HP Service Manager

Software Version: 9.41

For the supported Windows® and Unix® operating systems

Applications Upgrade Guide (from HP Service Manager 9.2x)

Document Release Date: June 2016

Software Release Date: September 2015

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: https://softwaresupport.hp.com/.

This site requires that you register for an HP Passport and to sign in. To register for an HP Passport ID, click **Register** on the HP Support site or click **Create an Account** on the HP Passport login 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 site at: https://softwaresupport.hp.com.

This website 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 website 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 to sign in. Many also require a support contract. To register for an HP Passport ID, click **Register** on the HP Support site or click **Create an Account** on the HP Passport login page.

To find more information about access levels, go to: https://softwaresupport.hp.com/web/softwaresupport/access-levels.

HPSW Solutions Catalog accesses the HPSW Integrations and Solutions Catalog portal website. This site enables you to explore HP Product Solutions to meet your business needs, includes a full list of Integrations between HP Products, as well as a listing of ITIL Processes. The URL for this website is https://softwaresupport.hp.com/group/softwaresupport/search-result/-/facetsearch/document/KM01702710.

Contents

Chapter 1: Upgrade overview 6
Before you begin an upgrade 6
Application upgrade 6
Application upgrade lifecycle 7
Upgrade phases and sub-phases 8
How does customization affect the upgrade process? 10
Upgrade Utility contents 13
Chapter 2: Planning an upgrade 15
Step 1: Identify the upgrade resources 15
Step 2: Meet the software requirements 15
Step 3: Perform a system health check 16
Step 4: Create development and test environments 16
Step 5: Develop an upgrade strategy 16
Step 6: Meet database requirements 18
Convert all tables and fields from lowercase to uppercase 18
Chapter 3: Upgrade the server and the client on the production environment 19
Chapter 4: Upgrade tasks on the development environment 20
Step 1: Duplicate the production environment 20
Step 2: Update Service Manager configuration files 21
Step 3: Purge existing upgrade files 23
Step 4: Load the application upgrade files 23
Step 5: Run the SQL compare utility 25
Running SQL Compare 26
Add new fields 29
Determine the correct structure 29
Process the NVARCHAR SQL type 30

Step 6: Run the Upgrade Utility 30
Upgrade Utility logs and error messages 32
Step 7: View the upgrade results 34
Description of upgrade results 34
Step 8: Manage the upgrade result data 39
Step 9: Resolve exceptions 40
Data type mismatches 40
Fixing the FolderRights delete field 44
Add new object record failure 45
Handle key change failure 46
Unexpected errors 48
Step 10: Resolve conflicts 48
Standard conflict resolution process 48
Display components 49
Display application 49
Display screen records 50
Display options and display events 50
RAD applications 51
Options for resolving RAD application conflicts 52
Using the Merge tool 54
Using a third party tool to visually compare objects 59
Using the Auto Merge and Revert options 61
Using the Mass Choose Upgrade feature 61
Using the Mark as Reconciled feature 62
Step 11: Perform additional manual tasks 63
Step 12: Return the system to normal operation 65
Step 13: Test the development environment (functional testing) 66
Step 14: Back up the system 66
Step 15: Build a custom upgrade 66
Upgrade Utility logs and error messages 68
Chapter 5: Upgrade tasks on the test environment 70
Step 1: Purge existing upgrade files 70
Step 2: Apply the custom upgrade to the test environment 70

Upgrade Utility logs and error messages 72
Tables and records that are not upgraded by the Upgrade Utility 72
Step 3: Perform additional manual tasks 73
Step 4: Test the test environment 75
Chapter 6: Upgrade tasks on the production environment 76
Step 1: Apply the custom upgrade to the production environment 76
Step 2: Perform additional manual tasks 77
Step 3: Test the production environment 78
Chapter 7: Troubleshooting 79
Troubleshooting: The Upgrade Utility appears to stop responding 79
Troubleshooting: The client session was terminated during an upgrade 79
Troubleshooting: Unexpected errors during an upgrade 80
Troubleshooting: Upgrade failed with a "Not enough shared memory available" error 80
Troubleshooting: Database transaction log full 80
Troubleshooting: Integrations do not work after an application upgrade 81
Troubleshooting: Automatic merge fails 81
Glossary 83
Index 87
Send Documentation Feedback 88

Chapter 1: Upgrade overview

The purpose of this guide is to describe how to upgrade the HP Service Manager 9.2x applications to Service Manager 9.41 Classic applications by using the HP Service Manager Upgrade Utility.

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 HP 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.

Application upgrade

You can upgrade your existing HP Service Manager applications to version 9.41 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 and Change Management are Service Manager applications.

New features that require an application upgrade

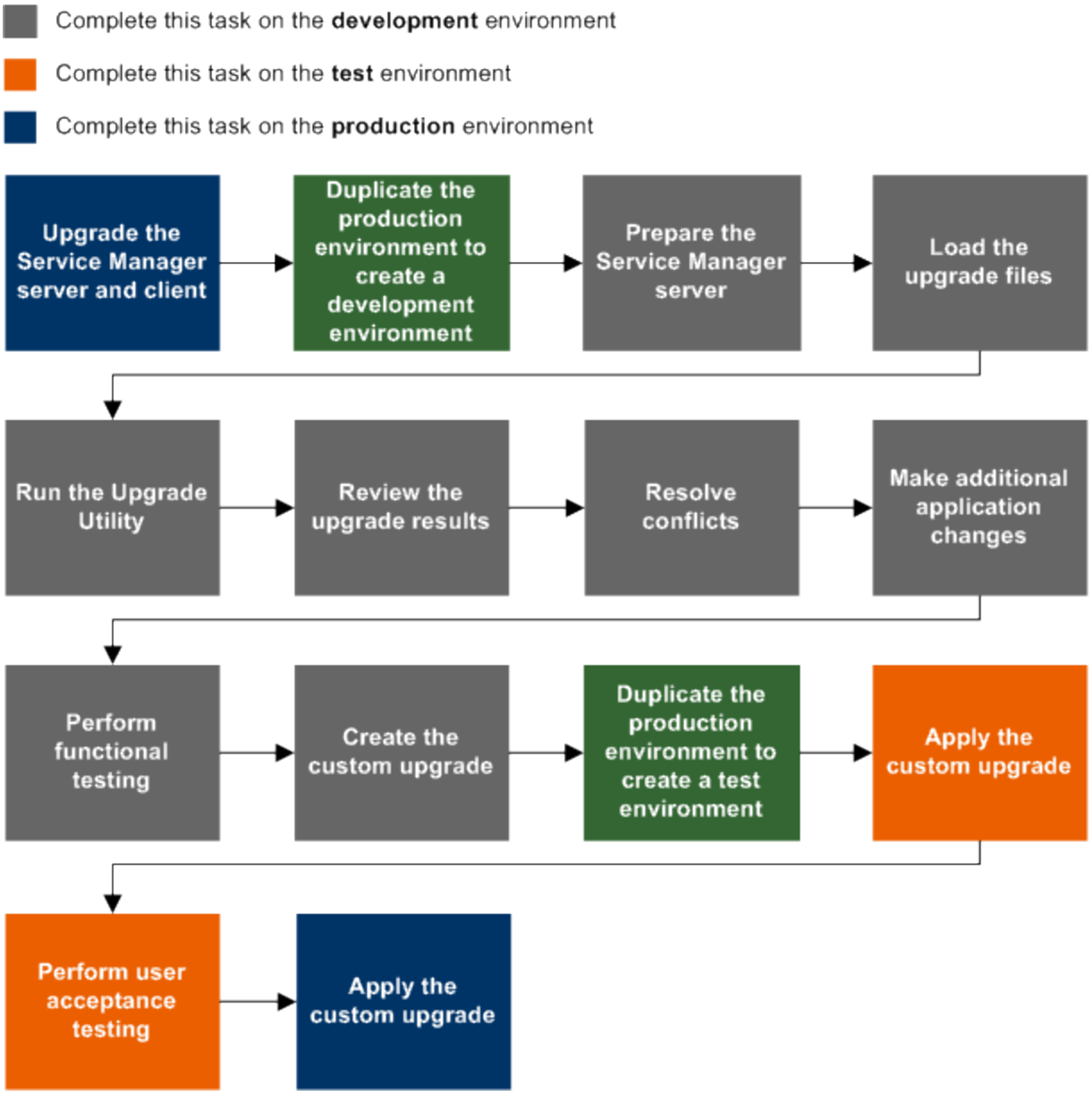
Some new features provided by the release of Service Manager 9.41 require an application upgrade. The following new features provided by the release of Service Manager 9.41 require an application upgrade:

- Enhanced Service Desk, Incident Management, Problem Management, Change Management, Request Fulfillment Management, and Service Level Management based on Process Designer (provided since Service Manager 9.40)

- Process Designer framework
- Smart Analytics (provided since Service Manager 9.40)
- Mobile Applications
- Service Request Catalog (SRC)
- Case exchange
- Simplified interaction
- Logical Name
- Global search
- Service Manager Collaboration
- Accessibility for the embedded Service Manager Calendar
- Service Manager Reports (provided since Service Manager 9.40)
- Service Manager Survey (provided since Service Manager 9.40)
- Entity Relationship Diagram (ERD and Data integrity check)
- Knowledge Management SOLR search engine (provided since Service Manager 9.30)
- The Primary Key and `Not Null` constraints (provided since Service Manager 9.32)

Application upgrade lifecycle

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



Upgrade phases and sub-phases

The following table describes the phases and sub-phases in the entire applications upgrade lifecycle. These sub-phases are logged in the upgrade log files during the upgrade. When an error occurs, the log files can help you find out during which phase and sub-phase the error occurs.

Phase	Sub-phases
Planning and preparation	• Load Transfer

Phase	Sub-phases
Running an out-of-box upgrade	• Pre Upgrade Action Check • Pre Upgrade Action Update • Pre Upgrade Action Purge • Pre Upgrade Action • Load Upgrade File • Upgrade Dbdicts • Load Upgrading Data • Upgrade Data • Post Upgrade Action • Post Upgrade Action Prior to SM930 • Post Upgrade Action Prior to SM940 • Post Upgrade Action Auto Merge • Post Upgrade Action Purge • Post Upgrade Action Update • Post Upgrade Action Notification • Post Upgrade Action Restore
Creating a custom upgrade	• Pre Create Action Check • Build Signatures • Build Distribution • Export Data • Transfer Data

Phase	Sub-phases
Applying the custom upgrade	• Pre Upgrade Action Check • Pre Upgrade Action Update • Pre Upgrade Action Purge • Pre Upgrade Action • Load Upgrade File • Upgrade Dbdicts • Load Upgrading Data • Upgrade Data • Post Upgrade Action • Post Upgrade Action Prior to SM930 • Post Upgrade Action Prior to SM940 • Post Upgrade Action Purge • Post Upgrade Action Update • Post Upgrade Action Notification • Post Upgrade Action Restore

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. When processing object changes, the Upgrade Utility behaves as described in the following tables.

Changed object

Object in DB has been tailored?	OOB version of object matches upgrade package?	Out-of-box upgrade	Custom upgrade
Yes	No	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, the Upgrade Utility marks the object as "Renamed." In both cases, the Upgrade Utility prefixes the object from the upgrade package with "NEW941," and then copies the object to your database and prefixes that object as "PRE<*version_number*>". For example, an object from application version 9.20.0000 would be prefixed with "PRE9.20.000."	The Upgrade Utility marks the object from the upgrade package as “Forced,” and then copies the object in its revision.
Yes	Yes	The Upgrade Utility keeps your local version, even if your version has been tailored and marks your local version as “Kept Customer”.	The Upgrade Utility marks your local version object as “Already Current.”
No	No	The Upgrade Utility overwrites the object in your database with the object from the upgrade package, and marks the object as “Upgraded.”	The Upgrade Utility marks the object from the upgrade package as “Forced,” and then copies the object in its

Changed object, continued

Object in DB has been tailored?	OOB version of object matches upgrade package?	Out-of-box upgrade	Custom upgrade
			revision.
No	Yes	The Upgrade Utility marks the object as "Already Current" regardless of whether this is the first upgrade or a custom upgrade.	

Added object

Object is added in ...	Behavior	Note
DB	The Upgrade Utility always keeps your local version and does nothing else.	The object in your database does not have a corresponding object from the upgrade package.
Upgrade package	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.	The object from the upgrade package does not have a corresponding object from your database.

Dbdict changes: The Upgrade Utility automatically adds new dbdict and merges new fields to existing dbdicts. The Upgrade Utility does not delete any existing field. For field and key changes, check "Field mapping changes" and "Key changes."

Field mapping changes: Normally the Upgrade Utility applies field mapping changes automatically, but there may be some exceptions. For example, when a length change is required, the Upgrade Utility automatically expands the length mapping. However, if the field mapping is a LOB-type change, the Upgrade Utility will not the change the type mapping. For detailed exceptions, check SQL Field Compare results before the upgrade and the except.log file after the upgrade.

Key changes: Normally the Upgrade Utility applies key changes automatically, but there may be some exceptions. For example, the Upgrade Utility automatically adds a new key and updates the existing key of the pre-upgrade out-of-box version. However, if a Unique key has been tailored, the Upgrade Utility will not apply the key change. For detailed exceptions, check the SQL Unique Key Compare Results before the upgrade and the except.log file after the upgrade.

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 HP 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 "SM710-9.41.*00xx* v9.41 *00xx* Upgrade Build *00xx*" indicates the following: • The Upgrade Utility upgrades Service Manager 7.10 and later releases to Service Manager 9.41. • The Upgrade Utility version number is *9.41.00xx*. • The Upgrade Utility build number for this version is *00xx*.
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. **Note:** preupg.bin includes the files in sqlupgrade.unl. To run SQL Compare, you do not need to load sqlupgrade.unl again after you load preupg.bin.
upgrade.inf	Signature information for the upgrade objects.
upgrade-pd.inf	Signature information for the upgrade objects of Process Designer.

List of Upgrade Utility files, continued

File	Contents
upgrade.str	Database dictionaries to be upgraded.
upgrade.ver	Version stamp for this upgrade.
*.dta (in the data or data-pd folder)	The data files for each table that needs to be upgraded. For example, upgradeactivityactions.dta and upgradeactivitytype.dta. **Note:** Service Manager 9.41 Upgrade Utility consists of 274 *.dta files in the data folder, and 276 *.dta files in the data-pd folder.
upgrade.mak	Signature definitions for the upgrade objects.
upgdbdct.dta	Temporary dbdicts needed for the SQL Compare process.
DeltaMigrationTool.unl	Files that allow you to run Delta Migration Tool.
.zip (in the 3waymerge\oob folder)	Each zip file includes the XML representation of the objects that have been signatured in the pre-upgrade out-of-box version for the Three-Way Merge tool. **Note:** Only five base versions are included, namely, SM7.10, SM7.11, SM9.20, SM9.30, and SM9.30 (with Process Designer Content Pack 9.30.3 installed).

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.

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 Interactive Installation Guide*. Additionally, you can obtain most Service Manager knowledge from the Service Manager 9.41 Help Center.
- HP Software Support Online: The HP Software Support Online web site has operating system and compatibility information, product documentation, and release notes. This site requires that you register for an HP Passport and sign-in. To register for an HP Passport ID, go to: http://h20229.www2.hp.com/passport-registration.html.

Step 2: Meet the software requirements

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

- Your RDBMS version, operating system, and client/server environment must meet all criteria listed in the Support Matrix for the target version. See the *HP Service Manager Support Matrices* to review the Support Matrix.
- Your existing Service Manager application release level must be HP Service Manager 9.2x.
- 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 HP 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 for the language to which you will be upgrading. For more information and instructions on how to install the language pack, refer to the *HP Service Manager 9.41 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.

Step 6: Meet database requirements

Before upgrading your system, verify that your system is pointing to the correct database.

- "Convert all tables and fields from lowercase to uppercase" below

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

HP 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.

Chapter 3: Upgrade the server and the client on the production environment

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 9.41 platform (server and client).

- To obtain the latest client, install the client from the HP Service Manager 9.41 release, and follow the instructions in the *Service Manager 9.41 Interactive Installation Guide*.
- To obtain the latest server, install the Service Manager 9.40 server from the Service Manager 9.40 installation DVD, and then install the Service Manager 9.41 server patch. For details, see the *Service Manager 9.41 Interactive Installation Guide*.

Chapter 4: Upgrade tasks on the development environment

Step 1: 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, follow these steps:

1. Identify a server to use for the development and test environments.
    - Linux: 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 15.
    - Upgrade your RDBMS to a version compatible with HP Service Manager 9.41. See the HP Service Manager 9.41 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 9.41 run time environment on the duplicated system. Do not load the

Service Manager 9.41 demonstration data files.

7. Install a Service Manager 9.41 client on the duplicated system.

Step 2: Update Service Manager configuration files

The following tables list the changes that you need to make to the HP 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 by executing the **sm.exe** command in the Service Manager installation directory (for example, C:\Program Files (x86)\HP\Service Manager 9.40\Server\RUN).

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 Center 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 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 6 MB.
shared_ memory:96000000	Replace the default **shared_ memory:32000000** with **shared_memory:96000000**.	This sets the shared memory size to 96 MB. However, if you have a large database, you may need to allocate more shared memory to accommodate the upgrade processing.
heartbeatinterval:120	Add this parameter to the end of the file if it does not exist.	This parameter controls the client heartbeat frequency. If the server does not receive a heartbeat from the client within the time-out limit as defined by the sessiontimeout parameter, the server

sm.ini, continued

Parameter	Changes	Description
		terminates the client. All unsaved data is lost and the client must establish a new connection.

Step 3: 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, follow these steps:

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

Step 4: Load the application upgrade files

You must load the preupg.bin file and the transfer.bin file into HP 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, follow these steps:

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

   Note: 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. Make sure the extraction does not miss any Upgrade Utility files. Otherwise, the upgrade process will fail.
3. Log in to the Service Manager Windows client as a system administrator.

   Note: Select **English** as the language when logging into the system for an upgrade.

4. Click **Window** > **Preferences** > **HP Service Manager**, and clear the **Client side load/unload** check box.

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

5. Load the preupg.bin file using Service Manager Database Manager.
6. Type `smupgrade` in the Service Manager client command box, and then press **Enter**to launch the Upgrade Utility.
7. Click **Load Upgrade Utility File - transfer.bin**.
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\`

   Linux: `/tmp/upgrade/`

9. Wait until the file is loaded and the system displays the `Transfer files loaded` message.

   Note: The loading process may take a long time.

10. Log out of Service Manager and then log on again.

Step 5: Run the SQL compare utility

The SQL compare utility is an informational tool that compares your existing table fields and unique key information with those of the HP Service Manager version you are upgrading to, and then reports the new and modified fields and unique keys that will merge into the existing tables. You can use the list of the fields and unique keys produced by the SQL compare utility to determine whether any fields or unique keys in your current system differ from those in the new version. You can also use the report to determine which new fields and unique keys you must add to RDBMS-mapped files if you choose to make the changes manually during the application upgrade.

After running the SQL compare utility on the table fields, refer to the following table for the result types and the recommended actions.

Result type	Recommended actions
(Null)	The related field is on the id field in the licenseinfo dbdict. No further action is necessary.
Nonexistent in HP Service Manager 9.41 fresh installation	This field is added by tailoring. No further action is necessary.
Not to be modified to other field type by the Upgrade Utility	Manually modify this field type according to the matching solution described in "Data type mismatches" on page 40.
Not to be modified to other field database type by the Upgrade Utility	Manually modify this field type according to the matching solution described in "Data type mismatches" on page 40.
Not to be moved to alias table by the Upgrade Utility	Manually modify this field type according to the matching solution described in "Data type mismatches" on page 40.
To be modified by the Upgrade Utility	This field type modification will be done automatically by the Upgrade Utility during the upgrade process. You can also manually modify this field type before the upgrade to improve the upgrade performance.
To be modified by the Upgrade Utility, and the original field name xxxx will be renamed to xxxx.old	This field type modification will be done automatically by the Upgrade Utility. No further action is necessary.

Result type	Recommended actions
To be added by the Upgrade Utility	This new field will be added automatically by the Upgrade Utility during the upgrade process. You can also manually add this new field manually before the upgrade to improve the upgrade performance.

After running the SQL compare utility on the table unique keys, refer to the following table for the result types and the recommended actions.

Result type	Recommended actions
Nonexistent in HP Service Manager 9.41 fresh installation	This key normally is added by tailoring. Check this key to see whether it includes any field of added primary key or unique key.
Exist in Old OOB, and will be removed by the Upgrade Utility	This key exists in Old OOB only, but does not exist in Service Manager 9.41. The Upgrade Utility will remove this key automatically. No further action is necessary.
To be added by the Upgrade Utility	This key does not exist in Old OOB, but exist in Service Manager 9.41. The Upgrade Utility will add this key automatically. No further action is necessary.
To be ignored by the Upgrade Utility	The field is added as a unique key because Service Manager does not support operations on tables with primary keys when the application version is earlier than 9.32. You can manually change the key type to primary key after the system upgrade.

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.

Running SQL Compare

The following SQL Compare files are included when you install the HP 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, follow these steps:

1. Type `smupgrade` in the Service Manager command line, and then press **Enter**.
2. Click **Run SQL Compare Utility**. A dialog box opens.
3. Type the full 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\`

   Linux: `/tmp/upgrade/`

   Do not include the file name (upgdbdct.dta) in this path.
4. Click the **Run** button.

   SQL Compare returns the following 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 of the table fields, follow these steps:

1. Type `smupgrade` in the Service Manager command line, and then press **Enter**.
2. Click **View SQL Field Compare Results**.
3. Click **Search**. The results are displayed in a record list.

Each table field difference found by the SQL Compare Utility appears as a separate record in the sqlupgradefield 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 sqlupgradefield record provides the following information for each field you must add or modify if you are updating your RDBMS mapped system manually.

List of sqlupgradefield fields

Field	Description
File Name	The exact file name which delegates the database dictionary.
Field Name	The exact field name to add or modify to the associated database dictionary.
Field Structure	The field structure in the database dictionary.
Field Type	The field type in the database dictionary.
Field Level	The field level in the database dictionary.
Result Type	The actions happened or will happen to this field.
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.
Field SQL Table Alias	The SQL table alias of this field in the database dictionary.
Field SQL Name	The SQL name of this field in the database dictionary.
Field SQL Type	The SQL type of this field in the database dictionary.
Field SQL RC	The encoding format of this field in the database dictionary.
Exceptions	The exceptions that occurs when adding or modifying this field.

To view the SQL Compare results of the table unique keys, follow these steps:

1. Type `smupgrade` in the Service Manager command line, and then press **Enter**.
2. Click **View SQL Unique Key Compare Results**.
3. Click **Search**. The results are displayed in a record list.

Each table unique key difference found by the SQL Compare Utility appears as a separate record in the sqlupgradekey file. This record also lists the new unique keys that you must add to the database dictionary, if you want to check the possible duplicated record manually.

The sqlupgradekey record provides the following information for each field you must add or modify, if you want to check the possible duplicated record manually.

List of sqlupgradekey fields

Field	Description
File Name	The exact file name which delegates the database dictionary.
Result Type	The actions happened or will happen to this field.
Key Type	The unique key types include primary, unique, and noduplicated.
Key Field	Concatenates all the key fields with the | character.
Exceptions	The exceptions that occurs when adding or modifying this unique key.

Add new fields

For the new fields to perform correctly, they must exist in both the HP Service Manager database dictionary and the SQL database. If you are updating your RDBMS mapped system manually, you must add them to the SQL database and update the existing Service Manager SQL mapping in the database dictionary. 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	Check the cm3r dbdict. There is a "middle" field of structure type.
The field is an array	If the field is an array, the field name appears twice in the new field list. Check any field of array type. The root field has array type, but the same name field normally has the real type (for example, VARCHAR(60)). 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.

In this instance	Add the field here
The field is part of an array of structures	Check the kmcategory dbdict permission field, which is a structured array field.

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

Process the NVARCHAR SQL type

Although the upgrade version uses the VARCHAR SQL type for fields, SQL Compare and Upgrade Utility retain the use of the NVARCHAR SQL type after the upgrade. For example, if the SQL type of a field is NVARCHAR(60) in the current version and the SQL type of this field in the upgrade version should be VARCHAR(60), the SQL type of this field will remain NVARCHAR(60) in the upgraded version.

If the current and updated versions of a field have different length SQL types, SQL Compare and Upgrade Utility use the longer length. For example, if the field SQL Type is NVARCHAR(60) in the current version and if the SQL Type of this field in the upgrade version is VARCHAR(100), the SQL Type of this field will be NVARCHAR(100) in the upgraded version.

If you want to tailor the SQL type of a field from VARCHAR to NVARCHAR and retain the same length, make sure to perform tailoring from the database instead of from the HP Service Manager database dictionary. Otherwise, the tailoring will be ignored by Service Manager.

Step 6: Run the Upgrade Utility

Now that you have a functional environment, you are ready to run the Upgrade Utility. Follow the steps in this step to run the out-of-box upgrade against the data in your development environment and to run your custom upgrade against your test and production environments. You must perform these steps in a HP 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.

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 the dbdict update phase, where the utility updates dbdicts.
3. The third phase is the data update phase, where the utility updates application data.

To run the HP Service Manager Upgrade Utility, follow these steps:

1. Type `smupgrade` in the Service Manager command line, and then press **Enter** to launch the Upgrade Utility.
2. Click **Configure and Apply the Upgrade**.
3. The system displays a series of information for your verification, including Applications version upgrading from, Applications version upgrading to, Applications base version, Full path to the Upgrade Utility files and Language(s) to be upgraded. Verify these information and then click **Next** to continue.

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

4. The system displays the following message:

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

   You are preparing a custom upgrade on a development system. Leave the selection as **Yes**, and then click **Next**.

5. The system displays the following message:

   Message: Do you want to force the replacement of the objects?

   If you want to replace each Renamed RAD application with its upgrade version, select the **Replace RAD** option. Each Renamed RAD application is replaced with the upgrade version, and a copy of old

RAD application is renamed to PRE<*old version number*><*object name*>. Its upgrade result is marked as "Replaced".

If you do not want to replace Renamed RAD application, do not select the **Replace RAD** option. Each Renamed RAD application is not replaced but still remains in the Renamed list, and the upgrade version of the RAD application is renamed to NEW941<*object name*>. Its upgrade result is kept as **Renamed**.

6. Select or clear the **Replace RAD** check box, and then click **Next**. This check box is selected by default.
7. The upgrade is now ready to start. Click **Next** to start the upgrade.
8. When you are asked whether you want to proceed, click **Yes**.
9. The Upgrade Utility displays the status when the upgrade is being processed.
10. When you receive an "UPGRADE IS COMPLETE" message, the Upgrade Utility has finished the data processing and you can follow the instructions in the message to complete the next steps. After you close the message dialog, you are automatically logged out.
11. Restart the server and log back in to the client.
12. Open the scversion table in the Database Manager, and verify that the Application Version field is **9.41.00xx**. If this field displays a value other than **9.41.00xx**, 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	This file contains specific information about the upgrade, including the following: • All information in upgrade.log. • Name of the file being purged. For example, “2014-03-20 13:58:32 dbdict: upgradestatus is purged.” • Progress of a file loading. For example, “2014-03-20 13:58:39 Adding record # 100

List of upgrade log files, continued

Log file	Contents
	from table upgradeobjects". • Changes made to fields during file processing. For example, "2014-03-20 14:05:24 Increasing field length for incidentlib.company in kmquery dbdict from VARCHAR(40) to VARCHAR(70)" • Signature of a file and the action on it. For example, "2014-03-20 14:19:53 Processing Format Record : cc.get.dependen, signature=(current=3843738292, oob=NONE, upgrade=3843738292), upgraderesult=current"
except.log	This file contains information about any exceptions reported by the upgrade, including the following: • Messages about data type mismatches that failed to be resolved, or database dictionaries failed to be upgraded. For example, "2014-03-20 14:04:29 dbdict:FolderRights, field:delete, field type is logical -- expected to be:character" See "Data type mismatches" on page 40. • Messages about the unique key changes that failed to be resolved. For example, "2014-03-20 14:06:31 dbdict:Todo, Unique Key is {"record.id", "itemType"} -- expected to be:{"record.id"}" If there are exceptions logged in this file, you will have to resolve them in the "Resolving exceptions and conflicts" phase.
upgrade.log	This file contains information about where the upgrade is at any point. This file contains only the main steps of the upgrade, including the following: • Starting and ending of each sub-phase. For example, "2014-03-20 13:58:07 **** Start Phase [Pre Upgrade Action Update] ****" • Main activities during each sub phase. For example, "2014-03-20 13:58:32 Purging upgrade files..." • Number of files to be processed. For example, "2014-03-20 14:04:25 There are 608 dbdicts to be processed." • Names of the files being processed. For example, "2014-03-20 14:04:25 Processing dbdict, AdvFilter"

Step 7: View the 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 Upgrade Results and Merge Conflicts**.

To view the upgrade results, follow these steps:

1. Type `smupgrade` in the HP Service Manager command line, and then press **Enter** to launch the Upgrade Utility.
2. Click **View Upgrade Results and Merge Conflicts**.
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.

Description of upgrade results

The search criteria, search results, and a description of the applicable action for each result are described in the table below.

Field	Definition
Object Type	Enter the type of object you want to search for, or leave this field blank to return all object types. The object types you could search for include Application Cluster, Object, Process, ScriptLibrary, displayoption, format, formatctrl, help, joindefs, link, scmessage, screlconfig, triggers, validity, and wizard.
Object	Enter the name of the object you want to search for, or leave this field blank to return

Field	Definition
Name	objects with any name. The object name is typically the unique identifier in the database table specified for the object type. The unique identifier for some object types (for example, format) may contain concatenate values of multiple fields according to the key setting in the signaturemake signature definition table.
Result: Added	Select this option to search for new objects that the Upgrade Utility added to the system. These objects did not exist in your system before this upgrade. No further action is necessary for these objects.
Result: Already Current	Select this option to search for objects that were already the latest version. No further action is necessary for these objects.
Result: Auto Merged	Select this option to search for objects that the Upgrade Utility automatically merged by using your local version, the out-of-box version and the upgrade version of the objects. **Note:** The result occurs only after applying the first out-of-box upgrade; it no longer occurs after applying the custom upgrade. **Required Action**: If the object was not merged the way you expect, use the **Revert** or **Mass Revert** option from the options menu to revoke the auto-merge and merge the objects manually.
Result: Error	Select this option to search for objects that encountered an error while being updated by the Upgrade Utility. For more information about the error, review the sm.log and except.log files. **Required Action**: Fix the cause of the error, or apply the upgrade again in a copy of your production system if it is needed.
Result: Forced	Select this option to search for objects that were not only tailored on your Service Manager system but also changed in the upgrade version. After the upgrade, your objects were automatically replaced with the objects in your custom upgrade package. The Upgrade Utility copied your object of the original version to its revision. **Note:** This result occurs only after applying the custom upgrade. No further action is necessary for these objects.
Result: Kept Customer	Select this option to search for objects that were tailored on your Service Manager system but not changed on the upgrade version. These objects were not changed.

Field	Definition
	Note: The result occurs only after applying the first out-of-box upgrade; it no longer occurs after applying the custom upgrade. **Tip:** If an object is Application Cluster and you want to use the upgrade version, you can select **Choose Upgrade**. The object <*object name*> is then renamed to PRE<*old version number*><*object name*> and the object NEW941<*object name*> is renamed to <*object name*>. No further action is necessary for these objects.
Result: Kept Customer Non-OOB	Select this option to search for objects that did not exist in the original version but were added on your Service Manager system, and also added on the upgrade version. **Note:** The result occurs only after applying the first out-of-box upgrade; it no longer occurs after applying the custom upgrade. **Note:** If the object is Application Cluster and you want to use the upgrade version, you can select **Choose Upgrade**. The object <object name> is then renamed to PRE<old version number><object name> and the object NEW941<object name> is renamed to <object name>. **Required Action**: Choose one of the following solutions for each object with this result. • Keep the old version — No further action is necessary. • Keep the new version — Select the object in the merge view and click **Copy all from left to right** on the toolbar. • Merge new and current versions — Determine which of the new features need be incorporated into your tailored object, and then modify the tailored object. When finished, delete the new object NEW941<object name> and the copied object PRE<old version number><object name>.
Result: Merged	Select this option to search for dbdict objects that were tailored on your Service Manager system, which Upgrade Utility has merged with the version in this upgrade. **Required Action**: Test these objects, and when satisfied change their result to **Reconciled**.
Result: Previously Reconciled	Select this option to search for objects that were tailored on your Service Manager system, that were marked as Reconciled during a previous upgrade or patch release, or where your object was not changed and the Upgrade Utility added a new object

Field	Definition
	NEW941<object name>. **Note:** The result occurs only after applying the out-of-box upgrade; it no longer occurs after applying the custom upgrade. **Required Action**: Choose one of the following for each object with this result. • Keep the old version — No further action is necessary. • Keep the new version — Select the object in the merge view and click **Copy all from left to right** on the tool bar. • Merge new and old versions — Determine which of the new features should be incorporated into your tailored object, and then make the changes in your tailored object. When finished, delete the new object NEW941<object name> and the copied object PRE<old version number><object name>.
Result: Reconciled	Select this option to search for objects that you have already marked as Reconciled. **Note:** The result occurs only after applying the out-of-box upgrade; it no longer occurs after applying the custom upgrade. No further action is necessary for these objects.
Result: Renamed	Select this option to search for objects that were not only tailored on your Service Manager system but also changed on the upgrade version. After upgrade, your tailored object was not changed, the Upgrade Utility added a new object NEW941<object name> and copied your tailored object as a backed up object PRE<old version number><object name>. **Note:** The result occurs only after applying the out-of-box upgrade; it no longer occurs after applying the custom upgrade. **Required Action**: Choose one of the following for each object with this result. • Keep the old version — No further action is necessary. • Keep the new version — Select the object in the merge view and click **Copy all from left to right** on the tool bar. • Merge new and old versions — Determine which of the new features should be incorporated into your tailored object, and then make the changes in your tailored object. When finished, delete the new object NEW941<object name> and the copied

Field	Definition
	object PRE<old version number><object name>. **Note:** If the object is Application Cluster and you want to use the upgrade version, you can select **Choose Upgrade**. The object <object name> is then renamed to PRE<old version number><object name> and the object NEW941<object name> is renamed to <object name>.
Result: Upgraded	Select this option to search for objects that were automatically replaced with the upgrade version objects. These are objects that were not tailored on your Service Manager system, but changed on the upgrade version. **Note:** The result occurs only after applying the out-of-box upgrade; it no longer occurs after applying the custom upgrade. No further action is necessary for these objects.
Result: Replaced	Select this option to search for RAD Application objects that were not only tailored on your Service Managersystem but also changed on the upgrade version when you select **Replace RAD**. After upgrade, your tailored object was renamed to PRE<old version number><object name> and the Upgrade Utility added a new object <object name>. **Note:** The result occurs only after applying the out-of-box upgrade; it does not occur after applying the custom upgrade. No further action is necessary for these objects. **Tip:** If you do not want to replace the RAD Application, you may select **Revert**. The object <object name> is then renamed to NEW941<object name> and the object PRE<old version number><object name> is renamed to <object name>. The result is then set back to **Renamed**, **Kept Customer** or **Kept Customer Non-OOB**.

See the following table for some examples about how the upgrade result types are marked for an existing record.

OOB record (9.30)	OOB record (9.33)	Customer record (9.33)	OOB record (9.4x)	Upgrade Utility check sequence	Upgrade Utility check condition	Conflict flag	Upgrade result type	Upgrade result value
111	112	112	112	1	Customer record	No	Already Current	112

OOB record (9.30)	OOB record (9.33)	Customer record (9.33)	OOB record (9.4x)	Upgrade Utility check sequence	Upgrade Utility check condition	Conflict flag	Upgrade result type	Upgrade result value
					(9.33) = OOB record (9.4x)			
111	112	112	113	2	Customer record (9.33) != OOB record(9.4x) and Customer record (9.33) = OOB record (9.33)	No	Upgraded	113
111	112	113	112	3	Customer record (9.33) != OOB record (9.4x) and Customer record (9.33) != OOB record (9.33) and OOB record (9.33) = OOB record (9.4x)	No	Kept Customer	113
111	112	113	114	4	Customer record (9.33) != OOB record (9.4x) and Customer record (9.33) != OOB record (9.4x) and OOB record (9.33) != OOB record (9.4x)	Yes	Renamed	113

Step 8: Manage the upgrade result data

To manage the upgrade result data more easily, do either of the following:

- 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 HP Service Manager Help Center documentation for more information.)

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, HP Service Manager features may not function as expected.

Step 9: Resolve exceptions

Exceptions are logged if the Upgrade Utility cannot add or 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. Follow these steps to modify field data type by using the Dbdict Utility:

1. Navigate to **System Navigator** > **Tailoring** > **Database Dictionary**.
2. Type the dbdict name in the File Name field, and then click **Search**.
3. Double-click the field you want to change the data type, and then make updates as expected.
4. Save your changes.

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 displays a list of data type mismatches that may appear in the except.log file for an Oracle database. Only some examples are listed for each category of data type mismatch. If you are using an MSSQL or a DB2 database, the actual error message may vary slightly. For example, the following list highlights 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:incidents, field:svc.options, SQL type is VARCHAR2(90) -- expected to be:BLOB dbdict:SMISTaskQueue, field:failedReason, SQL type is VARCHAR2(60) -- 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	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.

Error message in Oracle:	Solution:
dbdict:eventout, field:evnumber, field type is number -- expected to be:character	
dbdict:svcCatLanguage, field:catalogid, field type is character -- expected to be:number	This field resides in the catalog structure field of BLOB SQL type. To fix this issue, change the field type to number 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 44 section.
dbdict:cm3r, field:assets, SQL type is CLOB -- expected to be: VARCHAR2(200) dbdict:ocmgroups, field:approvers, SQL type is CLOB -- expected to be: VARCHAR2(80) dbdict:operator, field:cap.exec, SQL type is CLOB -- expected to be: VARCHAR2(50) dbdict:operator, field:profile.change, type:character, SQL type is CLOB -- expected to be: VARCHAR2(60) dbdict:operator, field:profile.request, SQL type is CLOB -- expected to be: VARCHAR2(60) dbdict:svcCatalog, field:access.list, SQL type is CLOB -- expected to be: VARCHAR2(50)	This kind of error message is generated because the Upgrade Utility does not re-map the field of array type from the main table to the alias table and the subsequent full-table copy may slow down the upgrade. Follow these steps to fix the issue: 1. Double-click the field of array type which contains this field in the related table, and then update the value in “SQL Table” to a<*number*> which is not used in the alias table. For example, a3. 2. Double-click the field of character type in the related table. Update the value in “SQL TYPE” to the expected value, and then update the value in “SQL Table” to the same a<*number*> which is not used in the alias table. For example, a3. 3. Click **OK** to save your changes.

Error message in Oracle:	Solution:
	Caution: If the related table contains more than 100,000 records, fixing the data type mismatches in the field may take more than one hour.
dbdict:inbox, Primary Key is inbox.id -- the primay key can not be added. Since SM doesn't support operation on table with primary key when application version is older than 9.32, then add inbox.id as unique key.	To fix this issue, change the unique key to primary key by using the Dbdict utility.

Note: Ignore the message if an array field has been moved to the alias table. For example:

```
dbdict:operator, field:names, SQL type is VARCHAR2(60) -- expected to be:CLOB
dbdict:operator, field:values, SQL type is VARCHAR2(60) -- expected to be:CLOB
dbdict:operator, field:groups, SQL type is VARCHAR2(60) -- expected to be:CLOB
dbdict:operator, field:month.ext, SQL type is VARCHAR2(60) -- expected to
be:CLOB
dbdict:operator, field:month.abv, SQL type is VARCHAR2(60) -- expected to
be:CLOB
dbdict:operator, field:security.group, SQL type is VARCHAR2(60) -- expected to
be:CLOB
dbdict:operator, field:assignment.groups, SQL type is VARCHAR2(60) -- expected
to be:CLOB
dbdict:operator, field:named.modules, SQL type is VARCHAR2(30) -- expected to
be:CLOB
dbdict:operator, field:ess.initial.app.names, SQL type is VARCHAR2(30) --
expected to be:CLOB
dbdict:operator, field:ess.initial.app.values, SQL type is VARCHAR2(30) --
expected to be:CLOB
dbdict:operator, field:misc.array, SQL type is VARCHAR2(30) -- expected to
be:CLOB
dbdict:operator, field:kmgroup, SQL type is VARCHAR2(30) -- expected to be:CLOB
```

Note: Ignore the message if the child fields of an array of structure field have been mapped to the database columns. For example:

```
dbdict:computer, field:video, SQL type is EMPTY -- expected to be:BLOB
dbdict:computer, field:vid.card.id, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
dbdict:computer, field:vid.card.bios, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
dbdict:computer, field:vid.manufacturer, SQL type is VARCHAR2(30) -- expected to
```

```
be:EMPTY
dbdict:computer, field:vid.model, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
dbdict:computer, field:vid.chip.type, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
dbdict:computer, field:vid.memory, SQL type is NUMBER -- expected to be:EMPTY
dbdict:computer, field:vid.resolution, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
dbdict:computer, field:vid.resolution.x, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
dbdict:computer, field:vid.resolution.y, SQL type is VARCHAR2(30) -- expected to
be:EMPTY
```

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, follow these steps to remove the delete.tmp field:
    a. In the Database Manager, search for all records in the dbdict table.
    b. Select the dbdict format.
    c. Type `FolderRights` in the **Name** field, and then click **Search**.
    d. Locate the row for the delete.tmp field. Remove the values in the Name, Type, Index, and Level fields respectively.
    e. Click **Save** and **OK**.
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."

Add new object record failure

All the upgraded objects are saved in the upgradeobjects table, and you need to add some objects from this table to the current upgradeobjects table. If the target table contains more than one unique type key, some errors may occur when adding new objects during the upgrade process. Refer to the following error message as an example.

```
2014-03-20 14:04:29 file:inbox, updated the field inbox.id value from 10000313 to
14042954310000313
```

The value of the inbox.id field is 10000313 in this example, which is used in the current upgradeobjects table. If the Upgrade Utility adds an inbox record that contains the same value for this field, the current value of the inbox.id field is automatically prefixed with a timestamp to avoid duplication errors. The timestamp is composed of hour (two bits), minute (two bits), second (two bits) and millisecond (three bits). In this example, the timestamp is 140429543.

After upgrading, you can update the temporary value of the inbox.id field as necessary.

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.

Follow these steps 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. Continue to follow *steps 1* through *step 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.

Follow these steps 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. Continue to follow *steps 1* through *step 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.

Follow these steps 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. Continue to follow *steps 1* through *step 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"}`

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 10: Resolve conflicts

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

Standard conflict resolution process

The standard process of resolving conflicts are listed as follows:

1. To view the conflicts, click **View Upgrade Results and Merge Conflicts** 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 NEW941 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 NEW941 prefix, or you can select **Choose Upgrade** to replace the object with the new object that is prefixed with NEW941.

     The Mass Choose Upgrade feature can help you replace multiple customized objects with new objects that are prefixed with NEW941. For more information, see the "Using the Mass Choose Upgrade feature" on page 61.
   - **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 NEW941 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 54 and "Using the Auto Merge and Revert options" on page 61 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 62.

For more specific guidelines and examples for conflict resolution, see white paper *Conflict Resolution for Upgrade to Service Manager (SM) 9.3x*.

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, **NEW941<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: NEW941apm.edit.problem_do_nothing_3 Result: This is the new SM 9.41 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 NEW941apm.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 NEW941apm.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 9.41 (for example "7.1"), are already current with the ones in Service Manager 9.41. Therefore, you should not change the Current Release Level field to SM 9.41. There are two cases in which a RAD application's Current Release Level may be marked as SM 9.41:

1. The RAD applications that are new in Service Manager 9.41.
2. The RAD applications whose code has been changed in Service Manager 9.41.

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 48, 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 NEW941 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, you may select "Choose Upgrade" to use the new object, or you may manually delete your customized RAD application and rename the new RAD application by removing the NEW941 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 NEW941 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.

Scenario	Action
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, you can use "Revert" or "Choose Upgrade" to resolve RAD application conflicts. Alternatively, if you want to resolve the conflicts manually, perform two operations: deleting a RAD application, and renaming a RAD application. To delete a RAD application, follow these steps: 1. Open the RAD application in the RAD Editor. 2. Click **Delete**. 3. Click **Delete All**. To rename a RAD application, follow these steps: 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:

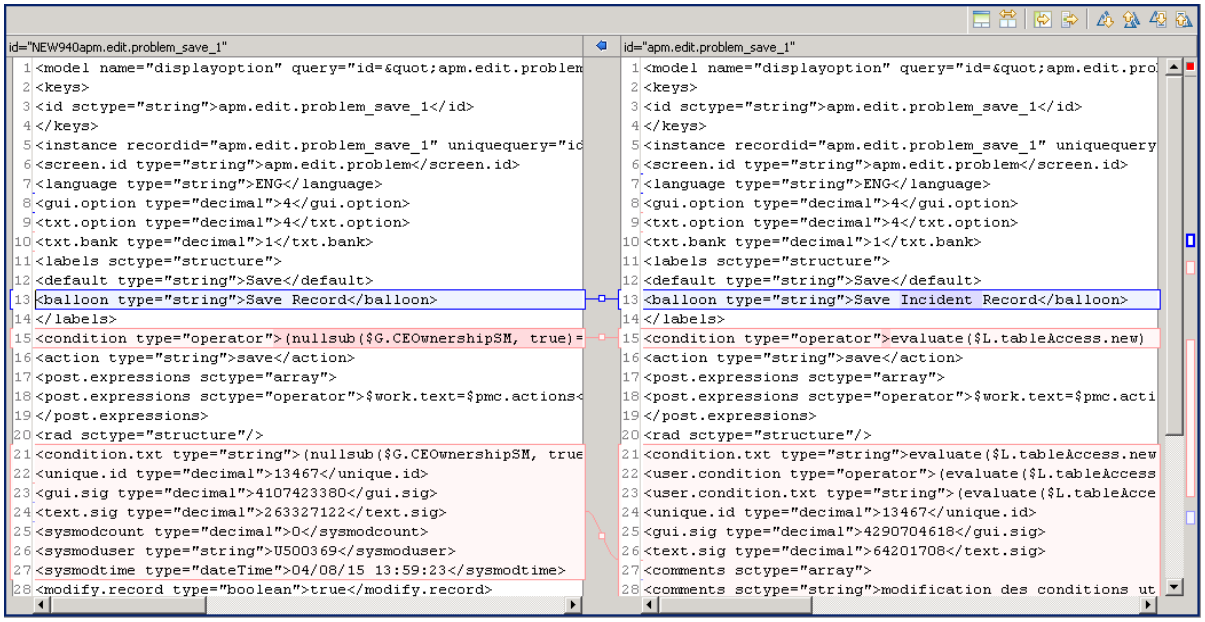
- It allows you to identify where changes are located before you can visually compare the objects and to make changes manually, such as in format records.
- It allows you to identify and merge changes directly between objects, such as ScriptLibrary records.

To use the Merge tool, follow these steps:

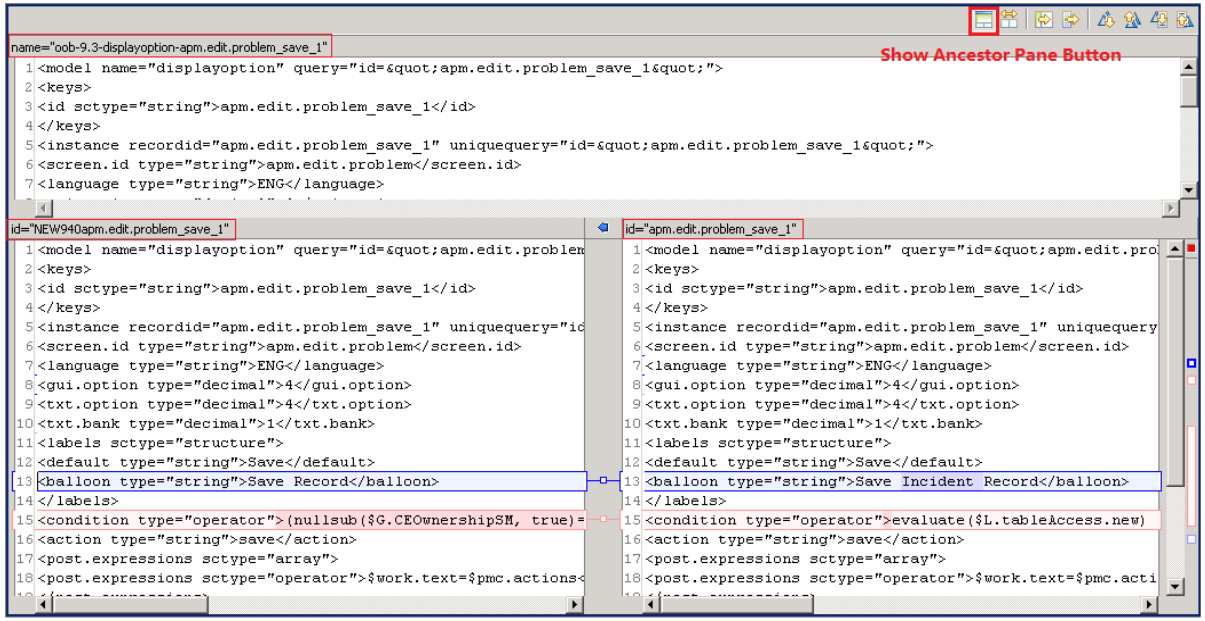
1. In the **UPGRADE UTILITY** section, click **View Upgrade Results and Merge Conflicts**.
2. In the **Result** drop-down list, select **Renamed**.

   Note: The Two-way/Three-way Merge tool is available only for “Renamed” records.

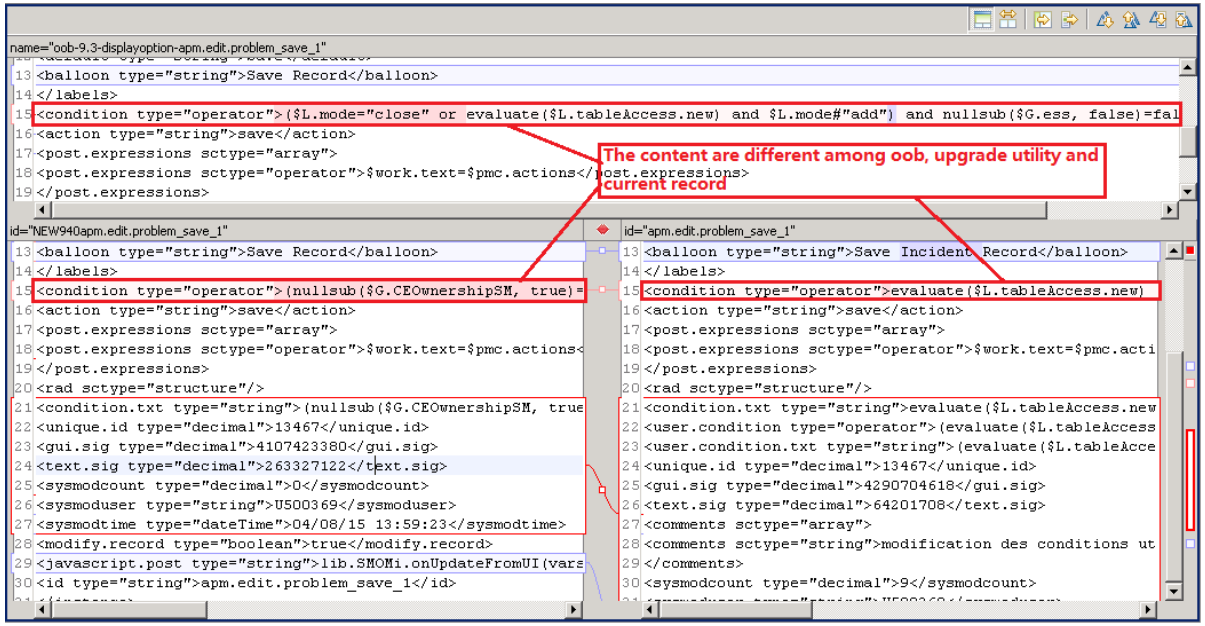
3. Click **Search**.
4. A list is returned that displays all the result records of the “Renamed” type. Select a record that you want to examine, and then click **Merge** on the toolbar.
5. The current default merge mode is Three-Way Merge mode. The merge option is not available for format records. Instead, 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.

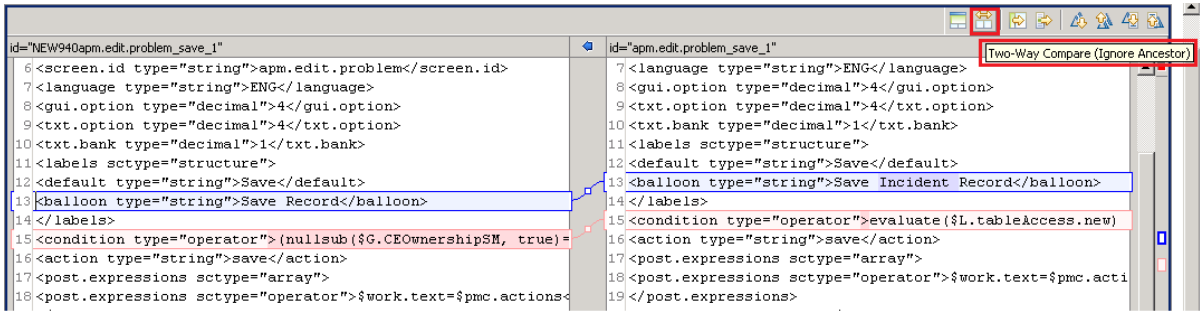
d. In Three-Way Merge mode, if the records on the left contain red background records, and you then click the **Copy All from Left to Right** button, the records in the right pane with blue background will be replaced with selected records from the left pane.

> **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.

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 HP 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 click the **Revert** button on the toolbar to restore "Auto Merged" or "Renamed" objects, or "Replaced" Application Cluster 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.

Note: If you do not choose to use the auto-merge option, you must manually unzip the OOB data to the same folder in which you extracted the Merge Tool. If you do choose the auto-merge option, the OOB data is extracted automatically by the Merge Tool.

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

You can use this feature to quickly update Application Cluster objects of the following statuses:

- Kept Customer
- Kept Customer Non-OOB
- Renamed

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

1. In the **UPGRADE UTILITY** section, click **View Upgrade Results and Merge Conflicts**.
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 Upgrade Results and Merge Conflicts**.
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, use the **Mark as Reconciled** button on the toolbar instead.

Step 11: 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.

Clean up artifacts

The Upgrade utility does not automatically clean up artifacts that were left over by the upgrade, such as objects that were prefixed with PRE<version_number> or NEW941, which are copied and renamed from pre-upgrade objects. These objects must be deleted from the system. Otherwise, the system may not work as expected. To delete those objects automatically, the purge tool provides a function to clean up artifacts that were left over by the upgrade. To run the purge tool, follow these steps:

1. Type `*aapm.upgrade.purge` in the HP Service Manager command line field and then press **Enter**.
2. Click **I'm done, and I want to remove the duplicate data prefixed by "NEW", "PRE" and "OLD"**.
3. Click **OK**.

Or, you may manually delete those objects. 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.

Adopt the logical name solution

In previous versions of Service Manager, the logical.name field in the device table was used as both the unique identifier and display name for a CI, causing loss of CI data integrity across Service Manager modules and complexity of CI reconciliation between integration products (for example, Service Manager does not allow duplicate CI names while UCMDB does).

Service Manager 9.41 solves the problems described above with the logical name solution. Once you have upgraded to Service Manager 9.41, you need to perform additional manual tasks to adopt the solution in your production environment. For a summary of this solution and details of these manual post-upgrade tasks, see the *Service Manager Logical Name Solution* white paper.

Adopt Smart Search

Service Manager 9.41 introduces Smart Search which enables you to search among a variety of content, including Service Manager records, SharePoint documents, static web pages, and KM documents. You can integrate multiple knowledge libraries by configuring different search connectors, so that users can search among the information that they can access.

Once you have upgraded to Service Manager 9.41, follow these steps to make sure the login.DEFAULT format control record has the expected code for the Smart Search feature:

1. Search for the login.DEFAULT format control record.
2. Click **JavaScript**.
3. Check whether the following codes exist:

   ```
   var idolserver = new SCFile('idolserverinfo');
   var rc = idolserver.doSelect('true');
   if (rc === RC_SUCCESS) {
   vars['$lo.idol.enabled'] = idolserver['enable'];
   vars['$lo.idol.img.enabled'] = idolserver['image.enable'];
   if (vars['$lo.idol.enabled']) {
   vars['$G.kmsearchengine'] = 'IDOL';
   }
   }
   ```

   If these codes do not exist, you must type `true` in the Add field and then add these codes manually.

Migrate SLA data

Follow these steps to execute the migration script for SLA:

1. Log on to Service Manager as a system administrator.
2. Type `db` in the Service Manager command line to open Database Manager.
3. Type `migrationSetting` in the **Table** field, and then click **Search** to open the Data Migration Tool form.
4. Click **Search** to find existing migration scripts.

5. Click **Legacy SLA to new Agreement**.
6. On the Migration Script setting detail form, click **Migrate Data** to run the migration script.

Caution: If you have large amounts of SLA data, your Service Manager system may experience performance issue during the SLA data migration process. See QCCR1E133978 and QCCR1E134320 for workarounds.

Step 12: Return the system to normal operation

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

Note: All upgrade-related files should be removed from the system after successful completion of the upgrade. For information on removing these files, see "Step 11: Perform additional manual tasks" on page 63.

To return to a normal operating environment, follow these steps:

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: HP 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 13: Test the development environment (functional testing)

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

Step 14: 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.

Step 15: 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, follow these steps:

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 1: Duplicate the production environment" on page 20.
3. Create a folder for the custom upgrade on the HP 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 command line, and then press **Enter**to launch the Upgrade Utility.
5. Click **Create a Custom Upgrade**.
6. The Upgrade Utility automatically checks for duplicate records prefixed with "PRE", "NEW" or "OLD".

   If no such data is found, you are directed to *step 9* below. If such duplicate data is found, you are directed to *step 7*.
7. Click **Next**. The Upgrade Utility asks you to confirm the purge operation.
8. Select **Yes** to purge the duplicate data.

   If you select **No**, you will need to manually perform the purge operation as described in "Step 3: Perform additional manual tasks" on page 73.
9. The system displays the patch file name and all the supported languages that are installed on your Service Manager system. Review these information, and then click **Next**.
10. The system displays the following message:

    Message: What is the name of this release?

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

    Example: SM941.
11. The system displays the following message:

    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*.

12. The system displays the following message:

    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. The system displays the following message:

    Message: Filter out the objects which are not changed?

    The unchanged objects include the “Already Current”, the “Kept Customer”, and the “Kept Customer Non-OOB” objects in the upgraderesults table, and other objects which are added but not by the Upgrade Utility. Filter out the unchanged objects will speed up the deployment process of the custom upgrade package.

    Select the **Filter out the objects which are not changed?** check box to filter out the records which do not exist in upgrade result or the results are "Already Current", "Kept Customer" or "Kept Customer Non-OOB" from the custom upgrade, and then click **Next**.

    Otherwise, click **Next** directly and all records defined in patches table (name=SM94) are built in custom upgrade.

14. The system displays the following message:

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

    Click **Yes** to continue.

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

Upgrade Utility logs and error messages

When building the custom upgrade, the Upgrade Utility creates a set of log files, which reside in the same directory as the upgrade files. The following table describes the contents of the log files during this step.

List of upgrade logs

Log file	Contents
detail.log	This file contains specific information about the upgrade, including the following: • All information in transfer.log and upgrade.log • Name of the file being purged. For example, “2014-03-20 13:58:32 dbdict: upgradestatus is purged.” • Progress of signature creation. For example, “2014-03-20 14:47:36 Created 100 signatures for Object on version SM940” • Building distribution information. For example, “2014-03-20 14:55:07 Building Distribution object for Application Cluster Action.run”
transfer.log	This file contains information about the object being transferred by the upgrade. For example, “2014-03-20 15:12:27 Initiating an export of scmessage on query " ((class="error" and message.id isin {"10"})) and syslanguage~="xxx""”
upgrade.log	This file contains information about where the upgrade is at any point, including the following: • Starting and ending of each sub-phase. For example, “2014-03-20 14:47:32 **** Start Phase [Pre Create Action Check] ****” • Main activities during each sub-phase. For example, “2014-03-20 14:47:35 Signaturing records.” • Name of the file being exported. For example, “2014-03-20 15:11:37 Exporting: table = Object query = 'true'”

Chapter 5: Upgrade tasks on the test environment

Step 1: 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, follow these steps:

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

Step 2: Apply the custom upgrade to the test environment

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, follow these steps:

1. On the test system, which is a copy of your production system, complete all the preparation tasks in "Upgrade tasks on the development environment" on page 20.
2. Load the preupg.bin file using Service Manager Database Manager.

Note: Instead of using the file that you extracted from the product installation package, you must use the preupg.bin file in your *CustomUpgrade* folder.

Caution: 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 command line, and then press **Enter** to launch the Upgrade Utility.
4. Click **Load Upgrade Utility File - transfer.bin** to load transfer.bin.

   Note: Instead of using the file that you extracted from the product installation package, you must use the transfer.bin file in your *CustomUpgrade* folder.

5. Click **Configure and Apply the Upgrade**.
6. The system displays a series of information for your verification, including Applications version upgrading from, Applications version upgrading to, Applications base version, Full path to the Upgrade Utility files and Language(s) to be upgraded. Verify these information and then click **Next** to continue.

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

7. The system displays the following message:

   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**.

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

processing and you can follow the instructions in the message to complete the next steps. After you close the message dialog, you are automatically logged out.

12. Restart the server and log back in to the client.

13. Open the **scversion** table in the Database Manager, and verify that the **Application Version** field is **9.41.00xx**. If this field displays a value other than **9.41.00xx**, 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.

Upgrade Utility logs and error messages

When applying the custom upgrade to the test system, the Upgrade Utility creates a set of log files, which reside in the same directory as the custom upgrade files.

The contents of these log files are similar to those in the log files when running an out-of-box upgrade. See "Upgrade Utility logs and error messages" on page 32.

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. Only changed records are packaged if you selected the **Filter out the objects which are not changed?** check box. 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 the following scenarios:

- If an object is in the patches record, and the Result field of its related “upgraderesults" record displays “Already Current”, "Kept Customer” or “Kept Customer Non-OOB”, change the Result field to “Reconciled”.

  For example, follow these steps after you have modified the cm.open.display_newphase displayoption record:

  a. Type `smupgrade` in the HP Service Manager command line, and then press **Enter** to launch the Upgrade Utility.

b. Click **View Upgrade Results and Merge Conflicts**.

c. Type `displayoption` in the Object Type field, type `cm.open.display_newphase` in the Object Name field, and then click **Search**.

> **Note:** Some object names consist of multiple key fields, you can find the definitions in the signaturemake record.

d. If the Result field in the search results is “Already Current”, “Kept Customer”, “Kept Customer Non-OOB”, manually change the Result field to “Reconciled”

- If an object is in the patches record, and the Result field of its related “upgraderesults" record is not “Already Current”, "Kept Customer” or “Kept Customer Non-OOB”, no additional task is needed.
- If an object is not in the patches record, do one of the following:
  - Create an unload file containing those objects by adding them to an unload script or using the standard HP 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.

Adopt the logical name solution

In previous versions of Service Manager, the logical.name field in the device table was used as both the unique identifier and display name for a CI, causing loss of CI data integrity across Service Manager modules and complexity of CI reconciliation between integration products (for example, Service Manager does not allow duplicate CI names while UCMDB does).

Service Manager 9.41 solves the problems described above with the logical name solution. Once you have upgraded to Service Manager 9.41, you need to perform additional manual tasks to adopt the solution in your production environment. For a summary of this solution and details of these manual post-upgrade tasks, see the *Service Manager Logical Name Solution* white paper.

Adopt Smart Search

Service Manager 9.41 introduces Smart Search which enables you to search among a variety of content, including Service Manager records, SharePoint documents, static web pages, and KM documents. You can integrate multiple knowledge libraries by configuring different search connectors, so that users can search among the information that they can access.

Once you have upgraded to Service Manager 9.41, follow these steps to make sure the login.DEFAULT format control record has the expected code for the Smart Search feature:

1. Search for the login.DEFAULT format control record.
2. Click **JavaScript**.
3. Check whether the following codes exist:

```
var idolserver = new SCFile('idolserverinfo');
var rc = idolserver.doSelect('true');
if (rc === RC_SUCCESS) {
               vars['$lo.idol.enabled'] = idolserver['enable'];
               vars['$lo.idol.img.enabled'] = idolserver['image.enable'];
               if (vars['$lo.idol.enabled']) {
                              vars['$G.kmsearchengine'] = 'IDOL';
                }
}
```

   If these codes do not exist, you must type `true` in the Add field and then add these codes manually.

Migrate SLA data

Follow these steps to execute the migration script for SLA:

1. Log on to Service Manager as a system administrator.
2. Type `db` in the Service Manager command line to open Database Manager.
3. Type `migrationSetting` in the **Table** field, and then click **Search** to open the Data Migration Tool form.
4. Click **Search** to find existing migration scripts.

5. Click **Legacy SLA to new Agreement**.
6. On the Migration Script setting detail form, click **Migrate Data** to run the migration script.

Caution: If you have large amounts of SLA data, your Service Manager system may experience performance issue during the SLA data migration process. See QCCR1E133978 and QCCR1E134320 for workarounds.

Step 4: Test the test environment

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 HP 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 test environment, follow these steps:

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 9.41 Help Center.
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.

Chapter 6: Upgrade tasks on the production environment

Step 1: Apply the custom upgrade to the production environment

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.

Note: Do not apply an upgrade to your production system if it has not been thoroughly tested. If issues were found in the custom upgrade, HP recommends you to fix the issues in the development system and re-create the customer upgrade.

To apply the custom upgrade to the production system, follow these steps:

1. Make sure the production system is not be available to users while you are applying the custom upgrade.

    a. Have all users log out of the server.

    b. Prevent users from logging into your Service Manager Server by running the **sm -quiesce:1** quiesce command from the operating system's command prompt.

2. Ensure the upgrade files you created are accessible to the production system (the files are located on the same server).

3. If you transfer the files to your production system by FTP, set FTP to binary mode.

4. Apply the customer upgrade to the production system.

    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.

5. Log out from your Service Manager server, and then log in again.
6. Allow users to log in to the server by running the **sm -quiesce:0** quiesce command from the operating system's command prompt.

Step 2: 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.

Adopt the logical name solution

In previous versions of Service Manager, the logical.name field in the device table was used as both the unique identifier and display name for a CI, causing loss of CI data integrity across Service Manager modules and complexity of CI reconciliation between integration products (for example, Service Manager does not allow duplicate CI names while UCMDB does).

Service Manager 9.41 solves the problems described above with the logical name solution. Once you have upgraded to Service Manager 9.41, you need to perform additional manual tasks to adopt the solution in your production environment. For a summary of this solution and details of these manual post-upgrade tasks, see the *Service Manager Logical Name Solution* white paper.

Adopt Smart Search

Service Manager 9.41 introduces Smart Search which enables you to search among a variety of content, including Service Manager records, SharePoint documents, static web pages, and KM documents. You can integrate multiple knowledge libraries by configuring different search connectors, so that users can search among the information that they can access.

Once you have upgraded to Service Manager 9.41, follow these steps to make sure the login.DEFAULT format control record has the expected code for the Smart Search feature:

1. Search for the login.DEFAULT format control record.
2. Click **JavaScript**.
3. Check whether the following codes exist:

```
var idolserver = new SCFile('idolserverinfo');
var rc = idolserver.doSelect('true');
if (rc === RC_SUCCESS) {
               vars['$lo.idol.enabled'] = idolserver['enable'];
               vars['$lo.idol.img.enabled'] = idolserver['image.enable'];
               if (vars['$lo.idol.enabled']) {
                              vars['$G.kmsearchengine'] = 'IDOL';
                }
}
```

If these codes do not exist, you must type `true` in the Add field and then add these codes manually.

Migrate SLA data

Follow these steps to execute the migration script for SLA:

1. Log on to Service Manager as a system administrator.
2. Type `db` in the Service Manager command line to open Database Manager.
3. Type `migrationSetting` in the **Table** field, and then click **Search** to open the Data Migration Tool form.
4. Click **Search** to find existing migration scripts.
5. Click **Legacy SLA to new Agreement**.
6. On the Migration Script setting detail form, click **Migrate Data** to run the migration script.

Caution: If you have large amounts of SLA data, your Service Manager system may experience performance issue during the SLA data migration process. See QCCR1E133978 and QCCR1E134320 for workarounds.

Step 3: Test the production environment

Test the upgraded environment and verify that it functions properly. If there are problems that you cannot resolve, contact HP Software Support.

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 the HP 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 HP 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 `sessiontimeout:1200` and `heartbeatinterval:120` to the sm.ini file, and then restart the upgrade. See also "Step 2: Update Service Manager configuration files" on page 21.

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 HP 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 2: Update Service Manager configuration files" on page 21), and then 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: Integrations do not work after an application upgrade

Symptoms

After you upgrade you applications to HP Service Manager 9.41, certain existing integrations do not work correctly.

Resolution

Certain integration fields that were optional in the old application version have been made required in the new Service Manager9.41 applications. Existing integrations that do not have those fields populated will become invalid. Therefore, you must remove and re-create these integrations in the Integration Manager. These integrations include the following:

- Release Control integration
- BSM OMi integration

For more information about your specific integration, see the Integrations section in the Service Manager Help Center.

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, follow these steps:

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 **SM9.20** for the **Base Version** previously, extract the **SM9.20** folder from inside **SM9.20.zip**, and then place the extracted folder (**SM9.20**) 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.

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

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

C

custom upgrade
  development system 16
  production 76
  test system 16, 70

D

database manager
  upgrade tool 15
DeltaMigrationTool.unl 14
detail.log file 32
displayscreen 50

E

except.log file 33, 40

P

preupg.bin file 70

R

resolving conflicts
  data 51
  display 49

S

sqlupgrade.unl 13

T

transfer.bin file 13, 71

U

upgdbdct.dta 14
upgrade-pd.inf 13
upgrade.inf 13
upgrade.log 33
upgrade.mak 14
upgrade.str 14
upgrade.ver 14

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Applications Upgrade Guide (from HP Service Manager 9.2x) (Service Manager 9.41)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to ovdoc-ITSM@hp.com.

We appreciate your feedback!


Java™
COMPATIBLE