System (DBMS) that stored ServiceCenter data in a proprietary flat-file format. All HP Service Manager data must be converted to a supported Relational Database Management System (RDBMS).

production system (environment)

The production system (environment) is where the custom upgrade is applied.

R

RAD

The RAD (rapid application development) language is the native system language that Service Manager uses to communicate with its varioius routines and processes.

RAD Comparison Utility

The RAD Comparison Utility is a tool that compares two RAD applications and presents the differences onscreen.

RDBMS

The Relational Database Management System (RDBMS) refers to the database system that Service Manager uses for its storage, such as Oracle and SQL Server.

run-time environment (RTE)

The Service Manager run-time environment refers to the binary, load library, or executable layers of the Service Manager server.

S

signature

A signature for an HP Service Manager record is a numerical representation of the record. Any change to the contents of the record causes the signature of that record to change, depending on the definitions in the signaturemake file.

T

tailoring

Tailoring refers to changes made to Service Manager by creating and modifying control records using Service Manager utilities. Tailoring is the normal method of adapting Service Manager to each installation's requirements. Tailoring involves no Rapid Application

Development (RAD) programming or coding changes.

test system (environment)

A test system (environment) refers to a Service Manager system that mirrors your current production environment for testing purposes. Run and verify the custom upgrade on the test system.

U

Unique

Unique is a key type where at least one field in the key must not be null and the value of the complete key must be unique in the index.

unload

The native HP Service Manager export format is the unload file. An unload file stores the database dictionary of Service Manager tables in addition to records.

Upgrade Utility

Upgrade Utility refers to a set of utilities that are shipped with the new software release for upgrading to the new Service Manager 9.30 applications.

upgrade.log

The upgrade.log file includes information about the upgrade status. It indicates where the upgrade is at a specific point. This file contains only the main steps of the upgrade.

Index


Java™
COMPATIBLE