

HP Service Manager

Software Version: 9.41

For the supported Windows® and UNIX® operating systems

Tailoring help topics for printing

Document Release Date: September 2015
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

© 1994-2015 Hewlett-Packard Development Company, L.P.

Trademark Notices

Adobe® is a trademark of Adobe Systems Incorporated.

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

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

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>.

About this PDF Version of Online Help

This document is a PDF version of the online help. This PDF file is provided so you can easily print multiple topics from the help information or read the online help in PDF format. Because this content was originally created to be viewed as online help in a web browser, some topics may not be formatted properly. Some interactive topics may not

Tailoring help topics for printing

be present in this PDF version. Those topics can be successfully printed from within the online help.

Contents

Tailoring	18
Tailoring Best Practices Guide	19
Environment configuration	20
Applying customizations to the production environment	20
Comparison of the tools to use for promoting customizations	21
Development auditing	22
Differential Upgrade utility	23
Unload script utility	24
Promote customizations from development to production	25
Best practices for promoting customizations	26
Development auditing	28
Access the development auditing utility	29
Turn the auditing on/off	30
View the audit history	31
Unload an audit delta	32
Purge audit records	33
Differential Upgrade utility	34
Patch records	34
Signature records	35
Checklist: Differential upgrade steps	35
Create or update a Patch record	36
Create Signature records on the production system	37
Move Signature records to the development system	38
Create the differential upgrade from the development system	38
Customize the Differential upgrade unload file	39
Create a differential upgrade unload from an internal file	40
Load the Differential upgrade into the production system	41
HP Service Manager Document Engine Guide	41
Revision control	41
Creating revisions	42
Create a baseline revision	42
Create a single revision	43
Revert to a previous revision	43

Search for revisions	44
Purge revisions	44
HP Service Manager Wizards Guide	45
HP Service Manager Programming Guide	45
Data management	46
Fields and Keys tab	46
Add a field to a table	55
Edit a field in a table	56
Edit a record key	58
Update the size of a field	59
Cascade Updates	59
Cascade Update example	59
Create a Cascade Delete	60
Create a Cascade Update	60
Create a Cascade Update/Delete Configuration Record	61
Create the example Cascade Configuration Record	61
Create the example trigger record	62
Test the example Cascade Update	62
Data Policy	64
Access Data Policy	64
Data Policy expressions	65
Data Policy and the object record	65
Creating revisions	66
Create a Data Policy revision	66
Example: Create and manage a revision	66
Revert to a previous revision	67
Purge revisions	68
Data Policy and encryption	68
Change the encryption key value	69
Change the columns in a record list	69
Create an alias for your custom CI display name field	70
Define CI Auto Complete Table Columns	71
Specify a referenced table for a field	72
Specify a display field for a referenced table	73
Data validation	74
Accessing the validity file	75
Deleting validity record components	76

Invalid application names	76
Invalid files and form names	77
Invalid field names	77
Printing validity definitions	77
Invoking validity table processing	78
Look-up processing	78
Format Control processes	78
Displayoptions	80
Display value summary details	80
Display range summary details	81
Create validity table definitions	82
Add field-level definitions	82
Add value definitions	83
Add range definitions	84
Validate the validity definitions	85
Delete an entire validity record	85
Delete a single value or range definition	86
Delete value or range definitions from a table	87
Print a detailed report of an entire validity record	87
Validate fields during record processing	88
Create a validity lookup option in Format Control	89
Call validate.fields from Format Control	91
Validity validation rules	92
Global lists	93
Lister	93
Global initer considerations	94
Returning a list to your client	94
Configuring lists with Format Control	94
Access a list record	94
Bind a list to a form	95
Build lists on startup	96
Configure lists with Format Control	96
Determine if a scheduler record exists	98
Determine if the regen cycle is realistic	98
Determine when a list was last regenerated	99
Move a Global List to a client	99
Regenerate all lists	99
Regenerate obsolete lists	100
Start the server side component of the global initer	101

See a complete list of all the global lists on my system	101
Verify lister status and configuration	102
View changes to a list	102
Record tag localization utility	102
Localization process	103
Enable a table for tag localization	103
Access and localize message records	104
Create localized global lists	104
Invoking the localization utility	105
Codes	106
Defining a set of values for a codes-based global list	107
Accessing and localizing message records for a codes-based global list	107
Creating localized codes-based global lists	107
Associate a field with a codes-based global list	108
Message maintenance with record tag localization	109
Update message records for activated or deactivated languages	109
Help systems	110
Online Help system	110
Field help editor	110
Creating field help	111
Preparing to create field help	111
Show context-sensitive help debug information	112
Activate the command line	113
Set up the columns displaying the help record list	113
Turn on the Administration perspective	114
Access the help table	114
Access field help from the System Navigator	114
Determine the fields that a form contains	115
Determine the fields that a table contains	116
Determine whether help exists for a field	117
How does Service Manager determine which help record to display?	117
Add or edit help records	118
Review field help records	119
Sequential number setup	119
Sequential number file	120
Access the sequential number file	120
Create a simple number counter	120
Use decrement in sequential numbers	121
Use prefix and suffix in sequential numbers	121

Update a sequential number record	122
Delete a sequential number record	122
Stored queries	122
Stored Query Maintenance utility	123
Using stored queries in display objects	123
Using menu buttons to run stored queries	123
Using stored queries to produce charts and marquees	124
Using stored queries in scripts	124
Menu option searches	124
Define which system processes manage message traffic	125
Grant access to stored queries	125
Add a stored query	125
Create stored queries from the Query Maintenance form	127
Run a stored query	128
Update a stored query	129
Form creation	132
Forms Designer	132
Access Forms Designer	132
Creating and editing forms	133
Create a form using the Form Wizard	133
Update a form	134
Using the drawing canvas	135
Forms Designer properties view	136
Setting properties	137
Forms Designer controls and tools	137
Attachments control	140
Button control	142
Calendar control	144
Chart control	146
Checkbox control	147
Combo Box control	149
Comfill control	153
Convert Form Layout tool	159
Date control	160
Decimal control	163
Dynamic Form control	165
Embedded Viewer control	166
File control	167

Frame control	169
Graph control	170
Grid view tool	171
Group control	171
HTML Editor control	174
HTML Viewer control	176
Image control	178
Label control	179
Link label control	181
List Builder control	183
Marquee control	185
Notebook Tab control	186
Notebook control	187
Radio button control	188
Selection tool	190
Script control	191
Subform control	193
Table column control	194
Table control	198
Text area control	201
Text control	203
Timer control	206
Visualization control	208
Web Preview tool	209
Wrap label control	209
Enabling HTML in forms	211
Using the HTML Editor	211
Add an HTML Editor or Viewer to a form	212
Enable HTML Editor whitelist	213
HTML Editor controls	215
HTML Editor keyboard shortcuts	217
Add a subform to a form	217
Sort the data in a subform	219
Using pop-ups	219
Add a pop-up to a form	220
Add a dynamic form to a form	221
Building a graph diagram	223
Components of the graph	223
Defining the graph in XML format	224
W3C schema for graph diagram XML	226

Valid values for graph elements and attributes	228
Referencing images in the XML for the graph control	230
Defining actions on nodes and edges in the XML for the graph control	231
Graph toolbar icons	233
Functions and displayevent definitions for actions in the graph diagram	233
Activate an action on a node or edge in a graph	234
Set the focus on the graph	235
Forms Designer best practices	235
Form design	236
Form layouts	237
Fonts	237
Web client forms	238
Collapsible sections	238
Sizing graphics	239
Form naming conventions	239
Building accessible forms	240
Accessible Web client forms	240
Dynamic View Dependencies	241
Use form dep.g to demonstrate Dynamic View Dependencies	243
Setting Dynamic View Dependencies properties	249
Using Dynamic View Dependencies vs. Data Policy	251
Field comparisons	251
Use field comparisons	251
Field value matching	252
Use field value matching	252
Dynamic functions	253
Use dynamic functions	254
Format Control	254
Format Control processes	255
Accessing Format Control	256
Open a Format Control record	257
Add a Format Control record	257
Using expressions in Format Control	258
Format Control system functions	259
Format Control Boolean (logical) fields	259
Format Control file variable	261
Turn on menu forms by using Format Control	261
Sequential numbering for Format Control	261
Sequential number file	262

Data types of sequential numbers	262
Create simple sequential numbers	262
Create the form for simple sequential numbers	263
Create a database file for simple sequential numbers	263
Create a Format Control record for simple sequential numbers	264
Open the number file	265
Add a numbered record to the database file for simple sequential numbers	265
Create sequential number prefixes and suffixes	265
Create a Format Control record with prefixes and suffixes	266
Add a record with prefixes and suffixes	267
Array maintenance	268
Array structure maintenance functions	268
Add array structure maintenance options to a format control record	269
Test the array structure maintenance options I added	270
Sorting simple arrays	271
Sort simple arrays	272
Create the form for sorting simple arrays	272
Create the Format Control record for sorting simple arrays	273
Create a data record for sorting simple arrays	273
Determine parameters without using the RAD Editor	274
Invoke auditing from Format Control	275
Open the audit specifications table	276
Test audit lookup functionality	276
Add an audit specifications record	279
Define an audit specifications entry	279
Invoke auditing	281
Invoke auditing for joindefs tables	281
Set up event triggers	282
Add lookup functionality to Format Control	284
Special processing considerations: Incident Management	285
Special processing considerations: Change Management	285
Special processing considerations: master Format Control record	286
Special processing considerations: detail Format Control record	287
Special processing considerations: approval Format Control record	287
Format Control and eventout records	287
Format Control error messages	288
Common routines called from Format Control	289
Links	291
Understanding links	291

Advanced link editing features	291
Specifying a link query	292
Copying fields by name during fill operations	292
Scalar/non-scalar field links	293
Keeping changes	293
Link dependencies within Service Desk	293
Document Engine master link record	294
Add a new link file	294
Modify an existing link	295
Test a link	295
Delete a link	296
Access advanced link maintenance	296
Copy information from structure fields to scalar fields	297
Link maintenance	297
Data relationships and the link file	298
Types of links	298
Find functionality	299
Fill functionality	300
Fill functionality with multi-select	301
Turn off multi-select functionality	301
Virtual joining functionality	301
Us.link	302
Variables used in links	302
Calling us.link	305
Skipping query writing	305
Accessing \$File/dates	307
Find from and Fill to a \$ variable	307
Find from and Fill to an array structure	307
The \$fill.display and \$fill.display.add functionality	308
Access the link record	308
Virtual joins	310
Understanding subforms	310
Example: Creating a virtual join	310
Build the sales form	311
Create the sales file	312
Create the sales record list	313
Add data to the sales file	314
Create the sales1 subform	314
Create the orders form	315

Build the virtual join into the form	316
Build the link	318
Use the virtual join	318
Verify that the sales1 form works	319
Entity Relationship Diagram creation utility	319
Checklist: Creating an ERD record for an application	320
What is the required input for ERD Create?	320
Create an ERD definition	320
Create an ERD record	321
Modify an ERD record	322
Create manual relationships	323
What are Master link records?	323
What are related link records?	324
Associate a data policy record with an application	324
Checklist: Generating a DDL file	324
DDL mode	325
Generate a DDL file	325
Import a DDL file into a database modeling tool	326
Posting	327
Posting link records	328
Posting link line definition file	328
Types of posting available in Format Control	328
Confirming the posting routine	329
Posting variables	329
Automatic updates using Format Control subroutines	330
Identify field input values for posting	331
Create the link record for posting	331
Create the Format Control record for automatic posting	332
Open a request for posting	334
Manual posting using Format Control additional options	335
Create the link record for posting	336
Create the Format Control record for manual posting	337
Modify a hardware change request for posting	338
Use the confirmation function in the posting routine	340
Do manual posting with confirmation	340
Display functions	342
Menus	342
Menu record	342

Button properties	343
Access menu records	343
Example: Adding a URL link to a menu	343
Customize System Navigator menu icons	345
Adding and changing the image icon for menu and toolbar items	346
Add or change the image icon for options menu and toolbar items	346
Common option icon names	347
Display application	350
Displayscreen definitions	351
Display application option definitions	352
Restricting access to display options	356
Selecting display options	356
Calling an application	357
Custom RAD	357
Creating displayscreen records	357
Displayoption database dictionary keys	358
Displayevent database dictionary keys	358
Access display records	359
Create a displayscreen record	360
Create a displayoption record	361
Define display conditions	362
Enable the Merge Conflicted Updates function for customized user operations	363
Task 1: Add a display option for the Merge Conflicted Updates function	364
Task 2: Add process call in the State record of a Service Desk Object.	364
Task 3: Enable auto merge operation for the Save operation on Service Desk.	365
Advanced functions	366
Service Manager macros	366
Macro conditions	366
Accessing macro records	367
Access a macro record	368
Create a macro	369
Definitions for macro forms	370
Macro list form definition	370
Macro editor definition	371
Macros provided with Service Manager	371
Publish and subscribe	373
Publish and subscribe workflow	373
Static messages	374

Publish the static message	374
Modify the form for static messages	374
Modify the form for a simple marquee	375
Publish the marquee	375
Stored queries for management view publishing	376
Agent records for publishing	376
Create an agent record for publishing	376
Scheduling background processes for publishing	377
Establish the publishing interval in the schedule file	377
Load agents into the startup record for my system	377
Start background processes	378
Stop background processes	378
Define which system processes manage message traffic	379
Marquee processor	379
Establish the publishing interval in the marquee processor	379
Schedule file	380
Subscribing	380
Publishing in the RAD environment	380
Scripting	381
Scripts	382
Script forms	382
Script flow	383
Scripting processing flow	383
Diagramming the script flow	384
Using fill boxes in script forms	384
Executing scripts	385
Executing a script from Incident Management	386
Executing a script from displayoption	386
Executing a script from Format Control	386
Executing a script from a stored query	386
Script reports	387
Access a script record from a menu	388
Access a script record from the Database Manager	388
Create a script	388
Define the scripts	389
Define an initial script in an Incident Management profile record	391
Execute a script from a displayoption	391
Execute a script from Format Control	392
Delete a script	393

Print a report on a script	393
Service Manager Web tier	394
Editing Web client keyboard shortcuts	395
Using the compact layout	395
Setting web client preferences	395
Set Web client preferences from the server	396
Optimize web client cache control	396
Branding the web client	398
Update the operator records to enable the branding rights	399
Specify the location for the branding files	399
Enable the branding menu in old applications	400
Branding implementation options	401
Additional branding implementation options	404
Web client forms	406
Configuring color indicator	407
Accessible Web client forms	409
Example: Running custom JavaScript from the Web client	410
Example: Sending Web tier URLs through e-mail notifications	410
Example: Notifying specified operators upon an incident update	415
Generating Web tier URL queries	417
Windows client	419
Administration perspective	419
Console view	420
Stopwatches view	420
XML views	421
Administrative views	422
Access an administrative view	424
Preferences	425
Set Windows client preferences	425
Appearance preferences	426
HP Service Manager preferences	427
Log preferences	428
Security preferences	429
Spell Checker preferences	429
Spell checker custom dictionaries	430

Change the spell checker dictionary	430
Help preferences	431
Help Server preferences	432
Setting Windows client preferences from the server	433
Set Windows client preferences from the server	433
Views	433
Views	435
Fast views	436
Create a fast view	436
Related objects	437
Workflow view	437
Messages view	438
Active notes	439
View messages	439
Perspectives	440
Create a new perspective	440
Open a list of perspectives	441
Implementing version control for your tailoring	442
Version control process	443
Version control best practice	449
Create an operator record for each developer	449
Streamline the code repository versions	449
Turn off this function when it is not needed	450
Applications data file represented in XML	450
Parameters to import and export files	456
Send Documentation Feedback	458

Tailoring

Service Manager Tailoring has information for System Administrators and implementers who install and configure Service Manager. Use Service Manager Tailoring to make further changes to support site-specific requirements, including special field validation, new or modified forms design, expanded or varied workflow, and automatic notifications.

Tailoring is any change to standard functionality without changing actual code. For example, you can:

- Change the look and operation of forms.
- Change default values for objects on forms that Service Manager uses for field validation.
- Create macros, scripts, and stored queries.
- Make changes to record definitions.

Important: Service Manager does not support excessive tailoring such as changing the user access of a module. For example, the Incident Management module is intended for power users only; if you tailor your system to make this module accessible through the employee self-service (ESS) interface, unpredictable results could occur to your system. For more information about the Windows client and different web client views, as well as their target user groups, see [Windows client](#) and "[Service Manager Web tier](#)" on page 394.

Tailoring Best Practices Guide

The *HP Service Manager Tailoring Best Practices Guide* aids Service Manager implementers who are responsible for tailoring Service Manager. The guide also has best practices on the following topics:

- How to create forms
- How to write efficient queries
- How to create efficient RDBMS mappings
- How to re-map an array
- How to manage global lists
- How to reduce upgrade conflicts

You can view and search this guide using Adobe® Reader, which you can download from the Adobe Web site.

The *Tailoring Best Practices Guide* is available from the help.

Environment configuration

A complete HP Service Manager environment consists of at least three different instances of Service Manager:

- A development instance, which is used to customize Service Manager to the exact needs of the customer
- A test instance, which is used to test these changes in a copy of the production system before promoting the customizations into production
- The production instance itself

These topics focus on environment configuration options within the system. The main utilities that enable environment configuration are:

- Development auditing — tracks changes to records during the development phase of implementation.
- Differential Upgrade — moves changes from a development environment into a test or production environment.
- Document Engine — defines processing rules for files.
- Revision control — reverts to a previous version of a file or format.
- Wizard creation tool — simplifies repetitive tasks.

Applying customizations to the production environment

HP Service Manager offers the system developer different tools to make the transitions from development to test, and from test to production, easier.

- Development auditing
- Unload script utility
- Differential Upgrade utility

- Revision control
- Change Management

Comparison of the tools to use for promoting customizations

Each method of promoting customizations has advantages and disadvantages. The following table compares what each method is capable of providing.

Note: Manual processing involves writing down each change as it is being made, and then unloading records one-by-one manually via the Service Manager Database Manager. Manually unloading each record takes a lot of time and is prone to errors.

Ability	Development auditing	Unload script utility	Differential Upgrade Utility	Manual Processing
Can monitor all files?	No. Hard-coded list of tailoring files	Yes	Yes	Yes
Possible to specify a detailed query?	By date only	Yes	Yes	Yes
Can select and deselect records to unload?	Yes, by removing records from the devaudit file	No	Yes	No
Can do dbdict field merge?	No	No	Yes on fields, but not on keys	Yes, by changing the dbdict load option
Are changes to the record included in unload?	All	Latest	Latest	Latest
Performance Ranking, includes creating and applying patch	Poor	Good	Excellent	Fair
Ranking by ease of use	Good	Excellent	Fair	Poor
Ranking by ease to set up	Excellent	Good	Fair	Poor

Development auditing

The development auditing utility (devaudit) tracks changes made to HP Service Manager records during the development phase of Service Manager implementation. Whether you make a few changes or extensively customize your system, it is critical to have a record of your changes to ensure loading of the correct version when you move to production. With it, you can create an unload of your changes, and load them into another system.

Note: The Differential Upgrade utility is a more powerful and flexible tool for this purpose. Hewlett-Packard recommends that you use it instead of the development auditing.

Though the auditing utility helps you find modified records, HP strongly recommends that you record each change more extensively either using tools outside of Service Manager or using the Revision control feature.

The development auditing utility tracks changes to the following files:

- application
- code
- datadict
- devaudit
- displayevent
- displayoption
- displayscreen
- enclapplication
- eventfilter
- eventmap
- eventregister
- format
- formatctrl
- joindef

- link
- menu
- Object
- Process
- querystored
- scmessage
- screlconfig
- scripts
- States
- trigger
- validity
- wizard

Differential Upgrade utility

The HP Service Manager Differential Upgrade utility enables a developer to compare the specified files of two Service Manager systems to create a single unload file. The resulting unload file contains all the necessary records to make the files identical between the two systems.

The Differential Upgrade utility simplifies the way you move changes from a development environment into a test or production environment. You can also use this utility to move files between any two systems, such as:

- Development and test
- Unit testing and acceptance testing
- Two different development systems

Important: HP recommends that you understand how to export files and create different types of unload files before you use the Differential Upgrade utility.

The Differential Upgrade utility, just like the upgrade utility that upgrades applications from one version to another, relies on patch records and signatures. For more information on how to use the Differential Upgrade utility, see the Service Manager online Help topics under "Application Development - Differential Upgrade utility."

Unload script utility

The unload script utility enables system administrators to create HP Service Manager unload files automatically. The unload script utility enhances the standard unload creation process in many ways. With unload scripts, you can:

- Save records from multiple tables into a single unload file.
- Specify a query for each source table that filters the records added to the unload file.
- Purge records during the unload process.
- Specify which formats to protect during a purge process.
- Add related records from the data map file to the unload file.

The unload script utility is available in Tailoring. By default, Service Manager includes a collection of unload scripts that you can use for common unload tasks. You can also use the default unload scripts as templates to create your own customized unload scripts.

To create and use unload scripts effectively, you must be familiar with the Service Manager Database Manager. You can create a query on any field in the file, but querying keyed fields improves system performance and response time. You can create record queries using Structured Query Language (SQL) syntax.

You can use the Purge/Archive utility to unload and purge records from one file; if you use datamaps, you can use Purge/Archive to unload and purge related records as well.

After you load a .unl file to import data into Service Manager, you must stop and then restart the Service Manager server. This ensures that the table changes are initialized.

Note: The Unload Script utility unloads tables using the binary unload file format.

Promote customizations from development to production

1. Before you promote customizations from development to production, thoroughly document every change made in the development system that will need to be part of the patch that is promoted to production. The information that you collect in this documentation depends on the environment, but should at a minimum include the following fields, which are mentioned in the log file:
 - o Change Number
 - o Request Description
 - o Task Number
 - o Task Description
 - o Change Requester
 - o Change Owner
 - o Record modified
 - o Date Modified
 - o Change Description
 - o Unload Name
 - o Patch Name
 - o Test plan

You can document changes either inside of HP Service Manager Change Management or by using an external program such as Microsoft Excel.

2. Test all changes in the development system.
3. After all changes are documented, tested, and functioning in the development system, create a single unload.

HP recommends that you:

 - a. Remove the field "keys" from the exclude list in the signaturemake file for the dbdict table before creating the signatures on both production and development. That way, dbdicts that had changes to the keys will be unloaded into the file that is loaded into test and then production and these changes are easier to find.

- b. Use the Differential Upgrade utility to create a single unload file containing all changes.
 - c. Manually modify all keys that were changed in any of the dbdicts contained in the unload
4. After the single unload is created, perform the following steps:
 - a. Apply it to a test system that is a recent copy of the production environment.
 - b. Test the changes thoroughly.
 - c. Document any reported issues and fix these in the development system after each test iteration.
 - d. Provide a repaired patch file (while still using a single unload file) to the testers at the beginning of each test iteration.
5. After the unload is thoroughly tested and accepted, the latest unload can be applied to a production system or to any other system that needs to be upgraded with the patch, such as a training system.

Note: Do not modify the patch after this point. Address new issues found after testing is complete in the next development cycle.

Best practices for promoting customizations

The following items are vital in any environment for successful customization of an HP Service Manager system:

- The process has to be defined and consistently applied.
- All changes have to be done in one system, the development system, and must be thoroughly documented.
- A single unload should contain all records that were customized. In some environments, many developers work on developing customizations. In such environments, a central system should be used to load the individual changes, using revision tracking. That central system can then be used to create the unload.
- All changes must be tested in a central system, the test system, and issues found there must be repaired in the development system.

Have a well defined process

It is important that you have a well defined and documented process for proceeding from development through test into production. A well defined process will promote user satisfaction and prevent delays in the development cycle.

Document all changes

HP strongly recommends that you document each change made when you tailor the system, regardless of how the promotion will be done.

Part of the development process should include documenting all changes. Changes can be documented in an audit log that is stored outside of Service Manager, such as in an Excel® spreadsheet.

Another possibility is to use Service Manager Change Management for keeping track of all changes done during the development process.

Important: You must keep track of every change done to the development system, so that no change is lost when creating the patch unload.

Create a single unload

After the customizations are completed on the development system, create a single unload to apply to the test system.

Warning: If you create and apply multiple unloads, you increase the possibility that records can be overwritten or lost.

Use a development system

Make all changes in the development system. Making changes in the test system or directly to the production system will cause the systems to become out-of-sync and more difficult to maintain. Applying changes to a system that is out-of-sync with the system on which the changes were based will most likely result in broken functionality.

Use a test system

Use a test system to apply the previously-created unloads and test their functionality. If problems are found during these tests, they can be fixed in the development system where a new unload will be created. Repeat this cycle until the test is successful.

After testing is complete, you can apply the tested unload to the production system. Some companies require training before an unload is released to production. You can apply the tested unload to a training system first, and then to production. The benefit of having a single tested unload is that it can be applied to as many systems as necessary.

Development auditing

The development auditing utility (devaudit) tracks changes made to HP Service Manager records during the development phase of Service Manager implementation. Whether you make a few changes or extensively customize your system, it is critical to have a record of your changes to ensure loading of the correct version when you move to production. With it, you can create an unload of your changes, and load them into another system.

Note: The Differential Upgrade utility is a more powerful and flexible tool for this purpose. Hewlett-Packard recommends that you use it instead of the development auditing.

Though the auditing utility helps you find modified records, HP strongly recommends that you record each change more extensively either using tools outside of Service Manager or using the Revision control feature.

The development auditing utility tracks changes to the following files:

- application
- code
- datadict
- devaudit
- displayevent
- displayoption
- displayscreen
- enclapplication
- eventfilter
- eventmap
- eventregister

- format
- formatctrl
- joindef
- link
- menu
- Object
- Process
- querystored
- scmessage
- screlconfig
- scripts
- States
- trigger
- validity
- wizard

Access the development auditing utility

Applies to User Roles:

System Administrator

To access the development auditing utility:

1. Click **Tailoring > Audit**.
2. Select a function from the Audit menu.
 - **Audit Log:** The repository for data gathered during the audit. It displays the old and new values of input field data, as well as who made the revisions and when.

- **Audit Specifications:** Provides instructions on how and when to perform an audit. Use it to define files and fields that you want the audit application to monitor.
- **Purge Audit History:** After the development phase is completed and all related changes have been transferred into the production system, you can purge the development audit data. Ensure you carefully review the audit records in the Audit History record list before filling in the date to purge files, as you cannot restore purged files.
- **Turn Auditing On/Off:** Enable or disable file auditing. If auditing is on, the system creates a separate audit record each time you make a change to a record and saves it to the database. HP Service Manager contains a default set of audit records to enable you to track development changes.
- **Unload an Audit Delta:** Unload a change you made during the development phase. It is critical that you check the audit files for the correct date of this delta. You need the date shown in the Audit History form to enter into the Unload form. When the Audit Unload form opens, the current date and time automatically fill in the **Unload delta since what date?** field. Type the beginning date for the unload that is found in the **View Audit History** record list.
- **View Audit History:** Search for a particular audit record or view a record list of all audits stored in the database from the Audit History form. It is important to look at this file before purging audit records or unloading a development change to production.

Turn the auditing on/off

Applies to User Roles:

System Administrator

This function enables you to enable or disable file auditing. If auditing is on, the system creates a separate audit record each time you make a change to a record and save it to the database. HP Service Manager contains a default set of audit records to enable you to track development changes.

1. Click **Tailoring > Audit > Turn Auditing On/Off**.
2. Update one or more of the following functions:
 - **Do you want to audit development changes?:** Enables/disables auditing of development changes. When selected, development auditing tracks the changes you make. You can put those changes into an unload file and load them into another system.

- **Do you want to keep backups of Changes?:** Enables/disables backups of development changes. When selected, development auditing creates a copy of the record you changed in the same table. The name of the backup record is prefaced with AUDIT and the number of the audit record and key. Restores the originals by renaming them to the original file name.
- **Do you want to create auditdelete records on unload?:** Enables/disables creating auditdelete records on unload. When selected, development auditing tracks your deletes, and the unload file makes the same deletes when you load it into another system.

3. Click **Save** and **OK**.

Best practices

Auditing should always be disabled on a production or test system, and enabled for development systems (when auditing is used to promote from development to production).

When selecting auditing settings for a development system:

- Select **Do you want to audit development changes?**
- Uncheck **Do you want to keep backups of changes?**
- Uncheck **Do you want to create auditdelete records on unload?**

View the audit history

Applies to User Roles:

System Administrator

Search for a particular audit record or view a record list of all audits stored in the database from the Audit History form. It is important to look at this file before purging audit records or unloading a development change to production.

If some of the records should not be added to the auditing delta unload, you can either change the search criteria of these records (such as changing the date to an earlier date) or delete the record altogether if it should never be part of the delta and does not need to be documented as changed.

Note: Be aware that the same record may be included in the devaudit file several times, since each change to a record is noted in devaudit.

1. Click **Tailoring > Audit > View Audit History**.
A blank Audit History record opens.
 2. Open an existing Audit record using one of the following procedures:
 - Enter any information you have from the record and click **Search**.
 - Click **Search** or press Enter. If more than one record matches the search criteria, the result screen is split with a record list at the top and the first record in the list open below.
- Note:** There is an entry for each time you add or update a form or file.
3. Click the entry you want to view. Information for this record is entered into the Audit History form.
 4. Click **Delete** to remove the current record from the audit list.

The records have the following structure:

- Audit ID: 1
- Filename: format
- Keys: IM.update.incident
- Event Type: update
- Date: 01/14/10 12:31:29
- Operator: falcon
- Last Record: true

Unload an audit delta

Applies to User Roles:

System Administrator

To specify what the system unloads, you must first view the audit history. Remove records from the development audit list by selecting them and then deleting them.

Important: Before unloading a change you made during the development phase, it is critical that you check the audit files for the correct date of this delta. You need the date shown in the Audit History form to enter into the Unload form.

To unload the audit delta:

1. Click **Tailoring > Audit > Unload an Audit Delta**.
The Audit Unload form opens, and the current date and time automatically fill in the **Unload delta since what date?** field.
2. Type the beginning date for the unload that is found in the **View Audit History** record list.
3. Type the path and file name that you want to send the unloaded data in the **Send Data to Which File?** field.
4. Type the path and file name of the log for the audit unload process in the **Enter Log Name Here** field.
5. Click **Proceed**. The Development Auditor menu opens with a message in the status bar indicating how many records were unloaded.

The unload you create can be loaded into the test system, and eventually into the production system.

Note: You can view the contents of an unload file in Database Manager before loading it onto the new system.

To view the contents of an unload file:

1. Open Database Manager.
2. Select **Import/Load** from the More Actions menu.
3. Select the file you want to load.
4. Click **View Contents**.

Purge audit records

Applies to User Roles:

System Administrator

After the development phase is completed and all related changes have been transferred into the production system, you can purge the development audit data. You are prompted to specify the start date for records to be purged. After the data is purged, it cannot be restored (unless an unload of the devaudit file was performed beforehand).

To purge audit records:

1. Click **Tailoring > Audit > Purge Audit History**.
The Audit Purge form opens with the current date and time filled in.
2. Replace the date displayed with the beginning date for the range of records you want to delete.
Warning: Ensure that you carefully review the audit records in the Audit History record list before filling in this date. You cannot restore purged files.
3. Click **Proceed**. A confirmation prompt opens informing you how many records have been purged.
4. Click **OK** to complete the process and return to the Development Auditor menu.

Differential Upgrade utility

The HP Service Manager Differential Upgrade utility enables a developer to compare the specified files of two Service Manager systems to create a single unload file. The resulting unload file contains all the necessary records to make the files identical between the two systems.

The Differential Upgrade utility simplifies the way you move changes from a development environment into a test or production environment. You can also use this utility to move files between any two systems, such as:

- Development and test
- Unit testing and acceptance testing
- Two different development systems

Important: HP recommends that you understand how to export files and create different types of unload files before you use the Differential Upgrade utility.

The Differential Upgrade utility, just like the upgrade utility that upgrades applications from one version to another, relies on patch records and signatures. For more information on how to use the Differential Upgrade utility, see the Service Manager online Help topics under "Application Development - Differential Upgrade utility."

Patch records

A patch record specifies which files the Differential Upgrade utility should compare, and creates a query that limits the comparison to certain records within files. There must be a patch record for each system in the comparison.

Note: The query should limit the number of records to upgrade. The more files in the patch record, the longer the Differential Upgrade process takes.

In a Differential upgrade comparison, you must have one patch record for each system to be compared. The patch record must point to identical files in each target system and generate the same queries. You can either create it once on each system; or create it on one system, unload it, and load it into the other system. You may prefer to create customized patch records before you start the Differential Upgrade process.

Note: Be sure to include all files that were modified during customizations that you promote to production.

One method of limiting the number of records included in the patch record is to use the `sysmodtime` field that many `dbdicts` contain. Service Manager updates this field automatically with each update of the record. If the file does not contain a `sysmodtime` field, you can either add it before starting the customization work on the development system, or use another limiting field such as `update.time`.

In the Patch record dialog box, you can specify that the Differential upgrade should perform Add only processing on a specific file. If you choose this option, Service Manager adds only new records to the Differential upgrade unload file and ignores changes to existing records in the development system.

Signature records

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, based on the definitions in the `signaturemake` file.

You must create signature records for any record that you compare in the Differential Upgrade process. The `signatures` file contains signature records.

Important: Never change a record in the `signaturemake` file.

Checklist: Differential upgrade steps

Applying a Differential upgrade from a development system to a production system requires these steps:

1. Create separate Patch records for the development and production systems. The patch records must reference the same files on each system.
2. Create Signature records for the production (target) system.

3. Move Signature records to the development system.
4. Create the Differential upgrade from the development (source) system. You can create an external unload file or an internal file.
5. Customize the Differential upgrade unload files. This is optional if you created an internal file.
6. Create a differential upgrade unload from internal files.
7. Load the Differential upgrade unload into the production (target) system.

Create or update a Patch record

Applies to User Roles:

System Administrator

To create or update a Patch record:

1. Click **Tailoring > Differential Upgrade > Patch Records**.
2. Type the name of the Patch record in the **Patch Name** text box.
3. Type a query (or false) in the **Application Clusters** text box. For example:
`application = "app_name"`
4. Type a query (or false) in the **Format Records** text box. For example:
`name = "format_name"`
5. Type the names of files to be included in the Patch record in the Dbdict fields.
6. Click the **Add** check box if you want to Add new records only. Otherwise, the Differential Upgrade utility processes both updated and new records.
7. Type true, false, or an advanced query in the Query field for each file. True includes all records from the file; false does not return any records. An advanced query enables you to customize the results.
8. For a new Patch record, click **Add**.
9. To update a Patch record, click **Save**.
10. Click **OK**.

There must be a Patch record for each system in the comparison. Each patch record must point to the same files and records. You can create it once on each system, or create it on one system, unload it, and load it into the other system.

Create Signature records on the production system

Applies to User Roles:

System Administrator

To create Signature records on the production system:

1. Click **Tailoring > Differential Upgrade > Differential Upgrade Wizard**.
2. Click **Build Signatures**.
3. Click **Next**.
4. Type the name of the system in the **System Name** text box. Remember the name because HP Service Manager uses it in later steps.
5. Click the drop-down list to choose an existing Patch record for the named system. Remember that there must be a Patch record for each system to be compared, and each Patch record must point to the same files and records.
6. Browse to locate or define the **Export Filename**. This is an external file that will store the signature records.
7. Optionally, click **Signature RAD components for delete processing** to delete all related RAD panel components.
8. Click **Run in background** if you want background processing. If you run in background, Service Manager creates a schedule record and creates the unload when the schedule record runs. If you choose Run in background, do the following:
 - Click **Repeat Daily** to create signature records daily. This step is optional.
 - Click the drop-down list to choose the **Date/Time to Run** schedule for background processing.
9. Click **Next**.

If the process runs in background, the unload file will reside on the server. If the process runs in foreground, the unload file location depends on whether you enable the Client Side Load/Unload option on the client. Processes running in foreground post progress messages in Active Notes, if that option is enabled.

Move Signature records to the development system

1. Ensure the signatures file on the development system is empty.
2. Load the signatures file on the development system using the standard HP Service Manager Import/Load utility.

If you choose background processing, Service Manager moves the signature records automatically.

Create the differential upgrade from the development system

Applies to User Roles:

System Administrator

To create the differential upgrade from the development system:

1. Click **Tailoring > Differential Upgrade > Differential Upgrade Wizard**.
2. Click **Create Differential Upgrade**.
3. Click **Next**.
4. Double-click the correct Patch record from the list of Patch records. This should be the same Patch record specified when you created the production Signature records.
5. Type a descriptive name for the system you are building in **Current Version**.
6. Type the descriptive name for the system you are upgrading in **Compare to Version**. This should be the System Name you specified when you built the Signature records.
7. Optionally, select **Perform Delete Processing** to delete records in the target system.
8. Decide whether you want to create an external or internal unload file. Depending on your choice, complete Step 9 or Step 10.
9. To create an external unload file and a text file listing all records included in the unload file, click **Create Unload** and do the following:
 - o Click **Next**.
 - o Click **Browse** to locate a file, or define an **Export Filename** for the unload file containing the Differential Upgrade.

- Click **Browse** to locate a file, or define a **Log Filename**.
 - Click **Run in background** if you want background processing. HP Service Manager creates a schedule record and creates the unload when the schedule record runs. If you choose Run in background, do the following:
 - Click **Repeat Daily** to create signature records daily. This is an optional step.
 - Click the drop-down list to choose the **Date/Time to Run** schedule for background processing.
 - Click **Browse** to locate the **Path to Signatures unload** where the signature file resides. For background processing, the signatures unload file must be on the Service Manager server.
 - Click **Next**. When the Differential upgrade is complete, the file named in Export Filename contains the information that aligns the production system with the development system. There will be a record for each difference between the development and production systems.
10. To add a record to the `diffupg` file for each record included in the upgrade, choose **Create Internal File** and do the following:
- Click **Next**.
 - Click **Run in background** if you want background processing. If you run in background, Service Manager creates a schedule record and creates the unload when the schedule record runs. If you choose Run in background, do the following:
 - Click **Repeat Daily** to create signature records daily. This is an optional step.
 - Click the drop-down list to choose the **Date/Time to Run** schedule for background processing.
 - Click **Browse** to locate the **Path to Signatures unload** where the signature file resides. For background processing, the signatures file must be on the Service Manager server.
 - Click **Next**. When the Differential upgrade is complete, there is one record for each record included in the upgrade.

Customize the Differential upgrade unload file

Applies to User Roles:

System Administrator

If you create an internal file, you can customize the unload file before processing begins. You can page through the `diffupg` file to examine each record, and specify whether it is to be included in the unload by selecting the Unload check box.

To customize the Differential upgrade unload file:

1. Ensure that you complete the steps to run the Differential upgrade wizard and create an internal file.
2. Click **Tailoring > Differential Upgrade > Differential Upgrade Records**.
3. Click **Search**.
4. Choose a record from the list of Differential upgrade records. HP Service Manager displays the record in the Differential Upgrade Results form.
5. Select or clear the **Unload?** checkbox to include or exclude the record from the Differential upgrade process.
6. Click **Next**.
7. Repeat these steps until you page through all of the records in the file.

Note: Click **Go To Record** to view the complete record.

Create a differential upgrade unload from an internal file

Applies to User Roles:

System Administrator

To create a baseline revision:

1. Click **Tailoring > Differential Upgrade > Create Differential Unload**.
2. Type an **Export Filename**.
3. Type the **System Name**. You are creating differential upgrade unload records for only this system. This is the name of the system listed in the differential upgrade records.
4. Click **Next** to create the unload.

Load the Differential upgrade into the production system

Use the standard HP Service Manager import/load utilities to load the Differential upgrade unload file into your production system.

HP Service Manager Document Engine Guide

The *HP Service Manager Document Engine Guide* aids Service Manager implementers and developers who are responsible for tailoring Service Manager. The guide provides an overview of the Document Engine and a description of the Document Engine components. It also includes a detailed example that demonstrates how to use the Document Engine to implement a work order system in Service Manager.

You can view and search this guide using Adobe® Reader, which you can download from the Adobe Web site.

The *HP Service Manager Document Engine Guide* is available from the help.

Revision control

Revision control provides developers and administrators with a means of reverting to a previous version of a file or form. If during the process of creating or modifying forms you find an error, revision control returns a working version of your file or form.

Revision control allows a developer to:

- Create a snapshot of a record
- Add SCR information and comments to the snapshot
- Replace the current version of the record with a working version of the record at any time.

Note: Every revision takes up as much disk space as the original record plus a few bytes for comments.

Use revision control in conjunction with the development auditing to track, record, and save changes to your system. The development auditing utility provides a record of the changes to ensure that you load the correct version when you move to production, whereas revision control documents these changes and enables you to create working snapshots.

HP Service Manager handles revisions as part of the Document Engine. Revisions are available in all utilities that use the Document Engine as base code, including Database Manager, Format Control, Link Editor, Forms Designer, the RAD Editor, and others.

The system stores revisions in a separate file whose name is specified in either the object record for the file or in the datadict record. The system creates this revision file via an option on the Data Policy or the object screen. Administrators specify the maximum number of revisions to store for each record in a file. If you do not specify a number, an unlimited number is stored.

Administrators need to determine in advance in which files to track revisions and then do a minor setup to establish them. Administrators also need to purge revisions prior to migrating to a production system.

Creating revisions

The revisions option for Data Policy creates a revision database dictionary record, whose name is specified on the Engine Specifications tab. You can create revisions either for an entire set of records or for a single form or record. You must be at the start panel of the application you are working with to create baseline revisions for an entire set of records. For example, you create a baseline revision of all the forms in the Forms Designer from the Forms Designer main menu.

When you create a revision, you create a copy of the record only, not the associated database dictionary record. As a result, any time a field is added to a database dictionary record, you add it to the revision file database dictionary record.

Create a baseline revision

Applies to User Roles:

System Administrator

To create a baseline revision:

1. Open a search screen for a form or file that supports revisions. For example, the Forms Designer application.
2. Click **More** of the More Actions icon.
3. Click **Revision**.
4. Click **Create Baseline Revision**. A dialog box opens.
5. Enter the SCR information or click **Ok** to begin.
The system creates a copy of all the records in the application, in this case every form in the Forms Design application.

Create a single revision

Applies to User Roles:

System Administrator

To create a single revision:

1. Open a form or file that supports revisions. For example, a RAD application in the RAD editor.
2. Click **More** of the More Actions icon.
3. Click **Create Revision**. The revision tracking panel opens.
4. Type in information relating to the revision. Make comments to help others understand the reason for the revision.
5. Click **Save**.
A copy of the revision saves and you return to the main panel.

Revert to a previous revision

Applies to User Roles:

System Administrator

To revert to a previous revision:

1. Open the form you want to use a previous version of.
2. Click **More** or the More Actions icon.
3. Click **Revision**.
4. Click **Revisions > Find Revisions**.
5. Click the revision you want to restore.
6. Click **Revert to this Revision** from the More Actions menu. The system prompts you as to whether you want to save the current version of the record as a revision. It is recommended you do.

Search for revisions

Applies to User Roles:

System Administrator

To access the wizard creation tool:

1. Open an application that supports revisions. For example, the RAD editor or the Forms Designer.
2. Click **More** or the More Actions icon..
3. Click **Revision > Find Revision**.

You can search for revisions using one or more of up to five of the following search criteria:

- **syslanguage** — Search for revisions by language indicator. For example, en for English.
- **name** — Search for revisions by name. For example, cc.incquick.
- **Revision Date** — Search for revisions by date in the format month/day/year.
- **Operator** — Search for revisions by operator name. Enter the name of the operator that created the revision. For example, SYSTEM ADMINISTRATOR.
- **SCR#** — Search for revisions by SCR number. For example, 42.

Purge revisions

Applies to User Roles:

System Administrator

Purge scripts help administrators with revision maintenance. The `sc.revision.purge.hanging` script purges all revisions that no longer have a parent record because the parent record was deleted or renamed. The `sc.revision.purge` script purges all revisions from the system.

There are two options available when purging revisions.

To search for revisions:

- **Purge All Revision Records** removes all revisions from your system.
- **Purge Hanging Revisions** removes only hanging revisions from your system. A hanging revision is a revision with no associated file or form. For example, if you create a form, create a revision of the form and then delete the original form, the revision remains on the system but with no associated file or form.

Note: You can restore a deleted form from its revision file.

1. Log in as a System Administrator.
2. Click **System Administration > Base System Configuration > Miscellaneous > Purge Hanging Revision Records**.
3. To remove hanging revisions from your system, click **Purge Hanging Revisions Records**.
4. To remove all revisions from your system, click **Purge All Revision Records**.

HP Service Manager Wizards Guide

The HP Service Manager wizard creation tool lets implementers and administrators add wizards to applications within HP Service Manager to assist users with performing certain tasks. For more information, refer to the Wizards Guide.

You can view and search this guide using Adobe® Reader, which you can download from the Adobe Web site.

The *HP Service Manager Wizards Guide* is available from the help.

HP Service Manager Programming Guide

The *HP Service Manager Programming Guide* provides an overview of the RAD system language in Service Manager. It also provides an overview of JavaScript functionality in Service Manager, using both examples and descriptions of JavaScript objects, properties, methods, and functions.

You can view and search this guide using Adobe® Reader, which you can download from the Adobe Web site.

The *HP Service Manager Programming Guide* is available from the help.

Data management

These topics focus on data management within the system. The main utilities that comprise data management are:

- Cascade updates — alters the data in one or more dependent files to match changes made to data in a source file.
- Data Policy — applies values, sets mandatory fields, and validates at the table level.
- Data validation — defines a field's valid values.
- Global lists — predefined lists within the system.
- Customizing the Help system — provides administrators with instructions on how to customize the online Help system.
- Sequential numbers — used in conjunction with Format Control to generate sequence numbers for records in a database.
- Stored queries — retrieves and displays information by using predefined search parameters.

Fields and Keys tab

The Fields and Keys tab describes the properties of all fields and keys in a HP Service Manager table. For administrators, this tab gathers field-related tasks into a central point of access.

The Fields and Keys tab contains the following sections and links. Some sections or links may not appear unless they relate to the selected field.

Section or link	Description	Button or property	Button functionality	Comments
Fields section	View, add, delete, or edit a field or array. A question mark decorator on the field icon indicates that there is a default help record for the field.	New field	Create a field in the tree.	Each field must have Service Manager data type. To create a new array use the New Array option first and then add fields to the array.

Section or link	Description	Button or property	Button functionality	Comments
		Delete	Remove a field.	Unavailable if the table contains records or the field is a key.
		New array	Create an array and a child node (inner field) that has the same name and an unspecified data type.	You must choose a type for the child node after you save the array.
		Search field	Search for a field or alias that contains a specified string value.	Look for the field by name, partial name, index value, alias, or Web Services API name.
Section or link	Description	Button or property	Button functionality	Comments
Keys section	View, add, delete, or edit keys.	New	Create a new empty key.	You must choose a key type after you save it.
		Delete	Remove a key.	
		Up	Move the key position up.	Not available for the first key.
		Down	Move the key position down.	Not available for the last key.
Section or link	Description	Button or property	Button functionality	Comments

Section or link	Description	Button or property	Button functionality	Comments
General properties section (fields and arrays)	View or edit the Service Manager definition of fields and arrays. The System Definition utility displays these properties when you select a field.	Name	Unique field name.	

Section or link	Description	Button or property	Button functionality	Comments
		Type	Specify the Service Manager data type of the field.	The Service Manager data types are abstractions of the RDBMS data types.
		Index	Identifies the order field within the Service Manager definition.	The index of an unsaved field is -1 until you save the field.
		Caption	Specify an optional default label for this field.	Forms Designer uses this value to create default column labels and chart titles. If you omit this information, Forms Designer creates a caption from the field name.
		Unique in domain	Appears only for a field in a structured array.	
		Rename the field	Click to change the field name. Proceed with caution. Renaming a field can invalidate existing queries.	If Service Manager has add permissions on the back-end RDBMS, it automatically creates the new field. Alternatively, the Service Manager administrator can provide the RDBMS administrators with DDL for field name change.
		Create/Edit default help on this field	Click to open the field Help Editor where you can view, add, delete, or edit field Help content. Service Manager accesses this help record whenever the field appears in a form.	
		Search for	Link to the field Help	

Section or link	Description	Button or property	Button functionality	Comments
		specific help on this field in forms.	Editor where you can click Search to view a list of other help records that will override the default help record for the field. The best practice is to create only one help record for each field. If you customize the same definition for each form where the field appears, field help maintenance increases dramatically.	
Section or link	Description	Button or property	Button functionality	Comments
General properties section (keys)	View or edit the Service Manager definition of keys. The System Definition utility displays these properties when you select a key.	Type	Specify the data type of the key.	Be sure to understand the data in the field before you change the Service Manager field type so that you can select a compatible data type. Hewlett-Packard recommends that you do not change the data type for any unique keys in out-of-box tables.
		Add	Add a field to the key from a list of valid fields.	If Service Manager has add permissions on the back-end RDBMS, it automatically creates a new index based on the key. Alternatively, the Service Manager administrator can provide the RDBMS administrators with

Section or link	Description	Button or property	Button functionality	Comments
				DDL for the new index.
		Remove	Remove a field from the key.	If Service Manager has add and drop permissions on the back-end RDBMS, it automatically removes the existing index corresponding to the key and creates a new index. Alternatively, the Service Manager administrator can provide the RDBMS administrators with DDL for the new index after the key has been removed.
		Up	Move the field up one level within the key.	If Service Manager has add and drop permissions on the back-end RDBMS, it automatically removes the existing index corresponding to the key and creates a new index. Alternatively, the Service Manager administrator can provide the RDBMS administrators with DDL for the updated index.
		Down	Move the field down one level within the key.	If Service Manager has add and drop permissions on the back-end RDBMS, it automatically removes the existing index corresponding to the key and

Section or link	Description	Button or property	Button functionality	Comments
				creates a new index. Alternatively, the Service Manager administrator can provide the RDBMS administrators with DDL for the updated index.
Section or link	Description	Button or property	Button functionality	Comments
Aliases section	View, add, delete, or rename an alias for a field.	New	Creates a separate name (alias) that Service Manager applications can use to find the field.	Field aliases must be unique within a table.
		Delete	Removes the selected field alias.	Deleting an alias does not delete the original field.
		Rename	Renames the selected field alias	Field aliases must be unique within a table.
Section or link	Description	Button or property	Button functionality	Comments
SQL Mapping section	View or edit the SQL mapping of Service Manager fields in a back-end RDBMS.	SQL table alias	Type the 3 character name that Service Manager uses to identify what type of back-end RDBMS table the field definition relates to. For example, m1 refers to the first main table and a1 to the first alias table.	Service Manager uses this name to look up the actual table name from the RDBMS.
		SQL field name	Type the name of the column in the back-end RDBMS you want to map to this field.	Make sure that the you do not use any special characters or reserved words that your RDBMS excludes.
		SQL data	Type the RDBMS data	Do not include the

Section or link	Description	Button or property	Button functionality	Comments
		type	type you want to use for this field. You can leave this value blank for new fields and Service Manager will assign a default data type.	length as part of the data type. Use the SQL data length field for this information.
		SQL data length	Type the number characters you want this field to use. You can leave this value blank for new fields and Service Manager will assign a default data length.	Do not include the RDBMS data type as part of the field length. Use the SQL data type field for this information.
Section or link	Description	Button or property	Button functionality	Comments
Data policy section	View or edit data policy rules for a field.	Default value	Specify a default value for the field.	Not valid for a field in a structured array. Enter an expression by preceding it with a caret (^). For example, ^operator() specifies that the field contains the current user log-on name.
		Data encrypted	Encrypt the data in the field.	Not valid for a field in a structured array.
		Match table	Specify the name of a table that has a key relationship with the new field.	Not valid for a field in a structured array.
		Match field	Specify the name of the field within the match table that has a key relationship.	Not valid for a field in a structured array.
Section or link	Description	Button or property	Button functionality	Comments
Editing rules	Describes the	Mandatory	The field must appear if	Not valid for a field

Section or link	Description	Button or property	Button functionality	Comments
section	rules for viewing or editing a field. Specify true, false, or a Boolean expression.		the value is true.	in a structured array.
		Validation rule	Specify the Boolean expression that must be true for the field to appear.	Not valid for a field in a structured array.
		Read-only	The field cannot change if the value is true.	Not valid for a field in a structured array.
		Invisible	The field must not appear if the value is true.	Not valid for a field in a structured array.
		Available in lists	The field must appear in a list if the value is true.	Not valid for a field in a structured array.
Section or link	Description	Button or property	Button functionality	Comments
Structured array properties	Create a separate table that you can access independently. For example, the computer table contains structured arrays that you can access directly through System Definition > Table Editor .	Separate table	Check to create a table that you can access through the Table Definition section of the System Navigator.	Valid only for a field in a structured array.
		Unique table	The name of the separated table. This name is <i>parent_table_name+structured_array_name+uniq</i> . For example: computerdriversuniq.	Valid only for a field in a structured array.
		Attribute table	The name of the attribute table. This name is <i>parent_table_name+structured_array_name+attr</i> . For example: computerdriversattr.	Valid only for a field in a structured array.
Section or link	Description	Button or property	Button functionality	Comments
IR properties	View or edit	Condition	Specify a Boolean expression that must	

Section or link	Description	Button or property	Button functionality	Comments
	information retrieval rules for the table.		be true if a table is searchable.	
	This section appears only when you select the IR key definition for the table.	Detail form	Specify the form that displays any record in the list generated by a query.	
		List form	Specify the form that displays the record list generated by a query.	
		Return field	Specify the name of the field to be written back to the data source.	

If you make any changes to fields or keys, click **Save**.

Add a field to a table

Note: You must use the Windows client whenever you need to add a new field/key to a database dictionary table.

1. From the System Navigator, click **Tailoring > Database Dictionary**.
2. Place the cursor in the File Name field, and then click **Search**.
3. Select any table from the record list.
4. Click the **Fields** tab.
5. Place the cursor in a type structure to add a field to the structure.
6. Click **New Field/Key**.
7. Specify information for all required fields.
 - Choose a data **Type** from the list. For an array, choose the appropriate data type if all elements are the same type. Otherwise, choose one of the following:
 - **Text** for an array of different data types
 - **Structure** for an array of structures

- Type a field **Caption**.
- Click **Rename field** if you decide to change the name of the field.
- Specify **Data policy** settings as needed for **Default value**, **Validation rule**, and **Data encrypted** (encryption status).
- Specify **Match table** and **Match field** values to define or enforce foreign key relationships. For example, you can specify that the contact.name field in one table must match a valid contact.name field from another table.
- Provide **Editing rules** (display properties) for **Mandatory** display, **Read-only**, **Availability**, and **Invisibility**.
- If necessary, specify one or more **Alias fields** with a different name but the same index, level, and type as a selected field.

8. Click **Save**.

Tip: You can edit any field or key directly from the System Navigator. Double-click the field or key to view the details with the Table Editor.

Edit a field in a table

Note: You must use the Windows client whenever you need to add or edit a field/key in a database dictionary table.

1. From the System Navigator, click **Tailoring > Database Dictionary**.
2. Place the cursor in the File Name field and click **Search**.
3. From the System Definition utility overview, click the **Fields** tab.
4. Click **Edit Field/Key**.
5. If necessary, click the **plus sign (+)** to expand array structures.
6. Click a field to view its general properties, data policy, and editing rules on the right side of the window. Add or update any of the following:

Section	Field	Description
General properties	Type	<p>You can change type if the field is one of these types:</p> <ul style="list-style-type: none"> Boolean Date time Decimal Text <p>Changing the data type can produce unpredictable results. Proceed with caution.</p>
	Caption	Field caption
	Rename the field	You can rename the field; however, proceed with caution. Renaming a field can invalidate existing queries.
	Create (or Edit) default help on this field	Create a new Help record or edit the existing Help record.
	Search for specific help on this field in forms	You can search for an existing field help topic to apply to this field. Click Search to generate a record list of available field help topics.
Aliases	New	Click New , type a new alias name, and click OK .
	Delete	Select an existing alias and click to remove it.
	Rename	Select an existing alias, click Rename , type a new alias name, and click OK .
Data policy	Default value	A default value for the field is optional.
	Data encrypted (encryption status).	Select or clear the checkbox.
	Match table Match field	Define or enforce foreign key relationships. For example, you can specify that the contact.name field in one table must match a valid contact.name field from another table.
Editing rules	Mandatory	Type true if the field must appear on the form.
	Validation rule	Specify a condition that is true or false.
	Read-only	Type true if users cannot change the field value.
	Invisible	Type true if the field should be invisible.

Section	Field	Description
	Available in lists	Type true if the field should appear in a list.

7. If you make any updates, click **Save**.

Tip: You can edit any field or key directly from the System Navigator. Double-click the field or key to view the details with the System Definition utility.

Edit a record key

Applies to User Roles:

System Administrator

HP recommends that you do not change the data type for any unique keys in out-of-box tables.

To edit a record key:

1. Click **Tailoring > Database Dictionary**.
2. Type a file name and then use search to find the table.
3. Select a record to edit.
4. Click the **Keys** tab to edit.
5. Click a key to view or change its general properties.
 - Change the key type as needed. Proceed with caution if you change the key type to a more restrictive value.
 - Click **Add** or **Remove** to change the content of the key.
 - For IR keys, you can also edit the associated IR properties.
6. To add a key, click **New**. A new key position appears in the Keys tree structure.
7. To remove a key, click **Delete**.
8. If a key is eligible to change position, click **Up** or **Down**.

Update the size of a field

Applies to User Roles:

System Administrator

To update the length of a field in a table:

1. Click **Tailoring > Database Dictionary**.
2. Type the name of the table in File Name. For example, use rootcause to update the size of a field in this table.
3. Click **Search**.
4. Select the field to update, dump for example.
5. Change the field SQL type from VARCHAR(30) to VARCHAR(100), for example.
6. Click **OK**.

Cascade Updates

The Cascade Update utility allows Administrators to maintain database integrity and consistency by altering or deleting the data in one or more dependent files to match changes made to data in a source file.

For example, incident records are linked to the contacts table by the contact name (probsummary, contact.name = contacts, contact.name). If someone were to edit the contact record and change contact.name, the associated incident records would lose their link and become orphaned. You could set up a Cascade Update record to monitor the contact.name field in contacts and change the contact.name field in related incident records to match accordingly.

Cascade Update example

In this Cascade Update example DEFAULT HQ has moved and now needs to update all of their contact records to reflect the new location "DEFAULT CORPORATE". Use the Cascade Update Utility to automatically update all DEFAULT HQ contact records when the location has been updated in the DEFAULT HQ location record.

There are two phases to make this happen:

- Add a trigger record.
- Add a Cascade Update Configuration record.
- 1. Set a trigger to monitor the location file and call the RAD application, `cascade.update.wrapper`, when the location record is updated (After Update) in the location file.
- 2. Configure a Cascade Update Configure Record to update the location in all contact records associated with that same location.
- 3. Test the Cascade Update.

Create a Cascade Delete

To set up a Cascade Delete between two related tables (for example `probsummary` and `problem`):

1. Open the `erddef` table using Database Manager.
2. Select or create the `erddef` record that defines the relationship between the two tables.
3. Select the **Cascade Delete** option.
4. Save and Exit.

Alternatively, you can use the Cascade Update tool and follow the instructions in that tool to set up the Cascade Deletes there.

Create a Cascade Update

Applies to User Roles:

System Administrator

To create a Cascade Update:

1. Create a trigger record to monitor the source table.
2. Create a Cascade Update configuration record to set parameters for the update.

Create a Cascade Update/Delete Configuration Record

1. Click **Tailoring > Tailoring Tools > Cascade Updates**.

2. Click **Search**.

A list of cascade update/delete configuration records opens.

3. Select a configuration record to open it.

4. Fill in all necessary fields.

Note: For more detailed information on the fields, insert your cursor in that field and press **Ctrl+H**.

5. Click **Save** to save changes to an existing record.

Caution: If you are attempting to add a new record from an existing record, make sure that you do not click Save because doing so will replace the existing record with the new record. If you are adding a new record, click Add.

6. Click **Add** to add a new record.

Create the example Cascade Configuration Record

1. Click **Tailoring > Tailoring Tools > Cascade Updates**.

The Cascade Update/Delete Config Record opens.

2. Type the following information in the Cascade Update/Delete Config Record:

- o **Name:** location
- o **Type:** After Update (Configuration Rules)
- o **Target File:** contacts
- o **Execute Condition:** location in \$L.new~=location in \$L.old
- o **Update Type:** Cascade Update
- o **Source Key:** location

- **Target Key:** location
 - **Source Fields:** location
 - **Target Fields:** location
3. Click **Add**.
HP Service Manager displays the message:
Cascade Update Config record updated.

Create the example trigger record

1. Click **Tailoring > Database Manager**.
2. In the **Form** field, type:triggers
3. Press **Enter**.
The trigger record opens.
4. Type the following information in the triggers record:
 - **Trigger Name:** update.contact.location
 - **Table Name:** location
 - **Trigger Type:** 4 - After Update
 - **Application:** cascade.update.wrapper

If you are configuring a Cascade Update for device records, type **am.cascade.update.wrapper** in the **Application** field. This wrapper works with the *device joindef* files.

5. Click **Add**.
The message line indicates that the trigger record was added.

Test the example Cascade Update

1. Click **Tailoring > Database Manager**.
2. In the **Form** field, type the following: contacts

3. Press **Enter**.
HP Service Manager displays a list of forms matching your search criteria.
4. Double-click contacts.
The contacts.g form opens.
5. Click **Search**.
Service Manager displays a list of all contact records.
6. From the pull-down options menu, select **Modify columns**.
7. Use the drop-down arrow to change the **company** field to **location**.
8. Click **Proceed**.
9. Click in the heading of the location column to sort the list so that all contacts with the same location are shown together.
10. Open a second session of Service Manager.
11. Click **Tailoring > Database Manager**.
12. In the **Form** field, type the following: =location.
13. Click **Search** or press **Enter**. The Search location records form opens.
14. Click **Search**.
Service Manager displays a list of all location records.
15. Double-click on an ACME HQ record in the record list.
The ACME HQ location details form opens.
16. Change the location by typing **ACME CORPORATE** in the **Location** field.
17. Click **Save**.
Service Manager displays the message:
location record updated.
18. Revert to the first session of Service Manager, which is still showing the record list of contact records.
19. Click **Refresh**.
20. All of the ACME HQ locations are now ACME CORPORATE.

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.

Format Control performs many common data tailoring tasks in HP Service Manager, is often complex, and is applied at the form level. If overused, Format Control can affect system performance. Data Policy operates at the table level and achieves many of the same results as Format Control without the complexity and without taxing system resources.

Access Data Policy

Applies to User Roles:

System Administrator

To access Data Policy from the System Navigator

1. Open **System Definition > Tables > *file_name* > Fields > *field_name***.

The **Fields and keys definitions for the application table** form opens.

2. Edit the field properties in the Data Policy section, and then click **Save** to save your changes.

To access Data Policy from the Database Dictionary utility

1. Open the table (File Name) that you want to set Data Policy for in the database dictionary.
2. Select **Data Policy** from the More Actions menu.

The `datadict.g` form opens, containing the selected file.

3. Edit the file properties, and then click **Save** to save your changes.

To search for Data Policy records

1. Open **Tailoring > Data Policy**.
2. Specify search criteria to select the records you want, or leave the form blank to see all records, and then click **Search**.
3. Select a table (file) from the list.
4. Select **Data Policy** from the More Actions menu.

The datadict.g form opens, containing the selected file.

5. Edit the file properties, and then click **Save** to save your changes.

Data Policy expressions

Data Policy rules apply to the GUI presentation of data. This allows some degree of data control into your system without the need to construct numerous different forms and views.

For example:

- In the Invisible or Read Only fields, you specify an expression. If this expression evaluates to true at the time the record is opened, any controls referencing the field in question are set to read-only or visible, as appropriate
- If you want only System Administrators to be able to modify the contact.name field in the contacts file, use this expression: `index("SysAdmin", $lo.ucapex)>0`

The record's database dictionary record defines the fields available on a file's Data Policy record. You cannot add new fields directly to a Data Policy record.

To see your changes to Data Policy you must restart the client by logging out of the system and logging in again.

Data Policy and the object record

Data Policy control for a table can associate and expand with an object record. If there is an associated object, the object offers control over revisions, IR searches, and record displays through SM Manage. If a record in Data Policy has an object associated with it, the fields on the Engine Specifications tab are

grayed out and unavailable. A button appears that takes you to the object record, where you can make changes.

If a Data Policy record does not have an associated object, then the Engine Specifications and SM Manage tabs contain fields you can edit that control revisions and display.

Creating revisions

The revisions option for Data Policy creates a revision database dictionary record, whose name is specified on the Engine Specifications tab. You can create revisions either for an entire set of records or for a single form or record. You must be at the start panel of the application you are working with to create baseline revisions for an entire set of records. For example, you create a baseline revision of all the forms in the Forms Designer from the Forms Designer main menu.

When you create a revision, you create a copy of the record only, not the associated database dictionary record. As a result, any time a field is added to a database dictionary record, you add it to the revision file database dictionary record.

Create a Data Policy revision

Applies to User Roles:

System Administrator

To create a Data Policy revision:

1. Click **Tailoring > Web Services > Data Policy**.
2. Click the **Engine Specifications** tab.
3. Type a revision file name in the **Revision File Name** field.
4. To limit the number of revisions allowed, fill in the **Max # of Revisions** field.
5. Click **Create Revision File** from the More Actions menu.

Example: Create and manage a revision

Applies to User Roles:

System Administrator

The following example steps through the process of creating and controlling a revision. The example uses the location file.

1. Open the location record in Data Policy.
2. Type a revision name on the Engine Specifications tab, for example type LocationRevision.
3. Type the maximum number of revisions allowed on the system for files controlled by this record. For example, type 2.
4. Save the location Data Policy record.
5. Go to the Database Manager and type location in the Form field. A record list opens. Every file in the record list is controlled by the Data Policy record Locations.
6. Select a record, for example ACME HQ, based in Chicago.
7. With the record open, click **Revision > Create Revision** from the More Actions menu. Include any relevant text to help you understand the revision and its purpose. Label this revision one.
8. Make a change to the ACME HQ record and save.
9. Click **Revision > Create Revision** from the More Actions menu. Label this revision two.

The maximum number of revisions has now been reached for the files controlled by the Locations Data Policy record. The next revision you make will eliminate the revision you labeled one.

Revert to a previous revision

Applies to User Roles:

System Administrator

To revert to a previous revision:

1. Open the form you want to use a previous version of.
2. Click **More** or the More Actions icon.
3. Click **Revision**.
4. Click **Revisions > Find Revisions**.
5. Click the revision you want to restore.

6. Click **Revert to this Revision** from the More Actions menu. The system prompts you as to whether you want to save the current version of the record as a revision. It is recommended you do.

Purge revisions

Applies to User Roles:

System Administrator

Purge scripts help administrators with revision maintenance. The `sc.revision.purge.hanging` script purges all revisions that no longer have a parent record because the parent record was deleted or renamed. The `sc.revision.purge` script purges all revisions from the system.

There are two options available when purging revisions.

To search for revisions:

- **Purge All Revision Records** removes all revisions from your system.
- **Purge Hanging Revisions** removes only hanging revisions from your system. A hanging revision is a revision with no associated file or form. For example, if you create a form, create a revision of the form and then delete the original form, the revision remains on the system but with no associated file or form.

Note: You can restore a deleted form from its revision file.

1. Log in as a System Administrator.
2. Click **System Administration > Base System Configuration > Miscellaneous > Purge Hanging Revision Records**.
3. To remove hanging revisions from your system, click **Purge Hanging Revisions Records**.
4. To remove all revisions from your system, click **Purge All Revision Records**.

Data Policy and encryption

Important: If you encrypted any field in your database, ensure that you store the KEY value defined in the `sm.ini` file in a safe place. Without the correct KEY value, you will not be able to decrypt.

When the Data Policy for a table changes, the system checks to see if the encryption status changes. If the status changes from false to true, each record in the file is read, the field is encrypted, and the data

is written back to the file. If the status has changed from true to false, each record in the file is read, the field is decrypted, and the data is written back to the file. The entire process includes reading and updating each record in the file, which could result in performance loss.

Note: Once a field in a table is encrypted, users can no longer search for the contents of the field. The field is not searchable.

Change the encryption key value

You can set the KEY value that encrypts data on your system with the encryptionkey parameter in the sm.ini file. This field must be exactly 8 bytes in length. If you change the KEY value, then every field in every file must be checked and re-encrypted. Changing the KEY value results in a significant performance hit to the system while re-encryption takes place.

To change the encryption key value, perform the following steps:

1. Shut down the HP Service Manager server.
2. Restart Service Manager from the command line using the changeencrkey parameter. For example, `sm -changeencrkey:XXXX` where XXXX is the new 8-byte key.

Starting Service Manager in this way decrypts all encrypted fields using the key defined in the sm.ini file and then re-encrypts those fields using the key specified in the command line parameter changeencrkey. The length of time the conversion takes depends on the size of the database and the number of encrypted tables. You need to update your sm.ini file to the new key immediately after performing this action.

Caution: Encrypting SQL data that is already mapped will increase the size of the data. Therefore, the existing SQL mapping and column definition may not provide enough space to store the whole encrypted value. Be sure to change the SQL data type to accommodate the new column size. If the encrypted value gets truncated, the value can no longer be decrypted.

Use this formula to calculate the new column length:

encrypted_length = (unencrypted_length + 12) * 2.

Change the columns in a record list

Applies to User Roles:

All roles

To change the columns in a record list, follow these steps:

1. Display any record list.
2. Click **More** or the **More Actions** icon.
3. Click **Modify Columns** to display a form that enables you to modify the columns (fields) that are displayed in the record list. Each field appears in a table row that you can edit.
4. Click the column name row to select a new field from the drop-down list.
5. To add a column between two existing column, insert the cursor in an existing row, and then click **Insert Line**. A new row appears above the current row. This creates a new column in the record list.
6. To specify the field to display in a column, select it from the drop-down list in the row.
7. To delete an existing column, insert the cursor by the name of the column that you want to delete, and then click **Delete Line**.
8. To cancel your changes, click **Back**.
9. To return to the default column settings, click **Use Default**.
10. When you finish changing the columns, click **Proceed** on the Toolbar. The record list displays the new column order.

HP Service Manager saves the column changes as a logon preference. Your logon preferences do not affect other users.

You can also create customized views with different column ordering and save them as selectable preferences to apply as needed. For more information, see "[Create customized column views for record lists](#)" on page 1.

Create an alias for your custom CI display name field

Applies to User Roles:

System Administrator

In the out-of-box system, the **device** table uses the **display.name** field to store CI display names. However, your production system may use a custom field to store CI names. For this reason, in the **device** dbdict record, the **sm.device.display.name** field is created as an alias for the **display.name** field. If your **device** table uses a custom field to store CI names, you only need to specify it in the data policy record so that Service Manager automatically recreates an alias named **sm.device.display.name** for it.

Caution: Before you proceed, make sure that your custom CI name field has a character data type, and is configured to be a No Nulls key and not a Unique key. If your custom field does not meet these conditions, it is not available for selection from the Display Field drop-down list in the **device** data policy record.

To create an alias for your custom CI display name field, follow these steps:

1. Open the **device** table from Database Dictionary.
2. From the **More** or More Actions menu, select **Data Policy** to open the data policy record.
3. In the **Display Field** field, select your custom CI display name field from the drop-down list, for example, **mycompany.ci.name**.

Note: When you change the Display Field value, Service Manager removes the alias from the previous field and adds it to the new field.

4. Click **Save** to save the data policy record.
5. Click **OK** twice to save the DBDICT record.
6. Open the **device** DBDICT record again, and verify that an alias named **sm.device.display.name** has been created for your custom field (that is, both the custom field and the alias have the same index number).
7. Execute the **refreshSubscriptionDisplayName** ScriptLibrary to update the CI information in subscriptions.

Note: CI information in subscriptions is derived from the Display Field defined in the **device** data policy record. This step is needed whenever you have changed the Display Field setting.

Define CI Auto Complete Table Columns

Applies to User Roles:

System Administrator

When a web client user enters CI information in the reference modules (such as Incident, and Change), a drop-down table for Auto Complete is displayed, which lists a number of field values of the matching CIs so that the user can easily select the right CI. By default, the following fields (columns) are displayed in

the Auto Complete table: Display Name, CI Identifier, Type, Environment, and Assignment. You can customize this field list.

Note: Auto Complete is always enabled for a field that has a referenced table defined.

To define the Auto Complete Table Columns for CIs, follow these steps:

1. Open the **device** data policy record.

There are several ways to do so. For example, open the **device** dbdict record, and then click **More > Data Policy**.

2. Click the **Auto Complete Table Columns** tab, and then edit the existing list of fields.

The **display.name** field is always the first column in the Auto Complete drop-down table that is displayed to users, even if you delete it from this tab. If you configure an empty or duplicate row, the system automatically removes it when you save your configuration.

3. Save the data policy record.

Specify a referenced table for a field

The Data Policy for a table can define a referenced table for a field in the table. A referenced table is the table that is referenced by the current field. From a business logic perspective, the current field must be used to store the same data as the unique key field of the referenced table. For example, in the out-of-box **cm3r** Data Policy record, the **device** table is defined as the referenced table of the **assets** field (the Affected Configuration Item field). Once you have specified a referenced table, Service Manager creates a link that points from the current field to the unique key field of the referenced table.

Furthermore, the referenced table must have a Display Field defined. The current field displays the values of the display field, instead of values of the unique key field of the referenced table. For example, the Affected Configuration Item field in a Change record displays the CI display names (stored in the display.name field), instead of the CI Identifier values (stored in the logical.name field).

This functionality automatically updates the CI data in the reference modules (such as Change, and Incident) when a CI name is updated in the Configuration module so that data integrity is maintained across the modules.

To specify a referenced table for a field, follow these steps:

1. Click **Tailoring > Data Policy**.
2. Open the corresponding Data Policy record. For example, for the **affected.item** field in the **cm3r** table, open the **cm3r** Data Policy record.
3. Select the **Field Settings** tab.
4. Select the field, and select a value from the drop-down list in the Referenced Table column. For example, select **device**.

Note: Currently, only the **device** table can be used as a referenced table.

5. Click **Save** to save the Data Policy record.

In this example, Service Manager creates a link between the field and the **logical.name** field of the **device** table, if the link does not already exist. In the link record, the **logical.name** field is the target field.

Next, you need to specify a display field for the referenced table. This is the field of the referenced table whose values will be displayed in the current field. The current field derives data from the display field based on the unique key field of the referenced table. For details, see "[Specify a display field for a referenced table](#)" below.

Specify a display field for a referenced table

You need to define a Display Field in the Data Policy (datadict) record of a referenced table. For example, the out-of-box system defines the **device** table as the referenced table for the **assets** field of the **cm3r** table, and defines the **display.name** field of the **device** table as the Display Field. Because of this configuration, Affected CIs in a Change record are retrieved according to their CI Identifier values and displayed as their display names.

Note: A referenced table must have only one field as the unique key. Currently only the **device** table can be set as a referenced table.

To specify a display field for a referenced table, follow these steps:

1. Click **Tailoring > Data Policy**.
2. Open the Data Policy record for the referenced table. For example, open the **device** Data Policy record.
3. In the **Display Field** field, select a value from the drop-down list. For example, select **Configuration Item Name - display.name**.

A display field has an alias defined in the dbdict record. When you change the Display Field value, Service Manager removes the alias from the previous display field and adds it to the new field.

Note: A display field must meet the following conditions:

- Has a character data type
- Is not a Unique key field.
- Is a No Nulls key field

4. Click **Save** to save the Data Policy record.

Data validation

The validity table enables you to define a field's valid values in a list of values, a list of ranges, a secondary file, and customized RAD subroutines. These values are then used for validating operator-entered data and for lookup processing (similar to HP Service Manager Find and Fill). These various validity definitions are consolidated in one file, resulting in easy maintenance. This can be of significant benefit when multiple forms access one database.

Validation routine

The value of a field is validated against a list of values (for example 10, 20, 30, or 40), a list of ranges (for example 50 through 60), a secondary file (similar to the Format Control secondary file query) and customized RAD subroutines. The fields within a file are validated in a particular order (for example fieldx, fielda, fieldg), in alphabetical sequence (for example fielda, fieldg, fieldx) or according to a field validation sequence number. The default processing of the validation routine validates the value of a field against the definitions in the validity table. If the current value of a field is invalid, a summary list of valid values and ranges is presented to the user. If no values or ranges are defined for the field, a list of records from the secondary file is presented. The user can then correct the invalid value and

continue processing; return to the calling application; or force the application to accept the invalid value. The validity table also enables you to override this processing and execute customized RAD code.

User-defined processing statements

After determining that a field contains a valid value or range, user-defined processing statements are executed. Validity processing continues to the next field in the validity definition table until all specified fields are validated.

Each validity table definition contains four levels of definition:

- Field Summary
- Secondary File Query Information
- Values Summary
- Range Summary

Maintenance procedures

When it is necessary to differentiate the maintenance procedures of these different levels, the order of the documentation is Field level, Secondary Queries level, Values level, and Ranges level. For instance, the section on updating shows how to update Validation Records at the Field Level, the Value Level, and then the Range Level. All applications and procedures described refer to a unique maintenance application that specifically supports validity table maintenance. If you access any of the validity table databases with any other application (such as Database Manager), you must ensure that all data is properly structured in order for the validity definitions to work correctly at run time. Hewlett-Packard recommends that you always follow the documented Validity procedures.

Accessing the validity file

Following are the ways you can access the validity file.

- **Access Validity from a menu**

1. Click **Tailoring > Tailoring Tools > Data Validation**. The Validity Table Specifications form opens.
2. Click **Search**. A list of validity maintenance records opens.

- **Access Validity through Database Manager**

1. Click **Tailoring > Database Manager**.
2. Type **validity** in the Form field, and then click **Search**. A list of the validity form records in your database opens.
3. Double-click a validity Format Name record, and then click **Search**. The Validity Table Specifications form opens.
4. Click **Search**. A list of data validation records opens.

- **Access Validity from the command line**

1. Click **Command** in the System Administrator's home menu.
2. Type `validity` in the command line.
3. Press **Enter**. The Validity Table Specifications form opens.
4. Type a field name or a file name in the form, and then click **Search**. The requested form opens.

Deleting validity record components

You can delete entire validity records or parts of records using several different methods.

- Delete the entire validity record and all its components.
- Delete a value or range definition from an individual record.
- Delete value and range definitions from a table format.

Invalid application names

If you have entered an invalid application name in the **Alt Application** field in the **Alternate Application** tab, a record list of valid applications opens and the following message appears in the status bar: *The Application Name is not valid*. Select one from the list. Double-click on the correct application to replace the invalid entry.

Invalid files and form names

If any element of the **Files/Formats** field is not a valid database dictionary name or a valid form name, the following message appears in the status bar: `The name <name > is not a valid dbdict or format.` Search for database dictionary records. A record list of valid file names is displayed. Double-click on the correct file to replace the invalid entry.

If you have entered an invalid form name in the **Files/Formats** field and want to view a record list of valid forms, open the More Actions menu and select **Display All Formats**. A record list of valid forms opens. Double-click on the correct form to replace the invalid entry. If the form you select is not associated with a file, you must respond by indicating the file with which the form will be associated at run time.

Note: The file associated with the form must contain the **Field Name**. If it does not and the first element in the list is being validated, you are prompted to select a valid field name. If the second or greater element is being validated, you are returned to the original record with an error message.

To switch back to the list of files, select **Display Table List** from the More Actions menu.

Invalid field names

If you enter an invalid field name, the following message appears in the status bar: *The field name <field name > is not valid.* A list of valid field names for the file you have entered opens. Select a valid field from the list. The invalid entry is replaced by the field you have selected.

Printing validity definitions

A detailed report of a validity record has four main sections that display all the information contained in the record and in the detailed summary forms:

- Field specifications
- Query specifications
- Values specifications
- Range specifications.

Note: Any scalar field or array element that is greater than 60 characters in length is printed in its entirety with the use of continuation lines. All elements of all arrays are also printed. All continuation lines start with: ***

Invoking validity table processing

You can invoke validity table processing by calling the RAD application **validate.fields** in the **Subroutines** process of **Format Control**. Invoke validity table processing to achieve the following two goals.

- Validate the fields in a record during record processing.
- Allow look-up processing during record editing (similar to HP Service Manager Find and Fill capability).

Look-up processing

Calling the validity routine while a record is being edited simulates the Find and Fill functions. This gives the user the ability to determine the valid values or ranges of a field before attempting to process a record.

You can execute the validity look-up routine from these three places.

- Format Control Additional Options
- Format Control subroutines
- Displayoptions

Format Control processes

There are eight major functional processes in Format Control that define the actions to be taken on a record. Each of these processes has a separate form within the Format Control Utility.

Main information — This form is the entry point in Format Control and has several functions.

- Initializes fields or variables that are later used in the Format Control record. Initialization expressions are the first operation performed for each evaluation of a Format Control record.
- Initially displays a value in a field when the record itself opens.

- Names special record list and initial query forms to use, sets up default sort sequences, and runs scripts.

Forms — This section in Format Control has two main functions:

- Specify alternate forms to display a list of records.
Note: This option cannot be used when the Record List (split screen) functionality is enabled.
- Specify alternate forms to display a single record.

The formatcontrol option allows you to display the information using different Service Manager forms based on conditions evaluated at run time. You can specify either QBE Forms to display a list of records or Alternate Forms to display a record. You may want to specify different lists according to the user role or capabilities.

By specifying alternate forms, whenever the condition is met, the user will have an option available called "Alternate Forms" where the user can select the form to use to display the information. Additional forms are useful to display different views such as financial or security information when the condition specified is met.

Queries — This process enables you to extract information from a file other than the primary file in order to perform calculations and validations and to report on information from more than one file.

Calculations — This process enables you to perform calculations on currently available fields or variables. The fields needed for calculations may be variables, fields in the primary file, or fields in any other secondary files that may have been queried.

JavaScript — This process enables you to call JavaScripts from Format Control.

Validations — This process enables you to set up a logical expression for checking data in fields or variables on the form. The validation expression you set up must evaluate to true upon the desired edit function for it to be successful. If the expression does not evaluate to true, a validation message opens, and the specified operation fails.

Subroutines — This process enables you to call RAD routines from Format Control.

Additional options — This process enables you to define menu options you want available to users on any form associated with the particular Format Control record. You can use Format Control to set up a menu option called Validity Lookup. This option performs a validity check on the field of focus in the form for which validity specifications have been defined. Invalid fields are highlighted and a message is displayed in the status bar giving details on the type of error incurred.

Note: This feature calls RAD subroutines and is available in Database Manager only.

Privileges — This process enables you to use security to control database options available to the user. If the contents of a field evaluate to true at processing time, the corresponding button is available to the user.

Displayoptions

You can create a Validity lookup option in the More Actions menu by creating a displayoption record. For detailed information on creating displayoptions, refer to the Display application help.

Use the following values in the **displayoption** record:

Field	Value
Default Label	Validity Lookup
Condition	true
RAD Application	validate.fields
Separate Thread?	true

Use the following values in the **displayoption** record:

Name	Value
name	cursor.field.name()
second.file	\$L.filed
cond.input	val("true", 4)

Display value summary details

Applies to User Roles:

System Administrator

You can display the details of a single entry in the values summary array, or view the details for all the entries displayed in a scrollable form.

To display the details of a single entry in the value summary array:

1. Place your cursor on a line in the value summary array.
2. Select **Options > Edit Line**. The details for the entry are displayed.

To display the details of all entries in the value summary array:

1. Place your cursor on a line in the value summary array.
2. Select **Options > Display All Values** in a validity record. Details of all the entries in the value summary array are displayed in a scrollable list.
Note: You may edit or sort the contents of this record as you would any table in Service Manager.

To display the typical table editing controls in the tool tray:

Select **Options > Edit Table**.

Display range summary details

Applies to User Roles:

System Administrator

You can display the details of a single entry in the range summary array, or view the details for all the entries displayed in a scrollable form.

To display a single entry in the range summary array:

1. Place your cursor on a line in the range summary array.
2. Select **Options > Edit Line**. The details for the entry are displayed.

To display the details of all entries in the range summary array:

1. Place your cursor on a line in the range summary array.
2. Select **Options > Display All Values** in a validity record. Details of all the entries in the Range Summary array are displayed in a scrollable list.
Note: You may edit or sort the contents of this record as you would any table in Service Manager.

To display the typical table editing controls in the tool tray:

Select **Options > Edit Table**.

Create validity table definitions

To create a new validity table definition, complete the appropriate records in the following order:

1. Add field-level definitions.
2. Add value definitions.
3. Add range definitions.
4. Validate the validity definitions.

Add field-level definitions

1. Click **Tailoring > Database Manager**.
2. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.

3. Double-click **validity.summary**, and then click **Search**.

The Validity Table Specifications summary form opens.

4. In the **Field Name** field, type the name of the field in which you want to create a validation record.
5. In the **Files/Formats** field, type the name of the file or files in which this field is found.
6. Click **Search** to search for records that may already exist with this field.
7. If no record exists, click **New** to add your new record.

The following message appears in the status bar: Data Validation record added.

8. Complete the remaining fields in the form (including the other tabs) with the exception of the Values Summary and Range Summary arrays.
9. Click **More** or the More Actions icon, and select **Validate** to validate the information you have entered in the form.

If you have entered an invalid value in the form, a message appears in the status bar specifying the error.

10. Click **Save** when you have successfully validated your record.

A message appears in the status bar, stating that the Data Validation record has been updated.

Add value definitions

1. Click **Tailoring > Database Manager**.

2. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.

3. Double-click **validity.summary**, and then click **Search**.

The Validity Table Specifications summary form opens.

4. Put the cursor in the first line of the Values Summary array.

5. Click the Options menu, and choose **Edit Line**.

The Validity Value Detail Specification form opens.

6. Complete the record as necessary, and then click **Add**.

Note: It is possible that the same range could be allowed under multiple conditions within the same validity record; therefore, the system does not check for duplicate field values.

Important: If the character field starts with a < or > symbol, you must insert a blank space before it. These symbols are dropped by the HP Service Manager parser, unless preceded by a blank space.

The record is added and the buttons in the tool tray change. The following message appears in the status bar: Range/Value successfully added.

7. Click **Back** to return to the validity record.

The system combines the values you entered in the Min Desc and Max Desc fields with the relational operator you specified to produce the string that appears in the Values Summary array.

8. Put the cursor in the next blank line of the Values Summary array and repeat the above steps until you have established all the ranges for your field.

9. Click **Save** to add the record to the validity file.

Add range definitions

1. Click **Tailoring > Database Manager**.

2. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.

3. Double-click **validity.summary**, and then click **Search**.

The Validity Table Specifications summary form opens.

4. Put the cursor in the first line of the Range Summary array.

5. Click **More** or the More Actions icon, and then choose **Edit Line**.

The Validity Range Detail Specification form opens.

6. Complete the record as necessary, and then click Add.

For additional information on writing conditions and expressions, see information on system language in the related topics.

Note: It is possible that the same range could be allowed under multiple conditions within the same validity record; therefore, the system does not check for duplicate field values.

Important: If the character field starts with a < or > symbol, you must insert a blank space before it. These symbols are dropped by the HP Service Manager parser, unless preceded by a blank space.

The record is added and the buttons in the tool tray change.

7. Click **Back** to return to the validity record.

The system combines the values you entered in the Min Desc and Max Desc fields with the relational operator you specified to produce the string that appears in the Range Summary array.

8. Put the cursor in the next blank line of the Range Summary array and repeat the above steps until you have established all the ranges for your field.

9. Click **Save** to add the record to the validity file.

Validate the validity definitions

The validation routine ensures that field, file, form, and application names referenced in the definition are valid. It also prompts for a data type if you have left the **Field Type** field blank.

To validate the validity definitions:

1. Open a validity record.
 - a. Click **Tailoring > Database Manager**.
 - b. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.
 - c. Double-click a validity Format Name record, and then click **Search**.

The Validity Table Specifications form opens.
 - d. Click **Search**.

A list of data validation records opens.
 - e. Select a record.
2. Click **More** or the More Actions icon, and choose **Validate**.

If the validation routine detects an invalid entry, a message appears in the status bar and a form opens with a list of field names to select the correct entry.

If all the record entries are valid, a message appears in the status bar stating that the validity record entries are valid.

Delete an entire validity record

1. Open a validity record.
 - a. Click **Tailoring > Database Manager**.
 - b. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.
 - c. Double-click a validity Format Name record, and then click **Search**.

The Validity Table Specifications form opens.

- d. Click **Search**.

A list of data validation records opens.

- e. Select a record.

2. Click **Delete**.

The following prompt is displayed: Are you sure you want to delete this record?

3. Click **Yes**.

Delete a single value or range definition

1. Open a validity record.

- a. Click **Tailoring > Database Manager**.

- b. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.

- c. Double-click a validity Format Name record, and then click **Search**.

The Validity Table Specifications form opens.

- d. Click **Search**.

A list of data validation records opens.

- e. Select a record.

2. Place the cursor in the line containing the value or range definition you want to delete.

3. Click **More** or the More Actions icon, and choose **Edit Line**.

4. Click **Delete** in the detail form.

The following prompt is displayed: Are you sure you want to delete this validity detail?

5. Click **Yes**. You are returned to the validity record.

6. Click **Save**.

Delete value or range definitions from a table

1. Open a validity record.
 - a. Click **Tailoring > Database Manager**.
 - b. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.
 - c. Double-click a validity Format Name record, and then click **Search**.

The Validity Table Specifications form opens.
 - d. Click **Search**.

A list of data validation records opens.
 - e. Select a record.
2. Click **More** or the More Actions icon and choose **Display All Values** or **Display All Ranges**.

A Validity Value Specifications form or Validity Range Specifications form opens.
3. Access the table editing controls. Click **More** or the More Actions icon, and choose **Edit Table**.
4. Put the cursor anywhere in the record you want to delete, and then click **Delete**.

Warning: Make sure this is the record you want to delete. When you click Delete, you are not prompted to confirm the command. Instead, the record is immediately removed from the table.
5. Repeat the process with any other value or range definitions you want to delete.
6. Click **End** to exit the table edit mode.
7. Click **Back** to return to the validity record.
8. Click **Save**.

Print a detailed report of an entire validity record

1. Click **Tailoring > Database Manager**.
2. Type **validity** in the Form field, and then click **Search**.

A list of the validity form records in your database opens.

3. Double-click a validity Format Name record, and then click **Search**.

The Validity Table Specifications form opens.

4. Click **Search**.

A list of data validation records opens.

5. Select a record to open it.

Note: To generate a detailed report, the validity record must contain values or range definitions.

6. Place the cursor in a *summary* field containing a value.

7. Click **More** or the More Actions icon, and then choose **Edit Line**.

A detail specification record opens.

8. Click **File >Print** or the Print icon.

Validate fields during record processing

During record processing, the fields in a record can be validated one at a time or in a particular order.

Note: To invoke validity table processing, see the related topics. You must validate each validity table definition before using it to validate data records.

1. Access the Format Control record for the form which you want to invoke validity table processing. If a record does not already exist, you must create one. For details on accessing and using Format Control, see the related topics.
2. Display the Subroutines process.
3. Enter **validate.fields** in the **Application Name** field.

Note: The logical values in the *Add*, *Update*, and *Before* fields are examples only. For information on how to decide when Format Control is processed, see the related topics.

4. Enter the applicable parameters for **validate.fields** in the **Name** and **Value** fields using the information from the following table:

Name	Value	Description
second.file	\$file	Executes all validity table records (all fields) defined for the file being processed.
name	<field name>	<p>Executes only one validity table definition (one field) defined for the file being processed.</p> <p>Note: You can repeat this type of definition in the Format Control subroutines section as many times as necessary. However, it is more efficient to call the subroutine once and pass a list of the fields to be validated.</p>
names	<list of fields> or \$<variable>	<p>Executes a specific list of validity table definitions defined for the file associated with the form.</p> <p>For example, define a \$ variable in either the Initializations or Calculations process of the Format Control record as: \$fieldlist={"reason", "risk", "planned.start"}</p> <p>In the Subroutine call, specify the Value input as:</p> <p>Name: names Value: \$fieldlist</p> <p>Note: You can use any variable name you choose, however, you are responsible for ensuring that your variable name does not corrupt a variable used by the application validate.fields.</p>
text	current.format () or <form name>	Executes a specific list of validity table definitions defined for the form just displayed to the user.

5. Enter a message in the **Error Message** field.

The recommended error message is: The original record is displayed.

If a message is not defined, an unfriendly error message is issued when returning to the data record.

6. Click **Save**.

Create a validity lookup option in Format Control

Access the Format Control record for the form in which you want to invoke validity table processing. If a record does not already exist, you must create one. For information on how to access and use Format

Control, see the related topics.

1. Click **Tailoring > Format Control**.
2. Type a **Name** and then click **Search**.

Service Manager opens a list of Format Control records.

3. Click the record you want to update to open it.
4. Click **More** or the More Actions icon and choose **Additional Options**, or click **Add Options** in the form.
5. Enter the following field values:

Field	Value
Condition for Option	true
Option	Validity Lookup
Application	Application validate.fields
Error Message	The original record is displayed. Note: This is the recommended message. If this field is left blank, you will receive an unfriendly error message when you return to the data record.
Reset on Return	true

6. Enter the applicable parameters for **validate.fields** in the **Name** and **Value** fields using the information from the following table:

Field	Value
name	cursor.field.name()
second.file	\$file
cond.input	val("true", 4) Note: The data passed to the Boolean parameter must be a type 4 (hence the VAL())

Field	Value
	statement for the Value). As an alternative, a \$variable could be defined in the <i>initialization expressions</i> (for example \$flag=true) and then passed to the subroutine.

7. Click **Add** to add a new Format Control record.
8. Click **Save** to save your updates.

Call validate.fields from Format Control

1. Click **Tailoring > Format Control**.
2. In the **Name** field, type the name of the form for which you want to invoke validity table processing.
3. Click **Search**.
4. If a Format Control record does not exist for your form, create one. Click **New**.
5. Click **Subroutines** in the form, or select it from the More Actions menu.
6. Enter the appropriate parameters for **validate.fields** in the **Name** and **Value** fields using the information from the following table:

Field	Value
name	cursor.field.name()
boolean1	false
second.file	\$current.file.variable
cond.input	true

7. Click **Save** to save changes to an existing record.

Caution: If you are attempting to add a new record from an existing record, make sure that you do not click Save because doing so will replace the existing record with the new record. If you are

adding a new record, click Add.

8. Click **Add** to add a new record.

Validity validation rules

- Each entry must have at least one entry in the **Files/Formats** field.
- Each entry in the **Files/Formats** field must be a valid database dictionary name or form name.
- The **Field Name** must be defined in each database dictionary listed in the **Files/Formats** array or to the database dictionary associated with the form.
- The data type of the **Field Name** must be the same for each file listed in the **Files/Formats** array or for the file associated with the form.
- The **Unique ID** is a user-provided value that uniquely identifies the record.
- The unique key of the validity table contains the following fields: **field.name**, **sequence**, and **filename**. Each combination of the Field Name, Sequence, and each element of the Files/Formats array in the validity record must also be unique within the validity table. This check is performed before the record is added or updated. If duplicates are found, the add/update action is not performed, and the following message is displayed: This record contains an invalid duplicate key.
- The value in the **Alt Application** field must be a valid RAD application.
- The **Secondary File Query Filename** value must be a valid database dictionary name.
- The **Lookup Fields** must be defined to the **Secondary Query Filename** and their data types must be the same as the Field Name.
- The **Val Fields** must be defined to the **Files/Format** file and their data types must be the same as the Field Name.
- The Record list format must be a valid form name.
- The value in the **No Recs Option** field must be the word **Proceed**, or **Return**. If neither is selected the system defaults to Proceed when the record is validated.

Global lists

The global initer and global lists work together to generate lists for the HP Service Manager system.

Global lists are stored in the system and are available to any application. Use the global lists tool to add a combo box to a form that displays a list of values, based on a table in your database. For example, set your combo box **DisplayList** value to *\$G.time.zones*, and the system displays a drop-down list of time zones from the *tzfile* file.

The global initer builds lists of records from the database. Generally, these lists are used to fill the drop-down lists in combo boxes on display forms. Examples of common lists include a list of all operators in the system, all Incident categories in the system, or all assignment groups on file.

The Startup Lists global list stores all the global lists where the *build.startup* field equals true. The global list is in the *listrepository* file. During logon, the system checks the *listrepository* entry and builds all global lists in this file. If the global list file changes, the Startup Lists global list is marked as expired and rebuilds the next time the *lister* background process runs. This updates the *listrepository* record.

Note: Since the *lister* background process does not run continually, there may be a slight delay between the time a global list changes and when it appears in the *listrepository* record.

The global initer consists of two parts:

- A server side element which generates and packages lists according to a user-defined schedule.
- A client-side requester which queries lists from the server and places them into client-side memory.

Lister

To build a dynamic list, generate a list definition block. The background scheduler (*lister*) wakes up every sixty seconds to search for obsolete lists and uses list definition blocks to generate new lists, which are then stored in the list repository.

Once *lister* has built a list on the server, multiple clients can request and share the same list without each client having to build its own copy. This method of building lists improves system speed and helps server-side performance.

Note: *Lister* variables can be defined for *scalar* type fields only and not for array fields.

Global initer considerations

- For performance considerations of the Service Manager server, a global list should contain no more than 2800 individual elements. Exceeding this limit may degrade the server performance.
- The total size of the **globallist** record, which contains the full lists, cannot exceed the logical file limit set for the system.
- The sort field must be a key on the table which is used to generate the list.
- No matter what you set as the reset time, the server initer can only refresh the lists once every time its scheduler wakes up. Thus, if you have a list with reset time = 5 seconds and the scheduler runs every 60 seconds then the list will actually refresh every 60 seconds.

Returning a list to your client

Use one of the following methods to return a particular list, or set of lists, to an HP Service Manager client:

- Build a list on startup.
- Configure the lists with Format Control.

Configuring lists with Format Control

Use Format Control to ask for a list only when a specific form is displayed. This procedure is most efficient when the form defined is not displayed frequently.

Access a list record

1. Click **Tailoring > Tailoring Tools > Global Lists**.
2. Enter optional search criteria, and then click **Search**.

A global list of definitions opens.

3. To create a new list, click **Add**.

4. If you edit an existing list, click **Save**.
5. Click **OK**.

Bind a list to a form

Lists are bound to the forms they serve. The global variables are applied in Forms Designer.

1. Click **Tailoring > Forms Designer**.
2. Type the name of the form you wish to modify in the **Form** field of the Forms Designer dialog box.

For example, type: `contact.detail.subform`. This is the name of the subform used by `problem.template.open.g`, `problem.template.update.g`, and `problem.template.close.g` to display contact information.

3. Click **Search**.

The contact detail subform opens.

4. Click **Design**.
5. Add a combo box with a label called **Time Zones** to the form, using the following values:

Field	Value
Input	time.zones
ValueList	\$G.time.zones
DisplayList	\$G.time.zones

6. Wait for lister to refresh the time.zones global list, or go to the list definition panel (**Tailoring > Tailoring Tools > Global Lists**) and reset the refresh time.

Note: If you leave the DisplayList value blank, the contents of the ValueList will be used for both lists.

Build lists on startup

Once you have created a global list with the **Build List on Startup** option selected, Service Manager automatically rebuilds the global list when users log in. This method moves the list to every client machine, whether the user requires it or not.

Caution: If you have created a Global List variable without the **Build List on Startup** option enabled, you need to manually rebuild the contents of the variable to make the variable ready for use. For example, if you create a global list \$G.test without selecting the **Build List on Startup** option and you do not rebuild it manually, when you go to the RAD Debugger and type `d $G.test`, the RAD Debugger returns a NULL value.

Tip: To avoid memory problems, you are recommended to enable the **Build List on Startup** option only for global lists that have less than 250 items.

Use the following procedure to set the **globallists** record to build on startup.

1. Click **Tailoring > Database Manager**.
2. Type **globallists** in the Table field of the Database Manager dialog box, and then click **Search**.
3. In the List Name field, type the name of the list you want to view. For this example, type **time.zones**.
4. Click **Search**.

The time.zones file opens.

5. Select **Build List on Startup?**.
6. Click **Save**.
7. Open an incident record to view the new field.

Configure lists with Format Control

1. Determine the base name of the form, which you added to your list.
2. Click **Tailoring > Format Control**.

3. Type the *base* name of the form in the **Name** field.

For this example, create a new record for the form `problem.template.update`.

Note: If you add a combo box to a subform, attach the Format Control record to any base form using that subform. (Do not attach the Format Control record to the subform itself.)

4. Click **Search**.
5. If a Format Control record does not exist for your form, create one. Click **New**.
6. Click **Subroutines** in the form, or select it from the More Actions menu.
7. Select **Show Expanded Form** from the More Actions menu.
8. Scroll to the first empty section and set up a Subroutine call using the following values:

Parameter	Value
Application name	apm.global.initer
Names/Values	<i>See next table</i>
Before	true
Display	true

Names	Values	Definition
names	{"time.zones"}	Array of lists to be moved
name	create	Action to perform (create lists)

9. Click **Save** to save changes to an existing record.

Caution: If you are attempting to add a new record from an existing record, make sure that you do not click Save because doing so will replace the existing record with the new record. If you are adding a new record, click Add.

10. Click **Add** to add a new record.

This Format Control record returns the **time.zones** list to all incident records displaying the Contact Detail tab, regardless of mode.

Calling the **apm.global.initer** several times from Format Control will not slow the system appreciably. The system automatically keeps track of what lists are already on the client and will not recover them unnecessarily; however, if you want the Global Initer to reassemble the list regardless, invoke it with the refresh parameter (rather than the create value listed in the above). called with the refresh parameter, the Global Initer returns all the lists passed to the names array, even if they are already in client memory.

Determine if a scheduler record exists

1. Click **Tailoring > Database Manager**.
2. Type `schedule` in the **Form** field of the Database Manager dialog box.
3. Press **Enter** or click **Search**.
4. Open `schedule` from the record list (double-click the record or click Enter). A blank Schedule File record is displayed.
5. Type `List Runner` in the **Name** field.
6. Click **Search**.
If a record is found, it will open for viewing and editing.

Determine if the regen cycle is realistic

1. Click **Tailoring > Database Manager**.
2. Type `apm.global.list.entry` in the Form field of the Database Manager dialog box.
3. Click **Search**.
A blank record from the **globallists** file is displayed.
4. Click **Search**.
5. Select the list you want to regenerate from the Record list.
6. Check the value in the **Regen Every** field.
Use a short time interval for lists that change frequently and a longer interval for lists that change only occasionally.

Determine when a list was last regenerated

1. Click **Tailoring > Database Manager**.
2. Type `apm.global.list.entry` in the Form field of the Database Manager dialog box.
3. Click **Search**.
A blank record from the globallists file opens.
4. Click **Search**.
5. Select the list you want to regenerate from the Record list displayed.
6. Check the value in the **Expiration** field to see when the list was last regenerated.

Move a Global List to a client

To move a global list into client side memory, you call a routine called **apm.global.initer**.

This routine accepts two parameters:

- An array containing the name of the lists that you want.
- A keyword, either create or refresh.

When **apm.global.initer** is invoked with the refresh parameter, it collects the newest version of its list from the server, whether that list already exists in the client's memory space. If invoked with the create parameter, it collects a list from the server only if a copy does not already exist.

If you are programming in RAD, you can invoke **apm.global.initer** directly from your code. If you are using the Global Initer to fill combo boxes for display on a form, you can invoke the Global Initer as a subroutine using Format Control for the format in question.

Regenerate all lists

If none of the troubleshooting steps to this point have succeeded in regenerating your list, you should regenerate all the lists in the system. This procedure makes all the lists in the system obsolete and enables you to update the entire file.

1. Click **Tailoring > Database Manager**.
2. Select **Administration Mode**.
3. Type **apm.global.list.entry** in the Form field.
4. Click **Search**.

A blank search form from the globallists file opens.

5. Click **Search** to display a list of lists.
6. Click **Mass Update**.
7. Click **Simple Update**.

A blank form opens. This form is identical in appearance to the *lister* record, but contains different option buttons.

8. In the **Expiration** field, change the date to any date in the past. You can either type the date and time, or select a date from the calendar.
9. Click **Execute**.

The expiration date of all the lists in the globallists file is reset.

10. Return to the home menu.
11. Type `*aapm.server.initer` in the command line.
12. Press **Enter**.
13. Log off HP Service Manager, and then log back on.

All the lists in the system are regenerated whether they are actually obsolete.

Regenerate obsolete lists

If you perform all the troubleshooting steps and the system still does not display the most current version of your list, regenerate all the obsolete lists in the system. For a large system with numerous obsolete lists, this procedure may take a few minutes.

You can change the expiration date of a list to a date in the past prior to regenerating, in order to force regeneration.

Note: If the expiration date of your list has not passed, the list will not be flagged as obsolete, and therefore not regenerated.

Regenerate obsolete lists

1. Type `*aapm.server.initer` in the command line.
Running this application will force the system to regenerate all obsolete lists.
2. Log off HP Service Manager, and then log back on.

Start the server side component of the global initer

The server side component of the Global Initer consists of a routine called `apm.server.initer`. A background scheduler called `lister` has been defined to run `apm.server.initer`.

If `lister` did not start with the system:

1. Click **System Status** on the System Navigator.

The System Status utility opens.

2. Click **Start Scheduler**.
3. Double-click **lister.startup**.

Every sixty seconds, `lister` checks and refreshes any global lists that have passed their reset time.

See a complete list of all the global lists on my system

1. Click **Tailoring > Database Manager**.
2. Type `globallists` in the Table field of the Database Manager dialog box.
3. Click **Search**.

A blank search form from the `globallists` file is displayed.

4. Click **Search**.

A list of all the global lists opens. You can scroll through the complete list of records.

Verify lister status and configuration

1. Click **System Status** in the System Navigator.
2. Check to see if lister appears in the list of processes running on your system.
3. If lister is not on the list, click **Start Scheduler**.
4. Double-click **lister.startup** in the Startup Record list to start the program.

Note: Double-click **startup** to start all processes.

View changes to a list

1. Click **Tailoring > Database Manager**.
2. Type `globallists` in the Table field of the Database Manager dialog box.
3. Click **Search**.

A blank record from the *globallists* file is displayed.

4. Click **Search**.
5. Select the list you want to view from the Record list.
6. Check that the changes appear.

Record tag localization utility

The record tag localization utility allows for the localization of fields used to identify records in a given table. Users may use this feature to translate such tags so that they display in their appropriate translations when a user logs into a localized version of Service Manager. This allows the fields to appear in the language of the session associated with the user rather than in their true saved form.

Localization process

Before you begin the localization process, make sure that all desired languages are activated in the language table.

The localization process consists of four steps:

- Enabling the table for tag localization by modifying the corresponding data policy record.
- Updating all corresponding message records to contain the correct translations.
- Creating global lists for the table that utilize these message records.
- Invoking the localization utility to define the fields that will use these new global list values.

Enable a table for tag localization

To enable a table for localization:

1. Access the data policy record for the table.
2. On the General tab, check the Localized Table check box, which then makes the Localized Tag Field available.
3. Set Localized Tag Field to the name of the field whose values you wish to localize. This provides the localized record tag.
4. Save your changes.

For example: Use the following inputs to enable localization for the name field in the category table.

Field Name	Value
Localized Table	checked
Localized Tag Field	name

Wait for the lister process to rebuild the localizedTables global list. When this is done, view the global list to ensure that it includes the table you enabled for tag localization.

Access and localize message records

Once localization is enabled, message records for each active language are created for every existing record in the table. If English and French are enabled for your system, then English and French message records exist for every category record once the category table is marked for localization. These messages contain the following values:

- **Class:** local:<TABLE>, where <TABLE> refers to the name of the table.
- **Message ID:** <UNIQUE>, this refers to the values that make up the unique key of the record.
- **Text:** <LOCALIZED VALUE>, this refers to the value of the localized field. This is the value that needs to be modified to contain the correct translation.

To access and localize these message records:

1. Access the table itself and bring up a record.
2. Click **More** or the More Actions icon and then select View Localized Record Tags. This lists every message that corresponds to the record.
Note: The View Localized Record Tags action is available only for enabled tables that utilize the db.view displayscreen. To make this option available for a table that uses a different displayscreen, you need to create a similar displayoption record for that displayscreen.
3. Drill into each message record and modify the Text field so that it contains the correct translation for the localized field.
4. Repeat these steps for the remaining records in the table.

Create localized global lists

The record tag localization utility makes it possible to localize a query-based list. You can use this utility to localize lists that are defined by a query.

To localize a query-based global list:

Note: You must log out and then log back in for the global list change to take affect.

1. Create or access an existing global list record that is based on the Limiting SQL field.
2. Check **Use Localized Values?**.
3. Ensure that List Variable and Display Variable are set to different variable names.

4. Save your changes.
5. Wait for the lister process to build the list.

Example: use the following inputs for Global List Definition:categories.active to return a list of all active categories.

Field Label	Value
List Name	categories.active
Build List on Startup?	checked
List Variable	\$G.categories.active
Display Variable	\$G.categories.active.local
Limiting SQL	active=true or active=NULL
Use Localized Values?	checked

After the list is built, the localized lists for all active languages display in the table under the Value List field on the Global List Definition form.

These lists are based on the contents of the message records that were updated in Accessing localizing message records ("[Accessing and localizing message records for a codes-based global list](#)" on [page 107](#)). The display variable will be bound to the display list for the language associated with the user session.

Invoking the localization utility

Once the global lists are defined and built, you may invoke the localization utility to define the format fields to utilize the global lists.

Note: To access the localization utility, click **Tailoring >Localization Utility**.

The localization utility is comprised of the following fields:

- **Table Name:** the table containing the fields you wish to localize.
- **Field Name:** a list of fields to localize.
- **Global List:** a list of global lists to be bound to those fields.
- **Formats:** a list of formats on which to apply each field-global list association.

Example: Use the following inputs to bind the Category (*category*) field in the probsummary table to the *categories.active* global list created in "Create localized global lists" on page 104. This example is set up to take effect on the *IM.update.incident* format.

Field Label	Value
Table Name	probsummary
Field Name	Category
Global List	categories.active
Formats	IM.update.incident

After defining your requirements, you must save the record and click **More** or the More Actions icon and then select Run Localization Utility to run the utility.

The localization utility:

- Updates the corresponding data policy record so that each specified field is set to the associated global list in the Global List field.
- Updates all the listed formats containing these fields to use the display list values of the associated global list.

The Messages pane of the localization utility screen displays a list of the updated data policy record and all updated formats.

Note: Once the formats are updated by the localization utility, removing them from the Formats list and rerunning the utility does **not** reverse the changes. The utility is designed to be a one-way process so any reversions need to be made manually inside Forms Designer. Likewise, all data policy changes are **not** reversed and will need to be reverted manually.

Once the localization process completes, all the defined fields should appear localized when displayed using one of the converted formats. Instead of English, they display in the language associated with the user session.

Codes

The codes table is a repository for storing user-defined values that you wish to localize. Enabled for localization, codes leverage the record tag localization utility and may be used for localizing lists of codes or status values. It offers an alternative to defining localized user-defined global lists without the difficulties of having to hardcode and to maintain the values inside the global list definition record.

To access codes, click

Tailoring > Codes.

Codes are comprised of the following fields:

- **Type:** a name by which to classify a set of values.
- **Value:** a value to be localized.
- **Sort Order:** a number indicating the sort order of this value.

Defining a set of values for a codes-based global list

When defining a set of values, a record needs to be created for each value and must be assigned to the same type.

For example: Use the Codes function to create (define) a series of codes for the Boolean values true and false.

Type	Value
Boolean	false
Boolean	true

Once all the codes are defined, proceed to ["Accessing and localizing message records for a codes-based global list" below](#) of the localization process.

Accessing and localizing message records for a codes-based global list

As with any table that is enabled for localization, the corresponding messages may be accessed from each codes record by clicking **More** or the More Actions icon and then selecting the View Localized Record Tags.

Once all the messages are updated to contain the correct translations, proceed with [Creating localized codes-based global lists](#).

Creating localized codes-based global lists

To define a global list for a set of codes, the global list definition must contain the following values:

- **List Field:** value
- **Filename:** codes

- **Limiting SQL:** type="<TYPE>", where <TYPE> refers to the type of codes you wish to localize.

In addition, you must:

- Check the Use Localized Values? check box.
- Ensure the List Variable and Display Variable fields are set to different variable names.
- Wait for the lister process to build the list.

Example: use the following inputs for Global List Definition: Boolean Values to create a localized list of Boolean Values.

Note: that the example references records that were created in the example found in [Defining a set of values for a codes-based global list](#).

Field Label	Value
List Name	Boolean Values
Build List on Startup?	checked
List Variable	\$G.booleans
Display Variable	\$G.booleans.local
Limiting SQL	type="Boolean"
Use Localized Values?	checked

Associate a field with a codes-based global list

To associate a field with a codes-based global list, you map the field to the global list in the same way you would with any other list.

Example: In the localization utility, use the following inputs to bind the Visible to Customer (cust.visible) field to the Boolean Values global list. It will take effect on the probsummary.qbe.g format.

Field	value
Table Name	probsummary
Formats	probsummary.qbe.g

Use the following values for the Field Name and Global List entries.

Field Name	Global List
Visible to Customer	Boolean Values

As a result of this operation, when you display a list of incidents, the column values for `cust.visible` appear localized if the `probsummary.qbe.g` format contains that field.

Message maintenance with record tag localization

The messages utilized by the localization process are maintained in the following manner:

- If a record is added to the localized table, corresponding message records are created for it.
- If a record in the localized table is updated so that the localized field is changed, its message records are deleted and recreated to reflect this. The Message ID of the new message records contains the new unique ID and the Text field is set to the new localized tag field value.
- If a record is deleted from the localized table, its message records are removed from the system.
- If the data policy for a localized table is deleted, all corresponding message records are removed from the system.
- If an enabled table is then disabled for localization, all corresponding message records are removed from the system.

Update message records for activated or deactivated languages

If a language is made active, new message records do not get created for it. However, rather than having to create these records from scratch you can perform the following steps.

To update messages for activated languages:

1. Access the data policy record for each localized table.
2. Click **More** or the More Actions icon and then select Update for Active Languages.
This creates message records for every active language that do not already exist.
3. Ensure that the new messages are properly translated.
4. Rebuild all applicable global lists to reflect the new changes.

Any languages that are deactivated or activated after launching the localization process are not automatically accounted for. If a language is made inactive, none of the corresponding message

records are removed. The records are left in the system in case the language is reactivated in the future. Hence, they will need to be manually deleted if desired.

Help systems

Service Manager has two Help systems:

- The online Help system explains Service Manager concepts and how to use Service Manager features. When you want to learn more about a feature or how to complete a task, online Help explains what you need to know.
- The field Help system contains a help on the fields in a form. When you are filling a form with data, field Help can tell you what information to provide in any field.

Note: In the web client, the field Help system is controlled by the **disableKeyHelp** parameter.

Online Help system

Service Manager 9.30 has a Web-based online help system, which consists of multiple documentation plug-ins that are not customizable. The documentation plug-ins are containers for the Help system XHTML topic files and each plug-in has two or three main folders:

- Concepts contains files that describe features and concepts
- Reference contains files of resource information, such as RAD function descriptions
- Tasks contains files with steps that you can follow to accomplish Service Manager tasks

Field help editor

The field Help system contains a help on the fields in a form. When you are filling a form with data, field Help can tell you what information to provide in any field.

The Field help editor enables you to add and edit help for any HP Service Manager field, form, table, term, or topic. Using it you can add a unique help record that describes the data in any field.

Service Manager stores field help as records in the `help` table. The table has fields that enable you to specify the brief description, detailed description, table name, form name, field name, related information, term and topic for a help record.

You can add, edit, or delete individual help records through the Field help editor. Alternatively, since the field help is stored in a standard Service Manager table, you can use the standard table related functions to add, edit, or delete batches of help records, including import, export, load, unload, and mass add, update, and delete.

You can run queries to find records in the help table, just as you would on any other Service Manager table, using either the standard QBE search, or the Advanced search functions.

Creating field help

When creating field help, keep in mind that the forms you see as you step through a process may not be the ones the user sees. What you are looking at is, most likely, not the only place the field is displayed.

- The forms displayed vary with the choices the user makes.
- The fields displayed on the forms vary with the choices the user makes and with his permissions.
- The use of the field varies depending on user permissions and where the field is being displayed. (For example, if data gets entered into read-only fields.)

When creating field help, you may need to do any of these tasks:

- Access the help table.
- Add or edit field help records.
- Run a query against the help table.
- Export a batch of existing field help records and edit them.
- Import a batch of new or edited field help records.
- Perform a mass update on a list of help records.
- Unload a batch of field help records from your local machine and load them into the development machine.
- Review new or updated field help records.

Preparing to create field help

To create field help efficiently, make these changes in your local HP Service Manager system.

- Show the context-sensitive help debug information.
- Activate the command line for the System Administrator (or the operator you want to enable to create field help) on your local server.
- Set up the columns displaying the help record list.
- Turn on the Administration perspective.

Show context-sensitive help debug information

1. Start the HP Service Manager client.
2. Log on or close the Connections dialog box.
3. Click **Window > Preferences**.

The Preferences dialog box opens.

4. Click **HP Service Manager > Appearance**.

The Appearance dialog box opens.

5. Select **Show context-sensitive help debug information**.
6. Click **Apply**.
7. Click **OK**.

Exit the Service Manager client, and then restart it.

Service Manager now displays the file (table), form, and field information when you press Ctrl+H for a field.

- You can use such field help information to add or edit field help records or identify field names to be used in stored queries.
- The Web client uses the "viewcontexthelp=true" URL parameter to display field help information.

Activate the command line

1. Start the HP Service Manager client.
2. Log on to your local server as a System Administrator.
3. Type `operator` on the command line and click **Execute Command**.

Service Manager displays the operator form (`operator.g`).
4. Type `System.Admin` as the log-on name (or specify the operator you want to set the permissions for), and then click **Search**.
5. Open the Startup tab and select **Activate Command Line at Startup**.
6. Click **Save**.
7. Click **Ok** to exit the record.
8. Click **Back** to exit the form.
9. Exit the Service Manager client, and then restart it.

Service Manager now displays the Command Line to that operator.

Set up the columns displaying the help record list

1. Open the help table in the Help Editor and click **Search** to display the help records.
2. Click **More** or the More Actions icon.
3. Click **Modify Columns**.
HP Service Manager displays a form that enables you to edit the columns in the record list.
4. Type or select the following, one per line in this order:
 - o `topic`
 - o `file.name`
 - o `format.name`
 - o `field.name`

- term
 - brief
 - detail
 - Pending Review
 - sysmodtime
5. Click **Proceed**.
Service Manager displays the record list with the columns you selected.

Turn on the Administration perspective

You can turn on the Administration perspective by doing one of the following:

1. Click **Window > Open Perspective > Other**, and then select **Administration**.
2. In the Service Manager System Navigator on the left side of the window, click **Administration Perspective**.

Service Manager now displays Administration tabs at the bottom of your forms.

Access the help table

Use one of the following procedures to access the help table:

1. Type `help` on the HP Service Manager command line and press Enter.
2. Use Database Manager:
 - Click **Tailoring > Database Manager**.
 - Type **help** in the Form field, and then click **Search**.

Service Manager displays the Help Description tab of the help form (help.g). Use this form to add or edit help records.

Note: If you do not see the command line, you must activate it for the operator you are using.

Access field help from the System Navigator

Applies to User Roles:

System Administrator

To access field help from the System Navigator:

1. Click **System Definition > Tables > tablename**.
2. Double-click a table name to view information about that table.
3. Click the **Fields and Keys** tab.
4. Select any field in the **Fields** section.
5. Select one of these links to create or edit the default help record.
 - **Create default help on this field**. This link appears when no field help exists for this field. Click this link to create field help.
 - **Edit default help on this field**. This link appears when field help already exists for this field. Click this link to edit existing field help.
6. Click **Search for specific help on this field in forms** to open and edit existing form-specific help for this field.
7. If you make changes, click **Save**.
8. Click **OK**.

Determine the fields that a form contains

1. Make sure you have the **Administration Perspective** open. To do this, do one of the following:
 - Click **Window > Open Perspective > Other**, and then select **Administration**.
 - In the Service Manager System Navigator on the left side of the window, click **Administration Perspective**.

This turns on the Administration Perspective, which allows you to see the administration tabs at the bottom of the form.

2. Open the form using Database Manager.

Note: Do not use the System Navigator as it will open the form in Forms Designer, which will not give you the data you need.

3. Select the **Detail Data** tab.

The Detail Data tab contains the XML that generates the form.

The sample XML below describes two fields:

- **Contact for this interaction:** A comfill for the `callback.contact` field.
- **Notify By:** A selection list for the `callback.type` field, containing the options None, Email, Telephone, and other.

```
<label x="2" scType="Label" id="Label13" width="32" height="2" forecolor="0"
y="2">Contact for this interaction:</label>
  <comfill ref="instance/callback.contact" x="36" scType="ComFill"
id="comfill11109271071725" mandatory="1" idFill="9" width="36" idFind="8"
arraylength="1" tabindex="1"
  foreignkeyref="Links/LinkLine[@SourceField='callback.contact']"
scComboButtonVisible="0" height="2" y="2" dataType="string" />
  <label x="2" scType="Label" id="" width="32" height="2" forecolor="0"
y="6">Notify By:</label>
  <select ref="instance/callback.type" BoxLines="4" x="36" id="" mandatory="1"
width="36" arraylength="1" selectonly="1" height="2" y="6" dataType="string">
    <option value="None">None</option>
    <option value="Email">Email</option>
    <option value="Telephone">Telephone</option>
    <option value="Other">Other</option>
  </select>
```

Note: You can copy and paste the data from this form into another file, and edit it down to something easier to read.

Determine the fields that a table contains

The System Navigator displays the fields in a table, along with many other table features.

To see the fields in the System Navigator:

1. Click **System Definition > Tables > *tablename***.
2. Select **Fields** to see fields.
3. Select **keys** to see the keys.

Determine whether help exists for a field

HP Service Manager may not display help that is available for a field when you press Ctrl+H in the Windows client or F1 in the Web client.

To determine the complete list of help records for a field, if there are any, you must search the help table.

1. Type `help` in the Service Manager command line and press Enter.
2. Specify your search criteria.
 - To search for help on a field in a table, specify the **Field Name** and the **Table Name**.
 - To search for help on all fields in a table, specify the **Table Name**.
 - To search for help on a field in a form, or if you don't know the table, specify the **Field Name**.
3. Click **Search** or press Enter.

Service Manager returns records that match your query, if there are any.

How does Service Manager determine which help record to display?

A field help record must have a field name and either a table name or a form name to be displayed.

- The default record for a field must have the field name and the table name, but no form name. HP Service Manager displays the default help record for a field for all forms that use the field, excluding those that have a form specific help record.
- A form specific help record for a field includes the form name. Service Manager only displays that particular help record when it displays that particular form.

If there is no help record for a field, Service Manager displays “No Description”.

If there is no form name or table name in a help record, Service Manager will not display that record as help on a field. However, you can still search for the record and open it using the Field help editor. This is useful if you want to create a glossary of terms, for example.

Best practice

To ensure consistent field descriptions on all forms, omit the form name from the field help record.

If you add a form name to a field help record, the text displays only when Service Manager displays that form. Adding a form name can be a useful technique for documenting special forms, but may increase the number of help records to maintain.

Add or edit help records

The user can also click “Accept” instead of Add/Save after editing the help. This will immediately publish the help topic with no additional review.

1. Click **Tailoring > Database Manager**.
2. Type **help** in the Form field, and then click **Search**.
3. Double-click the help (help.g) form.

Service Manager displays the Help Description tab of the help form (help.g). Use this form to add or edit help records.

4. Type a **Brief Description**.
5. Type the **Detailed Help** information, if necessary.
6. Select the **Keywords** tab.
7. Specify the **Field Name**.
8. Specify the **Table Name**.
9. Optionally, you can add the following in the Keywords tab:
 - Specify the **Form Name** — Use this field to create form-specific help.
 - Specify **Related Information** — Use this field to create a link to field help for related fields.
 - Specify the **Term** — Use this field to define glossary terms.
10. Optionally, you can specify the **Topic** in the Help Description tab. Use this field to input help on specific areas of HP Service Manager.
11. Click **Add** or **Save** to create new or revise existing field help.

The help topic is now marked **Pending Review**. The help topic is not published until it is reviewed and accepted. For more information, see ["Review field help records" on the next page](#).

Tip: You can click **Accept** instead to immediately publish the help topic without additional review.

Review field help records

New or revised help topics must be reviewed before they can be published. If you do not complete this step, the help records will not be visible to users.

To review field help records, follow these steps:

1. Connect to the server that contains the help you want to review.

2. Run this query against the help table.

```
pending.review="true"
```

HP Service Manager displays the records that currently need review.

3. Use one of the following procedures:

- Click **Accept** to indicate that you have reviewed an individual record.

It will take one minute or two for the background process to replace the original help topic with the revised topic after it is accepted.

- Use the following Mass Update to indicate that you have reviewed the list of records.

```
pending.review in $file=false
```

Note: If more than one person has input field help, you will need to refine your query to find the specific records you want to review.

Sequential number setup

The Sequential Number file is used in conjunction with Format Control to generate sequence numbers for records in a database. The sequence number is automatically incremented or decremented when a new record is added. For instructions on creating a Format Control record with sequential numbering, refer to Format Control Help.

Sequential number file

The Subroutines process of Format Control creates a Sequential Number File record when the application getnumb.fc is called. The parameters passed to the application from within Subroutines define the values in the number file fields. These values determine how the sequential number opens and what other information it contains. You can define number file values either directly in the number file or via parameters passed from Format Control. However, if you use the number file to change values also defined in the Format Control subroutines process, they are overridden by the Format Control values.

Access the sequential number file

The Sequential Number file is used in conjunction with Format Control to generate sequence numbers for records in a database.

To access the sequential number file:

Click **Tailoring > Tailoring Tools > Sequential Numbers**.

The sequential number file opens.

Create a simple number counter

The following example illustrates how to create a record called employee in the Sequential Number File to automatically increment employee numbers starting with 1.

1. Access the Sequential number file.
2. Create a new file. Type `employee` in the Class field for this example.
3. Type `0` in the Last Number field to begin incrementing from zero.
4. Type a short description for the number class.
For example: Employee ID number counter.
5. Type `1000` in the Reset Point field.
6. Type `1` for the Increment/Decrement field value.
Since the Decrement field was left blank, each number will increase by one.

7. Click **Add**.

The new sequential number record is added to the sequential number file.

Use decrement in sequential numbers

You can use sequential numbers to decrement a starting value. For example, you can decrement a quantity field when deleting stock from inventory. This example shows you how to decrement a value starting at 1000.

1. For this example, type `active.devices` in the Class field.
2. Type `1000` in the Last Number field.
3. Type a short description for the number class.
For example, type `Number of devices available`.
4. Type `true` in the Decrement field.
5. Type `0` in the Reset Point field.
6. Type `1` for the Increment/Decrement field value.
Since the Decrement field is set to true, each number decreases by one.
7. Click **Add**.
The new sequential number record is added to the sequential number file.

Use prefix and suffix in sequential numbers

This example uses prefixes and suffixes to assign character type ID numbers to workstation devices. The format of the ID is: `DEV<number>T` where `DEV` is a fixed character prefix, `<number >` is a sequential number starting with 1, and `T` is a fixed character suffix.

1. For this example, type `devices` in the Class field.
2. Type `0` in the Last Number field.
3. Type a short description for the number class.
For example, enter `Workstation device ID counter`.
4. Leave the Decrement field blank.

5. Type 1000 in the Reset Point field.
6. Type 1 for the Increment/Decrement field value.
7. Type 5 in the Length field.
8. Type DEV in the Prefix field.
9. Type T in the Suffix field.
10. Click **Add**.

The new sequential number record is added to the sequential number file. The first sequential number for the devices class is DEV00001T.

Update a sequential number record

1. Access the sequential number record. Use the search function or select the record from a record list.
2. Enter changes to the fields you want to update.
3. Click **Save**.

Delete a sequential number record

1. Access the sequential number record that you want to delete. Use the search function or select the record from a record list.
2. Click **Delete**.
3. Click **Yes** to confirm that you want to delete the record.

The form opens with the message: *Record deleted from the number file.*

Stored queries

Stored queries enable users to retrieve and display current information efficiently by using predefined search parameters. Depending on permissions, users can run predefined queries, modify predefined queries before running them, or store a personal list of modified queries and run a query from their list.

Administrators can use stored queries to display lists of specific records or populate dynamic display objects such as charts and marquees. They can add new queries and fine tune the out-of-box queries to decrease response time or implement best practices in their environment.

Stored Query Maintenance utility

The Stored Query Maintenance utility allows designated users to define and store queries that display lists of specific records or populate dynamic display objects such as charts and marquees.

Stored queries commonly run from the following features in HP Service Manager:

- Advanced and Expert Search menu options
- Display objects
- Buttons
- Scripts

You could use a stored query to do any of the following:

- Search for incident records that have reached a certain status
- Populate a chart that displays open records by category
- Display a list of change requests assigned to a particular approval group at the click of a button

Using stored queries in display objects

An administrator can use stored queries to retrieve and display dynamic data using menu buttons, charts, and marquees. Stored queries applied in this manner are not accessible to the user and operate in the background to retrieve records from the database.

For example, you could place a dynamic chart on a supervisor's startup menu showing all open records by category. By placing buttons that run individual stored queries on the bottom of the chart, the supervisor could display lists of records by category.

Using menu buttons to run stored queries

Buttons that run stored queries from a menu must have a button ID defined in a menu record. Use the following values to run a stored query from a menu:

- **Application:** query.stored
- **Parameter Name:** text
- **Parameter Value:** <stored query name>
- **Condition:** true (or a condition statement that checks for a particular query group)

Using stored queries to produce charts and marquees

Stored queries produce dynamic information that the system gathers and displays in a bar chart. Agent records that reference each display object in Forms Designer define these queries.

A marquee runs one stored query at a time and cannot provide access to actual records. Queries run from marquees that display messages about records.

Using stored queries in scripts

You can use a stored query to execute a script, and you can write a script that uses a stored query to locate specific records.

A script calls stored queries to populate forms with useful data. Use this capability to grant limited database access to certain users.

For example: You could allow Level 1 technicians access to a user information form through which they can update selected elements of a caller's contact record. The script calls the stored query. The query retrieves and displays the data from the contacts file. The technician makes the updates and saves them. The script then continues walking the Level 1 technician through the normal workflow.

Menu option searches

Users can run stored queries from the Expert Search menu option in search forms for the principal HP Service Manager applications (for example, Incident Management, Change Management, and Configuration Management) or from the Advanced Search menu option in the Database Manager. The appearance of these options and the features they control depend on the capabilities defined for each user in the operator record.

Define which system processes manage message traffic

1. Click **System Administration > Base System Configuration > Miscellaneous > System Information Record**.
2. Click the **Message Processors** tab.
3. In the **Processor Name** field, type the names of the background or scheduler processes that are to manage message traffic. The processor name must match the class name specified for the message class processor.
4. Click **Save**.
5. Start the message processors.

HP Service Manager rotates messages among all the processes you define. If you do not define any message processors, then Service Manager uses the problem processor to manage all message traffic.

Grant access to stored queries

To grant access to stored queries, add the appropriate capability words to the user's operator record.

- **QueryAdmin**: Grants administrative access to query options and maintenance and the ability to create and modify stored queries.
- **query.stored**: Enables a user to run stored queries that are assigned to the user (or the user's group).
- **query.stored.mod**: Enables a user to run and to temporarily modify stored queries that are assigned to the user (or the user's group). The user cannot save the modified stored query.
- **query.window**: Grants access to the Query window.

Add a stored query

Applies to User Roles:

System Administrator

You can add, edit, or run stored queries from the Query window, which is accessible from the form in Database Manager or from the Application search form. Data entered into the form is automatically entered into the Query window to form the query. Depending on your permissions, you can edit and run that query, or select a query from a list of stored queries.

In this example, a query is performed on a specific assignee for all contracts in Configuration Management. Once stored, operators or groups of operators can access the query and automatically generate a list of records without reentering the detailed search criteria.

Note: You must have the capability word **QueryAdmin** or **query.stored.mod** operator record to add or edit stored queries.

To add a stored query:

1. Click **Configuration Management > Contracts > Contracts**.
2. Use the Fill function in the Assignee field to choose an assignee. For this example, Carlton Hulman was chosen.
3. Click the More Actions icon and choose **Expert Search**.

The Query form opens. The new query is displayed in the **Query** field. For this example, the query for Carlton Hulman is displayed as follows: `assignee#"Carlton.Hulman"`

Note: In Service Manager, you can use field name information in field help to identify the field names you want to use in your stored queries. To display field name information in field help, make sure one of the following conditions is fulfilled:

- In the Windows client, the “Show context-sensitive help debug information” preference is enabled.
- In the Web client, “viewcontexthelp=true” is appended to the URL before you log in.

4. On the Query form, click **Keys**.

Note: Do not modify the automatically-generated query string, `assignee#"Carlton.Hulman"`, as it appears in the **Query** field.

5. On The Key Window (keylist form), type a key number to select the number of positions for the **assignee.name** key in the query. For example, type the number 3 to select 3 positions.

6. Click **OK**.

The Sort Fields field in the Query Window is now modified to agree with the key definition you selected.

7. On the Query form, click **Store** to create the **querystored** file.

8. Fill in the following fields:

- In the **Access List** field, type in the Query Groups or Operator Names that you want to have access to this query. If you want all users to have access, leave the list blank.
- In the **Name** field, type a unique name for your query.
- In the **Description** field, type an explanatory description.

9. Click **Add**.

The following message confirms your entry was successful: Query added to querystored file.

10. Continue to further refine your search criteria, or click **Search**. You can also click any record in the list of returned records to view the details.

Create stored queries from the Query Maintenance form

You can add, edit or run stored queries from the Query window, which is accessible from the form in Database Manager or from the Application search form. Data entered into the form is automatically entered into the Query window to form the query. Depending on your permissions, you can edit and run that query, or select a query from a list of stored queries.

1. Click **Tailoring > Tailoring Tools > Stored Queries**.

A blank Stored Query Maintenance form opens.

2. Click **Search**.

A list of existing queries opens.

3. You can copy an existing query, create a brand new query, or create a new query from an existing query.

Note: In Service Manager, you can use field name information in field help to identify the field names you want to use in your stored queries. To display field name information in field help, make sure one of the following conditions is fulfilled:

- In the Windows client, the “Show context-sensitive help debug information” preference is enabled.
- In the Web client, “viewcontexthelp=true” is appended to the URL before you log in.

4. Click **Save** to save any changes.

Caution: If you are creating a new stored query from an existing query, make sure you click Add instead of save. If you click save, you will replace the existing stored query with the new stored query you are attempting to add.

5. Click **Add** to add a new query.

You receive a message that states the record has been added to the querystored file.

Run a stored query

Applies to User Roles:

System Administrator

Depending on your permissions, you can add, edit, or run stored queries from the Query window, which is accessible from the form in Database Manager or from the Application search form. Data entered into the form is automatically entered into the Query window to form the query. Depending on your permissions, you can edit and run that query, or select a query from a list of stored queries.

Example: Selecting and running a stored query.

In this example, a stored query for Incident Management (the probsummary file) is selected and run.

1. Open the query window.
 - To open the query window from a HP Service Manager application:
 - i. Open the application Search form. For this example, click **Incident Management > Search Incidents**.

- ii. Open the More Actions menu.
- iii. Select **Expert Search**.
 - To open the query window from a Database Manager search form:
 - i. Open the form in Database Manager. For this example, type =probsummary in the **Form** field, and then press Enter.
 - ii. Open the More Actions menu.
 - iii. Select **Expert Search**.

The Query window opens.

2. Click **Select** to display the list of stored queries for the associated file.
3. Double-click a query to select it. For this example, select **status.resolved**. The Stored Query record opens.
4. Click **Select**.

Service Manager returns you to the Query window.

5. Click **Execute Search** to run the stored query.

Records matching the query display in a record list.

- If only one record matches your search criteria, that record opens.
- If no records match, the server displays the following message: *No records found*.

Update a stored query

Applies to User Roles:

System Administrator

You can add, edit, or run stored queries from the Query window, which is accessible from the form in Database Manager or from the Application search form. Data entered into the form is automatically entered into the Query window to form the query. Depending on your permissions, you can edit and run that query or select a query from a list of stored queries.

In this example, we update and save a stored query for Incident Management (the probsummary file).

To update a stored query:

1. Open the query window.

- To open the query window from an HP Service Manager application:
 - i. Open the application Search form. For this example, click **Incident Management > Search Incidents**.
 - ii. Open the More Actions menu.
 - iii. Select **Expert Search**.
- To open the query window from a Database Manager search form:
 - i. Open the form in Database Manager. For this example, type `=probsummary` in the **Form** field, and then press Enter.
 - ii. Open the More Actions menu.
 - iii. Select **Advanced Search**.

The Query window opens.

2. Click **Select** to display the list of stored queries for the associated file.
3. Double-click a query to select it. For this example, select **status.resolved**.

The Stored Query record opens.

4. Click **Select** to edit the query as desired. In this example, change the query to `flag=true` and `problem.status<>"Resolved"`.

Note: In Service Manager, you can use field name information in field help to identify the field names you want to use in your stored queries. To display field name information in field help, make sure one of the following conditions is fulfilled:

- In the Windows client, the “Show context-sensitive help debug information” preference is enabled.
- In the Web client, “viewcontexthelp=true” is appended to the URL before you log in.

5. Specify any other controls you want.

6. Click **Store** to save your edits to the selected query.

The Stored Query Maintenance form opens.

7. Give the query a unique **Name**. For this example, not .resolved.
8. Add a **Description**. For this example, type Unresolved records.
9. Click **Add**.

Service Manager returns you to the Query window and displays the message: *“Query added to querystored file”*.

You can click **Execute Search** to run the saved query at this point, or click **Back** to exit.

Form creation

These topics focus on creating and customizing forms within the system. The tools that are used in form creation are:

- Forms Designer — alters the data in one or more dependent files to match changes made to data in a source file.
- Dynamic View Dependencies (DVD) — applies values, sets mandatory fields, and validates at the table level.
- Format Control — defines a field's valid values.

Forms Designer

Forms Designer is the utility you use to design, create, and update Service Manager forms. This utility gives access to a drawing canvas upon which you construct your forms, a design toolbar from which you access controls and layout objects, and a properties view that you use to set attributes for each control or layout object. Forms Designer provides a Form Wizard that you can use to automatically create forms based on a particular database dictionary.

Access Forms Designer

Applies to User Roles:

System Administrator

You can open Forms Designer using one of the following methods:

- To access Forms Designer from the System Navigator, click **Tailoring > Forms Designer**.
- To access Forms Designer from the Service Manager command line, type `fd` and press Enter.

From this screen, you can create a new form or search for an existing form.

Creating and editing forms

The Form Wizard enables you to create forms based on a particular Service Manager table. You can automatically create both table-type record lists and single-record display forms. You can only use the Form Wizard to create new forms; it cannot modify existing forms.

Forms Designer design mode contains a **Drawing Canvas**, **Properties view**, and **Design toolbar**. You use them to create, design and modify Service Manager forms.

Forms Designer enables you to use the forms you create in three basic ways. A form can be used as a stand-alone, as a subform embedded in a parent form, or as a pop-up.

Create a form using the Form Wizard

Applies to User Roles:

System Administrator

You use the Form Wizard to create a new form for an existing database table. You must use the Windows client to create a new form in Forms Designer.

To create a form using the Form Wizard:

1. Click **Tailoring > Forms Designer** in the System Navigator, or type `fd` on the Service Manager command line and then press **Enter**.
2. Click **New**.
3. Click **Yes** to use the Form Wizard.
4. Type the name of the new form and click **OK**.
5. Type or select the name of the table you want to associate with the form, and click **OK**.
6. You can create a form that displays a single record or a form that displays a list of records. Select the type of form you want to create and click **OK**.
7. Select a size for the form.
8. Change the `true` to `false` in the Show (t/f) column if you do not want a particular field included on the completed form.

9. Change the text in the Field Label column to update the label on the form.
10. Click **Proceed**. The new form opens in Design mode.
11. Add a field to the new form, as follows:
 - a. Drag the control to its approximate location, select the field, and then modify its Properties.
 - b. To modify a field, select it and then modify the value of the Property you want to adjust.
 - c. To delete a field, select it and press **Delete**.
 - d. Click **Web Preview** to see what the form will look like in the Web client.
 - e. Click **OK** to see what the form will look like in the Windows client.
 - f. Click **OK** to save your changes.
12. Open the form using Database Manager to test your changes:
 - a. Click **Tailoring > Database Manager**.
 - b. Click **Search**. Service Manager opens a list of forms.
 - c. Click the form you just created to open it.

Update a form

Applies to User Roles:

System Administrator

To update a form:

1. Click **Tailoring > Forms Designer** in the System Navigator, or type `fd` on the Service Manager command line and then press **Enter**.
2. Type a form name and click **Search**.

Service Manager opens a list of forms. Click the form you want to update to open it.
3. Click **Design** to open design mode.

4. Add controls in any of the following ways:
 - Drag and drop the control from the toolbar to the drawing canvas to create a default-sized object.
 - Select a control in the toolbar and then click anywhere on the drawing canvas to create a default-sized object.
 - Drag and drop field(s) from a table in the System Definition onto the drawing canvas.
 - Cut, copy, or paste a control within the same form, from a different form, or from different Service Manager client.

The properties for the control you selected appear in the properties view.

5. Specify the name of the form in the Value text box of the Name property.
6. Specify the database field to access in the Value text box of the Input property.
7. Specify the height, width, and other properties as desired.
8. To delete a control from the drawing canvas, select the control and press **Delete**.
9. Click **Web Preview** to see what the form will look like in the Web client.
10. Click **OK** to see what the form will look like in the Windows client.
11. Click **OK** to save your changes.
12. Open the form using Database Manager to test your changes:
 - a. Click **Tailoring > Database Manager**.
 - b. Click **Search**.

Service Manager opens a list of forms.
 - c. Click the form you just created to open it.

Using the drawing canvas

When you access Form Designer to create a new form, you are provided with a drawing canvas on which you build the form.

To work effectively with the canvas, you should know the following:

- The structure of the canvas.
- How to size your forms on the canvas.
- How to position multiple objects.

The canvas contains a grid to which all design objects are referenced. Click **Grid Preview** to see the grid.

- The grid coordinates start at (0,0) in the top left corner of the canvas. The X (horizontal) coordinates increase as you move to the right. The Y (vertical) coordinates increase as you move down.
- There is approximately a 3 to 1 ratio between X units and Y units. For example, 30 X units cover approximately the same amount of space as 10 Y units.
- For best results, try to use even X Y coordinates for objects you place on the canvas. The space required to display one character is 2x2 and, overall, the client displays better when you use even coordinates.

Forms Designer properties view

The Properties view enables you to examine and set the property value for each object on your form. The values in the property fields determine how the object appears on the form, and how the object interrelates with the user and with the database.

The Properties view displays a window containing two columns, a property column and a value column. You can click the Property tab to sort the properties alphabetically or by category.

Like any view in the client, you can undock the properties view and move it to anywhere on your desktop.

Note: You can turn on the properties view and use it outside of Forms Designer to determine form field properties for any object in focus. However, since labels cannot gain focus, you cannot view properties for labels outside of Forms Designer.

The Properties view includes property specific in-place editors for changing values. There are six types of property editors:

- Text
- Boolean (check box)
- Color swatches
- Comboboxes

- Lists (with editor window)
- Images preview

Setting properties

When you select a control object, the properties view displays the type of control you have selected and the properties that apply to that object. Some properties are common to all control or layout objects, while others are object specific. The Forms Designer online Help contains a description of each control and its corresponding properties.




You can use the dynamic view dependencies feature to conditionally display some controls or conditionally change how some controls appear on a form.














Properties for a design object can be divided into the following categories:













- Common properties
- Font properties
- Box properties
- Object specific properties










Forms Designer controls and tools

Forms Designer has the following controls and tools. They are listed in the order in which they appear in the toolbar. Each control has different properties. Click the Name link to view the details.

Control or Tool	Name	Description
	"Grid view tool" on page 171	Toggles the alignment grid that you can use while designing the form.
	"Selection tool" on page 190	Enables you to select, deselect, and modify design objects.
	"Notebook control" on page 187	Adds a container that subdivides the contents of a screen into logical groups or categories.

Control or Tool	Name	Description
	"Notebook Tab control" on page 186	Adds a container for a notebook that has two or more pages. Navigate to each page by selecting its tab.
	"Group control" on page 171	Adds a container that groups logically associated items and includes a text label at the top.
	"Frame control" on page 169	Adds a container that groups logically associated items.
	"Label control" on page 179	Adds a place for you to type a title for the form, a label for an object within the form, or otherwise place a single line of text on the form.
	"Wrap label control" on page 209	Adds a multiline label.
	"Link label control" on page 181	Adds a label that includes a caption and a icon that links to another source.
	"Text control" on page 203	Adds a text box that displays the contents of a text field and conditionally enables users to enter or modify its contents.
	"Text area control" on page 201	Adds a text area that displays the contents of a text field and conditionally enables users to input several lines of data.
	"Decimal control" on page 163	Adds an input field that displays and enables users to update the contents of a numeric field and offers spinner buttons to increase or decrease a value.
	"Date control" on page 160	Adds a text box that displays and enables users to update the contents of a date field.
	"Combo Box control" on page 149	Adds a Combo Box that enables users to click a button and select from a drop-down list.
	"Comfill control" on page 153	Adds a combination Combo Box and fill button. You can specify a virtual join subform to display in a pop-up when the user hovers over this field.
	"List Builder	Adds a compound widget that allows the user to construct a list of choices.

Control or Tool	Name	Description
	"control" on page 183	
	"Button control" on page 142	Adds a button that activates a Control ID when clicked. You can use text, graphics, or both to customize buttons.
	"Checkbox control" on page 147	Adds a checkbox that displays and enables users to update the contents of a Boolean (logical) field, which can evaluate to true, false, unknown, or null.
	"Radio button control" on page 188	Adds a radio button that enables users to select one value from a set of mutually exclusive values.
	"Table control" on page 198	Adds a table that displays one or more columns of data in a scrollable pane.
	"Table column control" on page 194	Adds a column to an existing table.
	"Timer control" on page 206	Adds a clock timer.
	"File control" on page 167	Adds an Open File dialog box.
	"Chart control" on page 146	Adds a chart that displays the contents of a numeric array as a two-dimensional, color-coded bar chart with optional, definable buttons.
	"Image control" on page 178	Adds a place to display a Microsoft Windows image file.
	"Marquee control" on page 185	Adds an attention-getting text message that scrolls continuously across the screen from right to left.
	"Subform control" on page 193	Adds a sub-form to a larger form.
	"Attachments	Adds a box into which users can place non- Service Manager documents (for

Control or Tool	Name	Description
	control" on the next page	example, from Microsoft Word or Microsoft Excel).
	"Script control" on page 191	Adds a place for users to enter JavaScript into a color-coded syntax editor.
	"Dynamic Form control" on page 165	Adds a dynamic form that becomes visible when populated by XML code from a RAD application or JavaScript.
	"HTML Editor control" on page 174	Adds an HTML Editor that enables users to add standard HTML to the form.
	"HTML Viewer control" on page 176	Adds an HTML Viewer that enables users to view the HTML created using the HTML Editor.
	"Embedded Viewer control" on page 166	Adds a viewer that enables Web client users to view integrations.
	"Graph control" on page 170	Displays a diagram of connected nodes representing a system of interrelated things.
	"Calendar control" on page 144	Adds a calendar.
	"Web Preview tool" on page 209	Enables you to see what the form will look like when displayed by the Web client.
	"Convert Form Layout tool" on page 159	Converts all of the fields on the left side of the form into one group, and converts each of the notebook tabs on the right into a group, arranged vertically, under the left side fields. In the Web client, these groups are collapsible.

Attachments control

Use this control to add a box into which users can place non- Service Manager documents (for example, from Microsoft Word or Microsoft Excel).

To place an attachment container on a form, click **Attachments** and then click the form.

Note: To save attachments with incident records using Qopen (Quick Open), the Delay Assigning Problem Number field in the Incident Management Environment profile must be cleared. If Delay Assigning Problem Number is selected (set to true), then attachments are not saved with the incident record when performing a Qopen. Only the Qopen operation is affected by this setting.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Maximum Attachment Size	Specify the absolute size of an attachment, in bytes. Size limits defined in the Maximum Attachment Size field in a user's operator record override the size limit specified in the Company record. A non-zero value overrides any values specified in the Company or Operator record.
Maximum Attachment Size Condition	Specify a RAD expression to determine the maximum allowed size of an attachment based on some condition such as the capabilities of a user. For example, you can write a RAD expression that checks an operator's capabilities word and assigns a system administrator a higher attachment

Property	Usage
	size limit than a manager, who in turn could have a higher attachment size limit than a service desk user.
Total Attachment Size	Specify the maximum amount of memory, in bytes, that all attachments in a form can use. The total size of all attachments must be lower than this threshold or the form no longer accepts additional attachments.
Total Attachment Size Condition	Specify a RAD expression to determine the total amount of memory that all attachments in the form can use based on some condition such as the capabilities of a user.
Maximum Attachments Allowed	Select whether you want to allow one or any number of attachments.

Button control

Use this control to add an input field that displays and enables users to update the contents of a numeric field and offers spinner buttons to increase or decrease a value.

To place a button on a form, click **Button** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Note: Do not use the value 888 as a Button ID. The value 888 is a reserved number.

Best practices

- Use easy to understand names and descriptions for buttons.
- Make all buttons on a form the same type (all image, all text button or all text link).
- Place navigational buttons along the left edge or the top border of the form.
- Place other buttons at the top or bottom border.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.

Property	Usage
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Justification	Specify the justification for the object label as Left, Right, or Center.
Button ID	Specify a numeric identification that specifies a Control ID to transmit when clicked.
Button ID Condition	Specify an expression to override the Button ID property when that expression evaluates to true.
Image File	<p>Specifies the name of an image file to display on the control. The image file must reside in any of the following paths, in which Service Manager looks for the image:</p> <p>Windows client:</p> <ul style="list-style-type: none"> • <Service Manager>\Client\plugins\com.hp.ov.sm.client.common_x.xx.xxx\src\resources\icons • A local directory on the client machine, which is specified in the Image path field (Window > Preferences). For example: C:\smCustomImages. <p>Web client:</p> <ul style="list-style-type: none"> • <Service Manager web tier>\images. For example: <Tomcat>\webapps\webtier-9.41\images. <p>Note: When you specify the image file for this property, you need to provide both the file name and the file name extension.</p>

Property	Usage
Image Condition	Specify an expression to override the Image property when that expression evaluates to true.
Balloon Help	Specify a text string that appears when the cursor is held over the button to offer more information about the button's function.
Enabled	Specify an initial state.
Enabled Condition	Specify a condition that enables the button when a table row is selected. Example: [%tablename] == "selected" or [%tablename] != "selected" where tablename is the value of the Name property of the table on the screen.

Calendar control

Use this control in Forms Designer to add a calendar to a form. To place a calendar on a form, click **Calendar** and then click the form.

Note: The calendar widget is displayed well only in the web client. When you view the form from the Windows client, this widget is displayed as a blank area.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object. Example: calendar.
X	Specify the object's horizontal position based on the left edge of the object. Example: 0
Y	Specify the object's vertical position based on the top edge of the object. Example: 1
Width	Specify the width of the object in alignment grid units. Example: 223
Height	Specify the height of the object in alignment grid units. Example: 44
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form. Example: Select this option.

Property	Usage
Visible Condition	<p>Specify an expression to override the Visible property when that expression evaluates to true.</p> <p>Example: (empty)</p>
Input	<p>Specify the database field or variable to associate with this control. The primary use for this property is to set the focus on the graph.</p> <p>Example: (empty)</p>
Tab Stop	<p>Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button, or check box, you will almost always use the default of 0.</p> <p>Example: 0</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <div style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <p>Note: This function is only supported by the web client. If no description is present, the web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the web client will use the component's text. Accessibility software can query the web client to get this information. The client uses that information to present data to the user in a variety of ways. For example, speech simulation software will say the name and the type of component when the component gets focus, based on the settings in the Accessible Name and Accessible Description fields.</p> </div> <p>Example: (empty)</p>
Date	<p>Specify the start date for data display in the calendar.</p> <p>Currently only the <code>\$start.date</code> value is supported. This variable takes the value of the field specified in Field for Initial Date of each Embedded Calendar Configuration setting of an object (Change, Incident, and so on). See Configure search filters for the full calendar.</p> <p>For example, if you specify the Planned Start field as the Field for Initial Date for the Change object, in a change record whose Planned Start is May 23, 2013, the calendar displays data only for dates no earlier than May 23, 2013.</p>
Predefined Query	<p>Specify a query for data display in the calendar.</p> <p>Currently only the <code>\$predefined.query.string</code> value is supported. This variable takes the query specified in each Embedded Calendar Configuration setting of an object. See Configure search filters for the full calendar.</p>
Supported Views	<p>Specify the views that you want the calendar to support. The value can be any combinations of <code>day</code>, <code>week</code>, and <code>month</code>, separated with a semicolon (;).</p> <p>If you leave this property empty, all three views are supported.</p>

Property	Usage
	Example: day;week
Default View	<p>Specify the initial view when the calendar is opened.</p> <p>Currently only the <code>\$default.view</code> value is supported. This variable takes the Default View value specified in each Embedded Calendar Configuration setting of an object. See Configure search filters for the full calendar.</p> <p>For example, if you specify Week as the Default View of the embedded calendar configuration for the Change object, the calendar is displayed in the weekly view in the Calendar section of each Change record.</p>

Chart control

Use this control to add a chart that displays the contents of a numeric array as a two-dimensional, color-coded bar chart with optional, definable buttons.

To place a chart on a form, click **Chart** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	<p>Specify a list of values to display in the chart.</p> <p>Example: 3;4;10;3</p>
Input	<p>Specify the database field or variable to associate with this control.</p> <p>For more information on what to pass to the field or variable refer to the Publish and Subscribe Help.</p>

Property	Usage
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Foreground Color	Select the text color from the drop-down list.
Balloon Help	Specify the text string to display when the cursor is held over the button to offer more information about the button's function.
Bar Width	Specify the width of bars in alignment grid units.
Scale Max	Specify the maximum vertical bar height. Columns that exceed the Scale Max setting appear clipped. A setting of 0 causes the chart to automatically scale vertically to the tallest column.
Button Base	Select this option to enable a row of buttons across the base of the chart, one button per column.
Base Button ID	<p>Specify the Control ID for the left-most chart button. It is incremented by one from left to right. Using the default value of 801 as an example, the left-most button would transmit 801, the second button from the left would transmit 802, the third 803, and so forth.</p> <p>The Base Button ID relates to an option number in the menu record. Assign Button IDs to the chart and create an associated option number in the menu record, so that each button calls a stored query.</p>
Color List	Specify a list of semicolon delimited colors. The Color Scale and Color Percent properties determine how the list is applied to the columns. The available colors are: black, red, green, blue, gray, light gray, dark gray, yellow, cyan, magenta, white, forest, navy, purple, teal, brick, and manila.
Color Scale	Specify a list of semicolon delimited numeric values to determine the color of a column.
Color Percent	Select this option to use the Color Scale property a list of percentages from 1 to 100.

Checkbox control

Use this control to add a checkbox that displays and enables users to update the contents of a Boolean (logical) field, which can evaluate to true, false, unknown, or null.

To place a checkbox on a form, click **Checkbox** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information. The client uses that information to present data to the user in a variety of ways. For example, speech simulation software will say the name and the type of component when the component gets focus, based on the settings in the Accessible Name and Accessible Description fields.</p>

Property	Usage
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Foreground Color	Select the text color from the drop-down list.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Array Length	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none">- If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries.- If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array.- If the Input data type is scalar, only a single text box appears. <p>Note: The screen object must be associated with an array data structure.</p> <p>The default is 0, which means one vertical line of information appears.</p>
Data Changed Event	Specify the option number (such as Button ID) to call if the data contained in the object changed. This sends an event to the display RAD application.

Combo Box control

Use this control to add a Combo Box that enables users to click a button and select from a drop-down list. The items in the list are associated with a database field or variable. Users can also type in a value if the check box in the selectonly property is cleared (set to false).

To place a Combo Box on a form, click **Combo Box** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Best practice

Use eight drop-down lines unless the drop-down set contains a fixed number of elements less than or equal to seven, in which case the number of lines should be equal to the number of elements in the drop-down set.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	Specify a component name for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.
Accessible Description	Specify a component description for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information,

Property	Usage
	and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Mandatory	<p>Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only.</p> <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
Mandatory Condition	Specify an expression to override the Mandatory property when that expression evaluates to true.
Array Length	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none"> - If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. - If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. - If the Input data type is scalar, only a single text box appears. <p>Note: The screen object must be associated with an array data structure.</p> <p>The default is 0, which means one vertical line of information appears.</p>
Password	Select this option to make the text entered in the field a series of asterisks on the screen. The data is not encrypted, just transposed.
Maximum Chars	Specify the number of characters the user can enter in the field. The default is 0 (unlimited).
Maximum Characters Beep	Select this option to notify the user by sounding a beep that they

Property	Usage
	reached the maximum number of characters.
Case Conversion	Specify how to change the case of text entered in the field. The choices are None, Upper, Lower.
Decimals	Specify the number of decimal points for numbers in this field. Service Manager modifies numbers entered in the field to the number of decimal points you specify.
Parse	Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).
Data Changed Event	Specify the option number (such as Button ID) to call if the data contained in the object changed. This sends an event to the display RAD application.
Value List	<p>In conjunction with the Input property, defines how the Display List values are identified in the database. Value and Display Lists are entered using the Edit List dialog box. You can enter hard coded entries for each list, or you can supply a variable as the first and only entry. The run time values of the variable are used to populate these lists.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p> <p>Note: When defining Value List and Display List properties, you should avoid using the following reserved characters:</p> <ul style="list-style-type: none"> • semicolon (;) • tab key • newline (carriage return) • form feed • backspace • equal sign (=) <p>Important: Do not use keys (like backspace), as the system cannot process their ASCII representation entries correctly which causes unpredictable results.</p>
Value List Condition	Specify an expression to override the property when that expression evaluates to true.
Display List	Specify the values that appear in the drop-down list at run time. There

Property	Usage
	<p>must be a one-to-one correspondence between the values for Value List and for Display List. If the Display List is the only populated property, the display values are written to the database.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p> <p>Note: When defining Value List and Display List properties, you should avoid using the following reserved characters:</p> <ul style="list-style-type: none"> • semicolon (;) • tab key • newline (carriage return) • form feed • backspace • equal sign (=) <p>Important: Do not use keys (like backspace), as the system cannot process their ASCII representation entries correctly which causes unpredictable results.</p>
Display List Condition	Specify an expression to override the Display List property, when the expression evaluates to True.
Box Lines	Specify the window size of the drop-down list.
Select Only	<p>Select this option to require the user to select from the Combo Box drop-down list. If the user makes a manual entry, the value must already exist in the drop-down list (defined in the Value List and Display List properties).</p> <p>Clear this option to allow the user to select from the drop-down list or to manually enter a value.</p>

Comfill control

Use this control to add a combination Combo Box and Fill button. Comfill has all the properties of a Combo Box, plus the capability for Fill button. You can select which comfill buttons to display. For example, to display just a Fill button, set the Third Button Visible property to true by selecting the checkbox and set the Combo Button Visible property to false, by clearing the checkbox.

You can specify a form to display as a pop-up when the user hovers over the comfill. Subforms displayed as pop-ups are read-only; users cannot interact with them. If you want your users to be able to interact with the form, you must display the subform on another form instead of as a pop-up. To display the form as a subform, use the subform control.

To place a comfill box on a form, click **Comfill** and then click the form.

Put the name of the database field or variable that you want to associate with this control into the **Input** property.

Best practice

Use image id=7 for fill boxes. Find and Fill buttons should not use the down arrow icon, because it makes the Find or Fill look like a Combo Box.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Unused.
Caption Condition	Unused.
Input	Specify the database field or variable to associate with this control.
Accessible Name	Specify a component name for use with accessibility software. (Optional) Note: This function is only supported by the Web client.

Property	Usage
	<p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
<p>Accessible Description</p>	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
<p>Tab Stop</p>	<p>Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.</p>
<p>Read-Only</p>	<p>Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.</p>
<p>Read-Only Condition</p>	<p>Specify an expression to override the Read-Only property when that expression evaluates to true.</p>
<p>Mandatory</p>	<p>Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only.</p> <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
<p>Mandatory Condition</p>	<p>Specify an expression to override the Mandatory property when that expression evaluates to true.</p>
<p>Array Length</p>	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the</p>

Property	Usage
	<p>array entries.</p> <ul style="list-style-type: none"> - If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. - If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. - If the Input data type is scalar, only a single text box appears. <p>Note: The screen object must be associated with an array data structure.</p> <p>The default is 0, which means one vertical line of information appears.</p>
Password	<p>Select this option to make the text entered in the field a series of asterisks on the screen. The data is not encrypted, just transposed.</p>
Maximum Chars	<p>Specify the number of characters the user can enter in the field. The default is 0 (unlimited).</p>
Maximum Characters Beep	<p>Select this option to notify the user by sounding a beep that they reached the maximum number of characters.</p>
Case Conversion	<p>Specify how to change the case of text entered in the field. The choices are None, Upper, Lower.</p>
Decimals	<p>Specify the number of decimal points for numbers in this field. Service Manager modifies numbers entered in the field to the number of decimal points you specify.</p>
Parse	<p>Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).</p>
Data Changed Event	<p>Specify the option number (such as Button ID) to call if the data contained in the object changed. This sends an event to the display RAD application.</p>
Value List	<p>In conjunction with the Input property, defines how the Display List values are identified in the database. Value and Display Lists are entered using the Edit List dialog box. You can enter hard coded entries for each list, or you can supply a variable as the first and only entry. The run time values of the variable are used to populate these lists.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p>

Property	Usage
	<p>Note: When defining Value List and Display List properties, you should avoid using the following reserved characters:</p> <ul style="list-style-type: none"> • semicolon (;) • tab key • newline (carriage return) • form feed • backspace • equal sign (=) <p>Important: Do not use keys (like backspace), as the system cannot process their ASCII representation entries correctly which causes unpredictable results.</p>
<p>Value List Condition</p>	<p>Specify an expression to override the property when that expression evaluates to true.</p>
<p>Display List</p>	<p>Specify the values that appear in the drop-down list at run time. There must be a one-to-one correspondence between the values for Value List and for Display List. If the Display List is the only populated property, the display values are written to the database.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p> <p>Note: When defining Value List and Display List properties, you should avoid using the following reserved characters:</p> <ul style="list-style-type: none"> • semicolon (;) • tab key • newline (carriage return) • form feed • backspace • equal sign (=) <p>Important: Do not use keys (like backspace), as the system cannot process their ASCII representation entries correctly which causes</p>

Property	Usage
	unpredictable results.
Display List Condition	Specify an expression to override the Display List property, when the expression evaluates to True.
Box Lines	Specify the window size of the drop-down list.
Select Only	<p>Selecting the check box requires the user to select from the Combo Box drop-down list. If the user makes a manual entry, the value must already exist in the drop-down list (defined in the Value List and Display List properties).</p> <p>Clearing the check box allows the user to select from the drop-down list or to manually enter a value.</p> <p>The Select Only Forms Designer property for the comfill widget is not enforced in the Web client. Users can enter free-formed text even with the property checked.</p>
Input Conversion	Specify the name of an Input Conversion routine. This property is used by RAD subroutines that mask the display of data on an input field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Mask	Specify the mask value (the condition that sets a value to be passed for this field to the input and output conversion routines). When a field has a mask value and you save the record, the format of the mask value displays in the field. For example, you decide to update the purchase cost information of a contract in Configuration Management. You enter a date in the Currency EX Date field of the Financial tab as 02/26/09. The condition for the mask value is set as: xxx dd,yyyy . When you save the record, the Currency Ex Date field displays the date format of February 26, 2009. Valid values for the mask vary depending on the routine being called.
Output Conversion	Specify the name of an Output Conversion routine. This property is used by RAD subroutines that mask the display of data on an output field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Combo Button Visible	Select this option to make the Combo Button visible when the form opens.
Combo Button Visible Condition	Specify an expression to override the ComboButton Visible property when that expression evaluates to true.

Property	Usage
Fill Button ID	Specify a Control ID to transmit when clicked. For example, 3 corresponds to F3 , and 0 corresponds to Enter .
Fill Button Image	Specify an icon to display on the field, if desired. You can select from 1 to 17. The default value is 7.
Fill Button Visible	Select this option to make the Fill Button visible when the form opens.
Fill Button Visible Condition	Specify an expression to override the Fill Button Visible property when that expression evaluates to true.
Third Button ID	Specify a Control ID to transmit when clicked. For example, 3 corresponds to F3 , and 0 corresponds to Enter .
Third Button Image	Specify an icon to display on the field, if desired. You can select from 1 to 17. The default value is 10.
Third Button Visible	Select this option to make the Third Button visible when the form opens.
Third Button Visible Condition	Specify an expression to override the Third Button Visible property when that expression evaluates to true.
Popup Subform Format	Specify the form to display.
Popup Subform Input	Specify the database field or variable to associate with this control.
Popup Subform Enabled	Select this option to enable the Popup Subform for this control.
Popup Subform Enabled Condition	Specify an expression to override the Popup Subform Enabled property when that expression evaluates to true.
Auto Complete	<p>Enable the auto complete feature for this control.</p> <p>Note: This setting takes effect only when the comfillAutoComplete parameter is set to true. Auto complete is supported only for Service Manager 9.34 or later versions.</p>

Convert Form Layout tool

This button converts all of the fields on the left side of the form into one group, and converts each of the notebook tabs on the right into groups, arranged vertically, under the left side fields. These groups are collapsible when displayed in the Web client.

When you click this button, a dialog box will display to inform you that the action will modify the form layout to use collapsible sections and that this cannot be undone after you have saved the form. If you

click **Yes** to proceed, the new layout is displayed. All of the fields and controls that were on the left side of the form are now contained in one Group. Clicking at the top of the group displays the Properties for the group. You will see that the following checkboxes are already selected:

- **Floating group enabled:** The group to change its relative position on the form so that it is not hidden when sections above it expand downward, and does not leave a blank spot after being collapsed.
- **Collapse enabled:** The group is displayed as a collapsible group that can be expanded and collapsed by the user.
- **Default to Expanded:** The collapsible group is expanded (open), by default. The user can close it if desired.

You can scroll down to see that the fields that were previously contained in the first (leftmost) notebook tab are also now contained in a Group. The Default to Expanded checkbox is not selected, so that this group will display as collapsed by default, until the user opens it.

If you scroll down further you will see the contents of each of the notebook tabs, previously displayed from left to right, are now displayed as groups, from top to bottom.

If you want to change the order of the sections, you can do so by changing the Y coordinate, which controls the vertical placement of the group on the form.

Date control

Use this control to add a text box that displays and enables users to update the contents of a date field.

To place a date field on a form, click **Date** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check

Property	Usage
	box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Input Conversion	Specify the name of an Input Conversion routine. This property is used by RAD subroutines that mask the display of data on an input field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Mask	Specify the mask value (the condition that sets a value to be passed for this

Property	Usage
	field to the input and output conversion routines). When a field has a mask value and you save the record, the format of the mask value displays in the field. For example, you decide to update the purchase cost information of a contract in Configuration Management. You enter a date in the Currency EX Date field of the Financial tab as 02/26/09. The condition for the mask value is set as: xxx dd,yyyy . When you save the record, the Currency Ex Date field displays the date format of February 26, 2009. Valid values for the mask vary depending on the routine being called.
Output Conversion	Specify the name of an Output Conversion routine. This property is used by RAD subroutines that mask the display of data on an output field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Parse	Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).
Mandatory	Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only. <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
Mandatory Condition	Specify an expression to override the Mandatory property when that expression evaluates to true.
Duration	Select this option to indicate that this is a time duration field rather than a specific date.
Time Zone	Specify the time zone for the date set in the duration property.
Time Zone Field	Specify the field that indicates the time zone for the date set in the duration property.
Data Changed Event	Specify the option number (such as Button ID) to call if the data contained in the object changed. This sends an event to the display RAD application.
Array Length	Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.

Property	Usage
	<ul style="list-style-type: none"> - If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. - If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. - If the Input data type is scalar, only a single text box appears. <p>Note: The screen object must be associated with an array data structure. The default is 0, which means one vertical line of information appears.</p>

Decimal control

Use this control to add an input field that accepts only numeric input and offers spinner buttons to increase or decrease a value.

To place a decimal field on a form, click **Decimal** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.

Property	Usage
<p>Accessible Name</p>	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
<p>Accessible Description</p>	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
<p>Tab Stop</p>	<p>Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.</p>
<p>Input Conversion</p>	<p>Specify the name of an Input Conversion routine. This property is used by RAD subroutines that mask the display of data on an input field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.</p>
<p>Mask</p>	<p>Specify the mask value (the condition that sets a value to be passed for this field to the input and output conversion routines). When a field has a mask value and you save the record, the format of the mask value displays in the field. For example, you decide to update the purchase cost information of a contract in Configuration Management. You enter a date in the Currency EX Date field of the Financial tab as 02/26/09. The condition for the mask value is set as: xxx dd,yyyy. When you save the record, the Currency Ex Date field displays the date format of February 26, 2009. Valid values for the mask vary depending on the routine being called.</p>
<p>Output Conversion</p>	<p>Specify the name of an Output Conversion routine. This property is used by RAD subroutines that mask the display of data on an output field, or check</p>

Property	Usage
	and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Decimals	Specify the number of decimal points for numbers in this field. Service Manager modifies numbers entered in the field to the number of decimal points you specify.
Parse	Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).
Mandatory	Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only. <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
Mandatory Condition	Specify an expression to override the Mandatory property when that expression evaluates to true.

Dynamic Form control

Use this control to add a dynamic form that becomes visible when populated by XML from a RAD application or JavaScript. The aspect and content of the dynamic form depend on the XML.

To place a dynamic form on a form, click **Dynamic Form** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

To view the dynamic form, open the form on which it is located with Database Manager. Each time you save changes in Forms Designer, exit the form and reload it in Database Manager.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.

Property	Usage
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.

Embedded Viewer control

Use this control to add an Embedded Viewer that enables Web client users to view integrations.

To place an Embedded Viewer on a form, click **Embedded Viewer** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component</p>

Property	Usage
	gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.
Frameborder	Specify whether a border is displayed around the embedded viewer. If you do not specify anything or enter 1, a border is displayed. If you enter 0, there is no border.
Height	Specify the height of the object in alignment grid units.
Input	Specify the database field or variable to associate with this control.
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Use client styling	Select this option to display content with the font, font size, margins and background color set in the client.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Width	Specify the width of the object in alignment grid units.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.

File control

Use this control to enable users to launch a File Open dialog box and upload or download files.

To place a file on a form, click **File** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional)

Property	Usage
	This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.

Property	Usage
Mandatory	Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only. <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
Mandatory Condition	Specify an expression to override the Mandatory property when that expression evaluates to true.
Transfer direction	Specify whether you want the user to be able to download or upload the file. The default is download. Note: If you do not have the transfer direction set properly, the user will get an error message and will not be able to upload/download the file. For example, if pressing the Specify File button gets the following error message, "File not found. Please verify the correct file name was given.", you may need to change the transfer direction from upload to download.
File Directory	Specify the absolute path to a directory. For example, c:/temp. If no directory is specified, the Service Manager RUN directory defaults.
File Type	Specify the MIME-type of the file the File Open dialog prompts for. For example, .txt for text files. If no MIME type is specified, the default is All Files (*.*)).
Title	Specify the name of the file you want to transfer.

Frame control

Use this control to add a container that creates a cosmetic rectangular border. Use bevel frames to make your forms more attractive or easier to understand.

To place a frame on a form, click **Frame** and then click the form.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.

Property	Usage
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Background Color	Specify the Background Color. You must select the Opaque property check box for the Back Color to have an effect. The only Background Color available for a Windows client marquee is black.
Outer Bevel Width	Specify the amount, in points, to increase or decrease the standard font size.
Inner Bevel Width	Specify an expression to override the Font Increase property when that expression evaluates to true.

Graph control

Use this control to display a diagram of connected nodes representing a system of two or more interrelated things. To add a graph to a form, either click on the control in the toolbar and then click on the drawing canvas, or, drag and drop the control from the toolbar to the drawing canvas.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Input	Specify the database field or variable to associate with this control. The primary use for this property is to set the focus on the graph.
Accessible Description	Specify a component description for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio

Property	Usage
	buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.

Grid view tool

This button enables you to view the alignment grid while designing a form.

Click the **Grid Preview** icon to overlay the design area with a grid.

Click the icon again to remove the overlay.

Group control

Use this control to add a container that enables you to logically group associated items. In the Windows client the Group has a rectangular border with a text label at the top.

Note: In the Web client, a group can be displayed as a collapsible section; see the last three properties listed in the table below.

Caution: When designing forms with collapsible sections, use the following guidelines to avoid formatting problems when the form is displayed in the Web client.

- All groups on the form should be marked as Collapsible-enabled.
- All widgets must be placed in a collapsible group box; no widgets should remain floating independently on the form.
- Do not stack or mix collapsible group boxes with non-collapsible group boxes.

To place a group on a form, click **Group** and then click the form.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Foreground Color	<p>Select the text color from the drop-down list.</p> <div data-bbox="586 1129 1370 1549" style="background-color: #e0e0e0; padding: 10px;"> <p>Note: The foreground color property for group boxes is controlled by the current Microsoft Windows theme that you have selected. For example, when you select "Windows XP" for the theme and the color scheme is set to "Default (Blue)" or "Silver," the foreground color for group box captions is set to blue. If you select "Windows Classic" for the theme, the foreground color property and DVD condition can be programmatically controlled, based on the settings in Forms Designer. This applies to both the Windows client and the Web client when run in Internet Explorer. The theme selection in Forms Designer does not affect the Web client when run from a Firefox browser.</p> </div>
Foreground Color Condition	Specify a formula to evaluate the ForeColor property at run time. The result of this evaluation overrides the value assigned to the property.
Background Color	Specify the Background Color. You must select the Opaque property check box for the Back Color to have an effect. The only Background Color available for a Windows client marquee is black.
Background Color Condition	Specify an expression to override the BackColor property when that expression evaluates to true.

Property	Usage
Font	Select a font for the text of your label, table, timer, or marquee caption from the drop-down list.
Bold	Select this option to apply bold to the label, default text, frame caption, or marquee caption.
Bold Condition	Specify an expression to override the Bold property when that expression evaluates to true.
Italic	Select this option to apply italics to your label, default font, or caption.
Italic Condition	Specify an expression to override the Italic property when that expression evaluates to true.
Heading Level	<p>Specify the heading level for this control so that a corresponding HTML heading tag (<h1> to <h6>) is generated.</p> <p>Note: This function is only supported by the Web client.</p>
Justification	Specify the justification for the object label as Left, Right, or Center.
Font Increase	Specify the amount, in points, to increase or decrease the standard font size.
Font Increase Condition	Specify an expression to override the Font Increase property when that expression evaluates to true.
Array Length	<p>Specify the size of the scrolling region used to view array entries.</p> <p>A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none"> • If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. • If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. • If the Input data type is scalar, only a single text box appears. <p>Note: The screen object must be associated with an array data structure.</p> <p>The default is 0, which means one vertical line of information appears.</p>
Floating group enabled	This option enables the group to change its relative position on the form so that it is not hidden when sections above it expand downward, and does not leave a blank spot after being collapsed.

Property	Usage
	<p>Note:This option applies to the Web client only.</p>
Collapse enabled	<p>This option enables the group to be displayed as a collapsible section that can be expanded and collapsed by the user.</p> <p>Note:This option applies to the Web client only.</p>
Default to Expanded	<p>Displays a collapsible group as expanded, by default. The user can close it if desired.</p> <p>Note:This option applies to the Web client only.</p>

HTML Editor control

Use this control to add an HTML Editor that enables users to add standard HTML to the form. Service Manager displays the rendered HTML in the HTML Viewer.

To place an HTML Editor on a form, click **HTML Editor** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Important: For Accessibility, the HTML Editor must be the last widget in its form or page, because you will not be able to tab to any widgets placed under it.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that

Property	Usage
	expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Toolbar Configuration	<p>Enables you to select an HTML Editor Toolbar from the available toolbars.</p> <ul style="list-style-type: none"> - Standard includes options for the most common HTML tags. - Knowledge Management includes options for the HTML tags required for Knowledge Management plus the Standard tags. - Knowledge Management Extended includes options Knowledge Management development, plus the Knowledge Management and Standard tags.
Insert Image Option	This property is for internal Service Manager development use only. It functions properly only in association with appropriate application code.
Insert Attachment Option	This property is for internal Service Manager development use only. It functions properly only in association with appropriate application code.
Insert Link Option	This property is for internal Service Manager development use only. It functions properly only in association with appropriate application code.
Value List	<p>In conjunction with the Input property, defines how the Display List values are identified in the database. Value and Display Lists are entered using the Edit List dialog box. You can enter hard coded entries for each list, or you can supply a variable as the first and only entry. The run time values of the variable are used to populate these lists.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended</p>

Property	Usage
	to be used outside of the out-of-box Knowledge Management module.
Display List	<p>Specify the values that appear in the drop-down list at run time. There must be a one-to-one correspondence between the values for Value List and for Display List. If the Display List is the only populated property, the display values are written to the database.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p>

HTML Viewer control

Use this control to add an HTML Viewer that enables users to view the HTML created using the HTML Editor.

To place an HTML Viewer on a form, click **HTML Viewer** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Warning: Keep in mind that text you copy and paste from other sources may contain formatting tags, even if it looks like plain text.

Property	Usage
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Height	Specify the height of the object in alignment grid units.
HTML-Generating Script	A Javascript function that generates the content for the HTML Viewer. For example, open the form "configurationItem" in Forms Designer. On the "CI Changes" notebook page there is an HTML Viewer which has "configurationManagement.getPendingChanges" for that property. To see this function, type SL in the command line, and then type

Property	Usage
	configurationManagement in the Name field. Click Search , and then scroll down. You will see the "getPendingChanges" function.
Input	Specify the database field or variable to associate with this control.
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Use client styling	Select this option to display an HTML snippet. The viewer will display the text respecting the font, font size, margins and background color set in the client. A fully defined HTML document may not display correctly. Note: Mark the checkbox for Service Catalog. Clear the checkbox to display a fully defined HTML document. A fully defined HTML document contains <html><head></head><body></body></html>. If the input does not contain any HTML formatting it will display plain text. Note: Clear the checkbox for Knowledge Management.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Width	Specify the width of the object in alignment grid units.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Frameborder	Specify whether a border is displayed around the HTML viewer. If you do not specify anything or enter 1, a border is displayed. If you enter 0, there is no border.
Standards Mode	Select this option if you want the web browser to render the HTML document in Standards Mode. When this option is selected, a DOCTYPE is added to the beginning of the document.

Image control

Use this control to add an image from a Microsoft Windows image file.

To place an image on a form, click **Image** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Image	Specifies the name of an image file to display on the control. The image file must reside in any of the following paths, in which Service Manager looks for the image: Windows client: <ul style="list-style-type: none"> • <Service Manager>\Client\plugins\com.hp.ov.sm.client.common_x.xx.xxx\src\resources\icons • A local directory on the client machine, which is specified in the Image path field (Window > Preferences). For example: C:\smCustomImages. Web client:

Property	Usage
	<ul style="list-style-type: none"> <Service Manager web tier>\images. For example: <Tomcat>\webapps\webtier-x.xx\images. <p>Note: When you specify the image file for this property, you need to provide both the file name and the file name extension.</p>
Image Condition	Specify an expression to override the Image property when that expression evaluates to true.

Label control

Use this control to add a label. A label is a single line of text you can use to give titles to forms, give labels to objects within the form, or otherwise place text on the form.

To place a label on a form, click **Label** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Best practices

- Use black for label text.
- Left-align label text and end it with a colon.
- Do not create colored text "drop shadows." These do not render well in the Web client.
- Labels on menu forms (for example, menu.gui.home) should have a font increase of 2.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.

Property	Usage
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Label for	Specify the name of a control that uses this label. Note: If you specify a variable for this label, you need to provide a valid value here. Otherwise, the label cannot be read by JAWS.
Foreground Color	Select the text color from the drop-down list.
Foreground Color Condition	Specify a formula to evaluate the ForeColor property at run time. The result of this evaluation overrides the value assigned to the property.
Background Color	Specify the Background Color. You must select the Opaque property check box for the Back Color to have an effect. The only Background Color available for a Windows client marquee is black.
Background Color Condition	Specify an expression to override the BackColor property when that expression evaluates to true.
Background Style	Applies to the display of the label in the Web client only. Select Underline to display the label as underlined text. Select Solid Background to display the label background as a solid colored bar.
Font	Select a font for the text of your label, table, timer, or marquee caption from the drop-down list.
Bold	Select this option to apply bold to the label, default text, frame caption, or marquee caption.
Bold Condition	Specify an expression to override the Bold property when that expression evaluates to true.
Italic	Select this option to apply italics to your label, default font, or caption.
Italic Condition	Specify an expression to override the Italic property when that expression evaluates to true.
Font Increase	Specify the amount, in points, to increase or decrease the standard font size.

Property	Usage
Font Increase Condition	Specify an expression to override the Font Increase property when that expression evaluates to true.
Array Length	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none"> • If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. • If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. • If the Input data type is scalar, only a single text box appears. <p>The default value is 0, which means one vertical line of information appears.</p> <p>Note: The screen object must be associated with an array data structure.</p>
Heading Level	<p>Specify the heading level for this control so that a corresponding HTML heading tag (<h1> to <h6>) is generated.</p> <p>Note: This function is only supported by the Web client.</p>
Justification	Specify the justification for the object label as Left, Right, or Center.

Link label control

Use this control to add a label that includes a caption and an icon that links to another source.

To place a link label on a form, click **Link label** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>
X	Specify the object's horizontal position based on the left edge of the object.

Property	Usage
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Button ID	Specify a numeric identification that specifies a Control ID to transmit when

Property	Usage
	clicked.
Button ID Condition	Specify an expression to override the Button ID property when that expression evaluates to true.
Image	<p>Specifies the name of an image file to display on the control. The image file must reside in any of the following paths, in which Service Manager looks for the image:</p> <p>Windows client:</p> <ul style="list-style-type: none"> <Service Manager>\Client\plugins\com.hp.ov.sm.client.common_x.xx.xxx\src\resources\icons A local directory on the client machine, which is specified in the Image path field (Window > Preferences). For example: C:\smCustomImages. <p>Web client:</p> <ul style="list-style-type: none"> <Service Manager web tier>\images. For example: <Tomcat>\webapps\webtier-x.xx\images. <p>Note: When you specify the image file for this property, you need to provide both the file name and the file name extension.</p>
Description	Specify the description text for the label.
Balloon Help	Specify the text string to display when the cursor is held over the button to offer more information about the button's function.
Enabled	Specify an initial state.
Enabled Condition	Specify a condition that enables the button when a table row is selected. Example: [%tablename] == "selected" or [%tablename] != "selected" where tablename is the value of the Name property of the table on the screen.

List Builder control

Use this control to add a compound widget that allows the user to construct a list of choices.

To place a List Builder on a form, click **List Builder** and then click the form.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>

Property	Usage
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Allow Duplicates	Click the box to allow duplicate items.
Choices Name	Specify a label for the Choices list.

Property	Usage
Display List	Specify the values that appear in the drop-down list at run time. There must be a one-to-one correspondence between the values for and for Display List. If the Display List is the only populated property, the display values are what is written to the database.
Display List Condition	Specify an expression to override the Display List property when that expression evaluates to true.
Selection Name	Specify a label for the Selections list.
Sortable	Click the box to allow reordering of the list items.
Value List	<p>In conjunction with the Input property, defines how the Display List values are identified in the database. Value and Display Lists are entered using the Edit List dialog box. You can enter hard coded entries for each list, or you can supply a variable as the first and only entry. The run time values of the variable are used to populate these lists.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p>
Value List Condition	Specify an expression to override the Value List property when that expression evaluates to true.

Marquee control

Use this control to add an attention-getting text message that scrolls continuously across the screen from right to left.

To place a marquee on a form, click **Marquee** and then click the form. Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Mozilla-based browsers do not support the marquee tag and thus cannot display them in Web client forms. You must either use an Internet Explorer browser or not use marquees in Web client forms.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.

Property	Usage
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Foreground Color	Select the text color from the drop-down list.
Background Color	Specify the Background Color. You must select the Opaque property check box for the Back Color to have an effect. The only Background Color available for a Windows client marquee is black.
Font	Select a font for the text of your label, table, timer, or marquee caption from the drop-down list.
Bold	Select this option to apply bold to the label, default text, frame caption, or marquee caption.
Italic	Select this option to apply italics to your label, default font, or caption.
Font Increase	Specify the amount, in points, to increase or decrease the standard font size.

Notebook Tab control

Use this control to add a tab to an existing notebook. Navigate to each page of the notebook by selecting its tab.

Click to the right of the last tab on the notebook to open the notebook's properties. Click the notebook tab and then the blank area below the tab to open the notebook tab's properties.

Best practices

Forms with tabs should include a header section which is not part of the tab. This header should include important description information about the record. For example, incident records should show, at a minimum, the incident number.

To place a tab on a notebook, click **Notebook tab** and then click the notebook.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.

Notebook control

Use this control to subdivide the contents of a screen into logical groups or categories. Notebooks provide an aesthetic way of organizing large amounts of data into small spaces.

Click to the right of the last tab on the notebook to open the notebook properties. Click the tab and then the blank area below the tab to open the notebook tab properties.

To place a notebook on a form, click **Notebook** and then click the form.

Best practice

Do not place notebooks next to other form elements or other notebooks.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional)

Property	Usage
	This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Group rendering enabled	Causes the notebook to be displayed as a group rather than a tabbed notebook. The contents of each tab are displayed as separate sections of the group. The notebook tabs that were arranged from left to right inside the notebook are displayed from top to bottom in the group.
Heading Level	Specify the heading level for this control so that a corresponding HTML heading tag (<h1> to <h6>) is generated. Note: This function is only supported by the Web client.

Radio button control

Use this control to add a radio button to a form. A radio button enables the user to select one value from a set of mutually exclusive values. The items in the set are associated with a database field or variable. Use radio buttons to give the user a quick way to choose one option from a set of fixed and mutually exclusive string values. It is a good practice to group related radio buttons within a frame.

Note: Radio buttons are not supported on record lists.

Best practices

- If the choice is logical (true or false), use a check box instead.
- If choices are extensive or if the user may need to enter a choice manually, use a Combo Box instead.

To place a radio button on a form, click **Radio button** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	Specify a component name for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.

Property	Usage
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Tab Stop	<p>Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.</p>
Foreground Color	<p>Select the text color from the drop-down list.</p>
Read-Only	<p>Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.</p>
Read-Only Condition	<p>Specify an expression to override the Read-Only property when that expression evaluates to true.</p>
Mandatory	<p>Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only.</p> <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
Mandatory Condition	<p>Specify an expression to override the Mandatory property when that expression evaluates to true.</p>
Data Changed Event	<p>Specify the option number (such as Button ID) to call if the data contained in the object changed. This sends an event to the display RAD application.</p>
Value	<p>Specify the value to assign to the field or variable specified in the Input property when a user selects this radio button.</p>

Selection tool

This button enables you to select, deselect, and modify design objects.

To select a control, click **Selection tool** and then click the control.

Switch from selection mode to creation mode

If you want to create multiple instances of an object, hold down the **Ctrl** key and click anywhere on the drawing canvas.

Select one object

Click on the object. Grab points are displayed indicating that the object is selected.

Or, hold down the left mouse button and drag a lasso across any part of the object. After lassoing the object, handles are displayed.

Select multiple objects

Hold down the **Ctrl** key and click on multiple objects one at a time. Grab points are displayed on each object. The properties view displays the common properties of the objects selected.

Or, drag and release a lasso across any part of the desired objects. After lassoing the objects, handles are displayed on each object.

Clear one object

Hold down the **Ctrl** key and click on the object. Grab points disappear, indicating that the object is no longer selected.

Clear multiple objects

Click on a blank area of the drawing canvas.

Position one object

Click on the object and continue holding down the left mouse button. Drag the object to the desired position.

Position multiple objects

Hold down the **Ctrl** key and one at a time, click on the multiple objects you want to position. The movement cursor appears when you hover over the selected objects.

At this point you can drag and drop the objects as a group anywhere on the drawing canvas.

Or, hold down the **Ctrl** key and one at a time, click on the multiple objects you want to position. Then enter the X or Y coordinate for all the items in the properties editor. For example, if you want to position multiple objects with the same left border, set the X property. The Y property can be used in a similar way to position objects with the same top border.

Resize an object

Select the object. Drag the object handles to resize the object.

Delete one or more objects

Select the object(s). Press the **Delete** key.

Script control

Use this control to add a color-coded syntax editor to a form. This enables the user to enter JavaScript.

To place a script on a form, click **Script** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Accessible Name	Specify a component name for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.
Accessible Description	Specify a component description for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility

Property	Usage
	software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.
Script	Specify the type of script. The default is <i>js</i> for JavaScript.

Subform control

Use this control to display a subform on another form.

Subforms displayed on forms have full functionality. Subforms displayed as pop-ups are read-only; users cannot interact with them. To display the subform as a pop-up, use the Comfill control instead.

To place a subform on a form, click **Subform** and then click the form.

Put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Format	Specify the form to display. Be sure you type the exact form name.
Virtual Join	Select this option to associate virtual join run time processing with the subform object. Clear the check box to use a same-file join.
Display Blank	Select this option to display the subform even if no records were found for the virtual join. Clear the check box to skip display of the format if there are no

Property	Usage
	matching records. Note: Used only for virtual joins.
Display Using Table	Select this option to display the subform in table format.
Input	Specify the database field or variable to associate with this control.

Table column control

Use this control to add a column to an existing table.

Note: To avoid unpredictable behavior, do not set a DVD condition to use a variable that is based on a record displayed in the Table widget.

To add a column to a table, click **Table column** and then click the table.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Best practice

Column headings should follow book title capitalization rules, for example, File Name not file name or file.name.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.

Property	Usage
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Field	In the case where the array specified in the input property is an array of structure, it specifies the field in the structure for this object.
Column Width Percent	Enables you to set the column width to a percentage of the page.
Minimum Column Width	Enables you to set the minimum width of the column in a table.
Clickable Title	Select this option to allow the user to click the title.
Button ID	When an individual column Button ID is set to 0, the Default Button ID is used.
Show Title	Select this option to make the title appear in the table.
Value List	In conjunction with the Input property, defines how the Display List values are identified in the database. Value and Display Lists are

Property	Usage
	<p>entered using the Edit List dialog box. You can enter hard coded entries for each list, or you can supply a variable as the first and only entry. The run time values of the variable are used to populate these lists.</p> <p>This property is supported only in association with the toolbar configuration Knowledge Management Extended, which is not intended to be used outside of the out-of-box Knowledge Management module.</p>
<p>Image File</p>	<p>Specifies the name of an image file to display on the control. The image file must reside in any of the following paths, in which Service Manager looks for the image:</p> <p>Windows client:</p> <ul style="list-style-type: none"> • <Service Manager>\Client\plugins\com.hp.ov.sm.client.common_x.xx.xxx\src\resources\icons • A local directory on the client machine, which is specified in the Image path field (Window > Preferences). For example: C:\smCustomImages. <p>Web client:</p> <ul style="list-style-type: none"> • <Service Manager web tier>\images. For example: <Tomcat>\webapps\webtier-x.xx\images. <p>Note: You do not need to include the image file suffix for this property; the image file name must be unique.</p>
<p>Display List</p>	<p>Specify the values that appear in the drop-down list at run time. There must be a one-to-one correspondence between the values for Value List and for Display List. If the Display List is the only populated property, the display values are what is written to the database.</p>
<p>Select Only</p>	<p>Setting Select Only to Yes by checking the box requires the user to select from the Combo Box drop-down list. If the user makes a manual entry, the value must already exist in the drop-down list (defined in the Value List and Display List properties).</p> <p>Setting it to No by leaving the box unchecked allows the user to select from the drop-down list or to manually enter a value. No works with or without a drop-down list as defined in the and Display List properties.</p>
<p>Read-Only</p>	<p>Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.</p>
<p>Maximum Chars</p>	<p>Specify the number of characters the user can enter in the field. The default is 0 (unlimited).</p>

Property	Usage
Maximum Characters Beep	When set to Yes by checking the box, it notifies the user by sounding a beep that they reached the maximum number of characters. Not available on UNIX systems.
Case Conversion	Specify how to change the case of text entered in the field. The choices are None, Upper, Lower.
Decimals	Specify the number of decimal points for numbers in this field. Service Manager modifies numbers entered in the field to the number of decimal points you specify.
Parse	Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).
Justification	Specify the justification for the object label as Left, Right, or Center.
Input Conversion	Specify the name of an Input Conversion routine. This property is used by RAD subroutines that mask the display of data on an input field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Mask	Specify the mask value (the condition that sets a value to be passed for this field to the input and output conversion routines). When a field has a mask value and you save the record, the format of the mask value displays in the field. For example, you decide to update the purchase cost information of a contract in Configuration Management. You enter a date in the Currency EX Date field of the Financial tab as 02/26/09. The condition for the mask value is set as: xxx dd,yyyy . When you save the record, the Currency Ex Date field displays the date format of February 26, 2009. Valid values for the mask vary depending on the routine being called.
Output Conversion	Specify the name of an Output Conversion routine. This property is used by RAD subroutines that mask the display of data on an output field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Password	Select this option to make the text entered in the field a series of asterisks on the screen. The data is not encrypted, just transposed.

Table control

Use this control to add a table that displays one or more columns of data in a scrollable pane. The tables you create in the Form Designer appear as a rectangular region subdivided by rows and columns. The look and feel mimics tables in Microsoft Office applications, such as Excel.

Table have the following features:

- Cells may display text or images.
- Columns may vary in width. The user can resize the columns using the mouse.
- Columns may have an optional heading button, which can be mapped to perform an action.
- All rows are the same height, defined by the table's font.
- The user can scroll through data by use of scroll bars or the keyboard.
- All expected keyboard navigation is supported (Up, Down, Next, Prev, Last, First, and so forth)
- You can edit cells or make them read-only.
- Selecting a read-only cell results in a highlight of the entire row.
- Selecting an cell to edit results in the in-place creation of an edit field or drop-down Combo Box that lets the user edit the cell.
- Users can edit a field by clicking on it or by pressing the **Enter** key while the table has focus. Tabbing events received while the table is in edit mode result in navigation from one cell to the next. Pressing **Enter** while a cell is being edited results in the changes being accepted and the cell's row being highlighted. Tabbing events received when a row is highlighted are handled by passing focus to the next object in the window.

Note: To avoid unpredictable behavior, do not set a DVD condition to use a variable that is based on a record displayed in the Table widget. The only exception is the Foreground Color Condition property, which takes effect at the record level.

To place a table on a form, click **Table** and then click the form.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional)

Property	Usage
	This field is required if you enable the Multiple Selection property. This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Font	Select a font for the text of your label, table, timer, or marquee caption from the drop-down list.
Foreground Color Condition	Specify a formula to evaluate the ForeColor property at run time. The result of this evaluation overrides the value assigned to the property.
Bold Condition	Specify an expression to override the Bold property when that expression evaluates to true.
Italic Condition	Specify an expression to override the Italic property when that expression evaluates to true.
Font Increase	Specify the amount, in points, to increase or decrease the standard font size.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Columns	Specify a list of columns in the table.

Property	Usage
Column Headings	Select this option to make the column headings appear above the columns.
Multiple Selection	<p>Select this option to allow the user to select several rows in the table. Clear the check box to allow the user to select just one row.</p> <div data-bbox="586 470 1369 896" style="background-color: #f0f0f0; padding: 10px;"> <p>Note:</p> <ul style="list-style-type: none"> If you enable this option, you must enter the associated table's name in the Name property. The Multiple Selection property in Forms Designer has no effect on enabling or disabling the multi-select capability when performing a fill from an array field. The Multiple Selection functionality applies to multiple rows in a single table only. Even if this option is enabled, selecting multiple rows in multiple tables will lead to unexpected results. </div>
Selection Field	Select this option to allow the user to select several rows in the table. Clear the check box to allow the user to select just one row.
Default Button ID	When an individual column Button ID is set to 0, the Default Button ID is used.
Double-Click Button ID	Specify the Button ID returned by the double-click action.
Double-Click Field	Specify the field to return to <code>cursor.field.contents()</code> which can then be used in a query to find the record.
Accessible Description	<p>Specify a component description for use with accessibility software (JAWS). (Optional)</p> <div data-bbox="586 1346 1369 1415" style="background-color: #f0f0f0; padding: 10px;"> <p>Note: This function is supported only by the web client.</p> </div> <p>If no description is present, the Web client uses the default table summary as described below:</p> <ul style="list-style-type: none"> For a record list on a list page or on the list panel of a list-detail page: Record List + [grouped by] + [sorted by] + [ascending/descending]. Example: Record List Sorted by Update Time Ascending 11x51 For a Table widget on a detail form: Table. Example: Table 4x3

Text area control

Use this control to add a text area that displays the contents of a text field and conditionally enables users to input several lines of data. This object contains scroll bars and allows text wrapping.

Note: A field associated to a text area must be defined as an array of characters in the dbdict.

To place a text area on a form, click **Text area** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	Specify a component name for use with accessibility software. (Optional) Note: This function is only supported by the Web client. If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software

Property	Usage
	<p>can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
<p>Accessible Description</p>	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
<p>Tab Stop</p>	<p>Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.</p>
<p>Read-Only</p>	<p>Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.</p>
<p>Read-Only Condition</p>	<p>Specify an expression to override the Read-Only property when that expression evaluates to true.</p>
<p>Mandatory</p>	<p>Select this option so that a red asterisk indicating a required field is displayed. This is a visible change only.</p> <ul style="list-style-type: none"> - To make the field mandatory for any form it appears on, use the System Definition Utility. - To make the field mandatory for a small number of forms only, use Format Control.
<p>Mandatory Condition</p>	<p>Specify an expression to override the Mandatory property when that expression evaluates to true.</p>
<p>Password</p>	<p>Select this option to make the text entered in the field a series of asterisks on the screen. The data is not encrypted, just transposed.</p>
<p>Maximum Chars</p>	<p>Specify the number of characters the user can enter in the field. The default is 0 (unlimited).</p>
<p>Maximum Characters Beep</p>	<p>Select this option to notify the user by sounding a beep that they</p>

Property	Usage
	reached the maximum number of characters. This is not available on UNIX systems.
Case Conversion	Specify how to change the case of text entered in the field. The choices are None, Upper, Lower.
Decimals	Specify the number of decimal points for numbers in this field. Service Manager modifies numbers entered in the field to the number of decimal points you specify.
Parse	Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).

Text control

Use this control to add a text box that displays the contents of a text field and conditionally enables users to enter or modify its contents.

To place a text box on a form, click **Text** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Caption	Specify a text caption for the object.

Property	Usage
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Accessible Name	<p>Specify a component name for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no name is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, it uses the component's text. Accessibility software can query the Web client to get this information and use it to present data to the user in a variety of ways. For example, speech simulation software says the name and the type of component when the component has focus.</p>
Accessible Description	<p>Specify a component description for use with accessibility software. (Optional)</p> <p>Note: This function is only supported by the Web client.</p> <p>If no description is present, the Web client defaults to looking for a nearby label to use for the name of the component. For buttons, check boxes, or radio buttons, the Web client will use the component's text. Accessibility software can query the Web client to get this information, and use it to present it to the user in a variety of ways. For example, when a component gets focus, speech simulation software will say the name and the type of component, based on the settings in the Accessible Name and Accessible Description fields.</p>
Tab Stop	Specify the tab sort order for this field. By default, the TAB key navigates the focus on a form from top to bottom and left to right. Objects that have a non-zero Tab Stop are visited first, in ascending order. For a button, file, radio button or check box, you will almost always use the default of 0.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.
Mandatory	Setting Mandatory to Yes by checking the box indicates that the field is required and therefore a red asterisk indicating a required field is displayed. This is a visible change only. To make the field mandatory for any form it appears on, use the System Definition Utility. To make the

Property	Usage
	field mandatory for a small number of forms only, use Format Control.
Mandatory Condition	Specify an expression to override the Mandatory property when that expression evaluates to true.
Password	Select this option to make the text entered in the field a series of asterisks on the screen. The data is not encrypted, just transposed.
Maximum Chars	Specify the number of characters the user can enter in the field. The default is 0 (unlimited).
Maximum Characters Beep	Select this option to notify the user by sounding a beep that they reached the maximum number of characters. This is not available on UNIX systems.
Case Conversion	Specify how to change the case of text entered in the field. The choices are None, Upper, Lower.
Decimals	Specify the number of decimal points for numbers in this field. Service Manager modifies numbers entered in the field to the number of decimal points you specify.
Parse	Select this option to verify that the text entered in a field is the correct syntax. The syntax for the field is defined by the type of field it displays (date, expression, number, and so forth).
Array Length	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none"> - If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. - If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. - If the Input data type is scalar, only a single text box appears. <p>Note: The screen object must be associated with an array data structure.</p> <p>The default is 0, which means one vertical line of information appears.</p>
Input Conversion	Specify the name of an Input Conversion routine. This property is used by RAD subroutines that mask the display of data on an input field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Mask	Specify the mask value (the condition that sets a value to be passed for this field to the input and output conversion routines). When a field has

Property	Usage
	a mask value and you save the record, the format of the mask value displays in the field. For example, you decide to update the purchase cost information of a contract in Configuration Management. You enter a date in the Currency EX Date field of the Financial tab as 02/26/09. The condition for the mask value is set as: xxx dd,yyyy . When you save the record, the Currency Ex Date field displays the date format of February 26, 2009. Valid values for the mask vary depending on the routine being called.
Output Conversion	Specify the name of an Output Conversion routine. This property is used by RAD subroutines that mask the display of data on an output field, or check and validate the entry of data into an input field. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.

Timer control

Use this control to add a clock timer to the form. The timer initiates events at a certain time, based on settings in the Input and Expiration Event fields.

To place a clock timer on a form, click **Timer** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.

Property	Usage
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Font	Select a font for the text of your label, table, timer, or marquee caption from the drop-down list.
Foreground Color	Select the text color from the drop-down list.
Foreground Color Condition	Specify a formula to evaluate the ForeColor property at run time. The result of this evaluation overrides the value assigned to the property.
Background Color	Specify the Background Color. You must select the Opaque property check box for the Back Color to have an effect. The only Background Color available for a Windows client marquee is black.
Background Color Condition	Specify an expression to override the BackColor property when that expression evaluates to true.
Bold	Select this option to apply bold to the label, default text, frame caption, or marquee caption.
Bold Condition	Specify an expression to override the Bold property when that expression evaluates to true.
Italic	Select this option to apply italics to your label, default font, or caption.
Italic Condition	Specify an expression to override the Italic property when that expression evaluates to true.
Font Increase	Specify the amount, in points, to increase or decrease the standard font size.
Font Increase Condition	Specify an expression to override the Font Increase property when that expression evaluates to true.
Output Conversion	Specify the name of an Output Conversion routine. If specified, the routine is called at runtime to convert the contents of the field before it is displayed on the screen. For examples, refer to the RAD input and output routines.
Array Length	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none"> - If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. - If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array.

Property	Usage
	<p>- If the Input data type is scalar, only a single text box appears.</p> <p>Note: The screen object must be associated with an array data structure.</p> <p>The default is 0, which means one vertical line of information appears.</p>
Expiration Color	Specify the color that the timer should change to when it reaches zero.
Expiration Event	Specify the event to send to the server when the timer reaches zero. Specify a numeric value; for example, a button or event ID.

Visualization control

Use this control to graphically display Configuration Item (CI) relationships.

Property	Usage
Name	<p>Specify a unique identifier for the object on the screen. (Optional)</p> <p>This name is used by external applications, such as RAD, to dynamically change the properties of the object.</p>
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Read-Only	Select this option to disable editing capabilities and provide only viewing access to the field. Give read-only fields a tab-stop value of -1 to prevent users from tabbing into them.
Read-Only Condition	Specify an expression to override the Read-Only property when that expression evaluates to true.

Web Preview tool

This button enables you to see a preview of what the form will look like when displayed by the Web client. The preview does not take into account any active DVD on the form.

Click **Web Preview** to display the form in the Web Preview tab.

Wrap label control

Use this control to add a multi-line label. You can use this label to give a title to the form, give labels to objects within the form, or otherwise place text on the form.

To place a wrapping label on a form, click **Wrap label** and then click the form.

Be sure to put the name of the database field or variable that you want to associate with this control into the **Input** property.

Property	Usage
Name	Specify a unique identifier for the object on the screen. (Optional) This name is used by external applications, such as RAD, to dynamically change the properties of the object.
X	Specify the object's horizontal position based on the left edge of the object.
Y	Specify the object's vertical position based on the top edge of the object.
Width	Specify the width of the object in alignment grid units.
Height	Specify the height of the object in alignment grid units.
Visible	Select this option to make the object visible on the form. Clear the check box to hide the object from view on the form.
Visible Condition	Specify an expression to override the Visible property when that expression evaluates to true.
Caption	Specify a text caption for the object.
Caption Condition	Specify an expression to override the Caption property when that expression evaluates to true.
Input	Specify the database field or variable to associate with this control.
Label for	Specify the name of a control that uses this label.

Property	Usage
	<p>Note: If you specify a variable for this label, you need to provide a valid value here. Otherwise, the label cannot be read by JAWS.</p>
Foreground Color	Select the text color from the drop-down list.
Foreground Color Condition	Specify a formula to evaluate the ForeColor property at run time. The result of this evaluation overrides the value assigned to the property.
Background Color	Specify the Background Color. You must select the Opaque property check box for the Back Color to have an effect. The only Background Color available for a Windows client marquee is black.
Background Color Condition	Specify an expression to override the BackColor property when that expression evaluates to true.
Font	Select a font for the text of your label, table, timer, or marquee caption from the drop-down list.
Bold	Select this option to apply bold to the label, default text, frame caption, or marquee caption.
Bold Condition	Specify an expression to override the Bold property when that expression evaluates to true.
Italic	Select this option to apply italics to your label, default font, or caption.
Italic Condition	Specify an expression to override the Italic property when that expression evaluates to true.
Font Increase	Specify the amount, in points, to increase or decrease the standard font size.
Font Increase Condition	Specify an expression to override the Font Increase property when that expression evaluates to true.
Array Length	<p>Specify the size of the scrolling region used to view array entries. A scroll bar appears beside the fields to allow users to view the array entries.</p> <ul style="list-style-type: none"> • If a field is assigned an Array Length of 5, the form stacks five fields vertically to allow users to view the five array entries. • If Array Length is set to -1, the form dynamically creates as many fields as there are entries in the array. • If the Input data type is scalar, only a single text box appears. <p>The default value is 0, which means one vertical line of information appears.</p>

Property	Usage
	Note: The screen object must be associated with an array data structure.
Heading Level	Specify the heading level for this control so that a corresponding HTML heading tag (<h1> to <h6>) is generated. Note: This function is only supported by the Web client.

Enabling HTML in forms

Forms Designer has an HTML Editor and an HTML Viewer that you can add to forms.

- The **HTML Editor** enables users to add and edit HTML text in Service Manager forms. It uses standard HTML formatting.
- The **HTML Viewer** enables users to view (but not edit) the formatted HTML text entered by using the HTML Editor.

Using the HTML Editor

To add HTML, simply type or paste text into the HTML Editor and format it as desired. The Knowledge Management and Knowledge Management Extended toolbars enable you to switch between source and WYSIWYG mode. If you have a Knowledge Management toolbar, you can view the HTML source code and edit it directly.

Note: While in WYSIWYG mode, the editor makes choices about how to apply the tags you select. If you want more control over how the tags are applied, ask your administrator for a Knowledge Management toolbar and use source mode for editing.

In the Windows client and web client, press Enter for a new paragraph <p> and press Shift+Enter for a line break
. To exit a numbered or bulleted list and start a new paragraph <p>, you can press Enter twice.

The HTML Viewer will render valid HTML, but you must ensure that the HTML you enter is valid and is properly integrated into the HTML document. For example:

- If you paste in HTML containing a JavaScript requiring a file provided by another application, that JavaScript will not work in the HTML Viewer.
- If you copy an HTML form with buttons, the form will render, but the buttons will not work in the HTML Viewer.

In the Windows client, if you paste text from a Microsoft Word document into the HTML Editor, the editor opens a window that allows you to select options to clean the HTML work created.

Add an HTML Editor or Viewer to a form

Applies to User Roles:

System Administrator

To Add an HTML Editor or Viewer to a form:

1. Click **Tailoring > Forms Designer** in the System Navigator, or type `fd` on the Service Manager command line and then press **Enter**.
2. Type a form name and click **Search**.

Service Manager opens a list of forms. Click the form you want to update to open it.

3. Click **Design** to open design mode.
4. Drag the HTML Editor icon to the desired location.

–or–

Drag the HTML Viewer icon to the desired location.
5. Type the name in the Value text box of the Name property.
6. Specify the field or variable in which to store the HTML in the Value text box of the Input property.
7. Specify the height, width, and other properties of the HTML Editor or viewer, as desired.
8. Click **Web Preview** to see what the form will look like in the Web client.
9. Click **OK** to see what the form will look like in the Windows client.
10. Click **OK** to save your changes.

11. Open the form using Database Manager to test your changes:

a. Click **Tailoring > Database Manager**

b. Click **Search**.

Service Manager opens a list of forms.

c. Click the form you just created to open it.

Enable HTML Editor whitelist

The HTML Editor enables users to visually add and edit rich text content formatted with HTML tags. Though the HTML Editor is powerful, web sites may be abused without a proper security control. Therefore, as of HP Service Manager 9.41, you can use the HTML Editor whitelist to define a list of allowed HTML elements.

To enable the HTML Editor whitelist and define a list of allowed HTML elements, follow these steps:

1. Log on to Service Manager as a system administrator.
2. Click **Tailoring > HTML Editor Whitelist**.
3. Select the **HTML Editor Whitelist** check box. By default, this check box is cleared.
4. Define a list of allowed HTML Tags and attributes in the **Allowed Tags/Attributes** section. For example, add `script` as an allowed tag so that you are able to input the `<script>` tag by using the HTML Editor.
5. Define a list of allowed URL protocols for some elements in the **Allowed URL Protocols** section. For example, add `a` as an allowed tag, specify `href` as the attribute, and then specify one or more protocols that are used by the href attribute, such as `http,https,mailto` and `ftp`.
6. Click **Save** and **OK**.
7. Do either of the following to make the customized HTML Editor whitelist effective:
 - Restart the web application server if you are working with the Service Manager web client.
 - Log out and then log on to the Service Manager Windows client again.

After the HTML Editor whitelist became effective, you are only allowed to input the elements as listed in the whitelist when editing HTML contents using the HTML Editor. You need to click the **HTML validate**

button to validate your input before saving your changes. If there are tags, attributes or protocols that violate the whitelist, the system displays some warning messages. For example:

The HTML document contains some tags `<script>`,`<javascript>` that are not allowed by the system. Contact your Service Manager system administrator for assistance.

The HTML document contains some attributes for tags (`<tag attribute...>`) `<div onclick>`,`<p onerror>` that are not allowed by the system. Contact your Service Manager system administrator for assistance.

The HTML document contains some values in the attributes of tags (`<tag attribute>`) ``,`<a href>` that are not allowed by the system. Contact your Service Manager system administrator for assistance.

You must review your input and avoid using these invalid tags, attributes, or protocols. Alternatively, you can ask your Service Manager system administrator to include these tags, attributes or protocols to the whitelist. Otherwise, the system automatically removes these invalid elements from your input after you click **Save** in the HTML editor.

Note:















The **HTML validate** button is visible only when you have installed the Service Manager 9.41 components (including server, client, and applications) and enabled the HTML Editor whitelist.

















The **HTML validate** button is available in both the source mode and the wysiwyg mode of the HTML Editor on the web client. However, this button is only available in the wysiwyg mode of the HTML Editor on the Windows client. When you use the source mode to edit an HTML document on the web client, it is very convenient to click the **HTML validate** button to validate your input. However, you need to switch to the wysiwyg mode on the Windows client to validate the HTML document before saving it.


Caution: HP Service Manager 9.41 introduces the HTML Editor whitelist solution and provides a default whitelist for tags, attributes or protocols that are allowed in Service Manager out-of-box knowledge documents. HP recommends you to enable this enhancement. However, your HTML documents may contain tags, attributes, or protocols that are not defined in the whitelist before the 9.41 release. HP recommends the Service Manager system administrators to perform extensive tests in the development environment and add all necessary tags, attributes, or protocols to the whitelist before enabling the HTML Editor whitelist. Otherwise, enabling the HTML Editor whitelist without prior tests may cause data loss to the existing knowledge documents in the production environment.

HTML Editor controls

The HTML Editor has the following formatting controls. Their availability depends on the Keyboard property selected in the HTML Editor control on the form you are using.

Icon	Name	Function
	Source	Toggles between source mode and wysiwyg mode for editing.
	Preview	Opens the text in a browser for web preview.
	Spell checking	Checks the spelling of the entered text.
	Cut	Cuts the selected text.
	Copy	Copies the selected text.
	Paste	Pastes the copied or cut text.
	Undo	Revert to last saved record.
	Redo	Redoes the reverted action.
	Find	Finds the specified text.
	Replace	Replaces found text with the specified text.
	Select All	Selects entire document.
	Remove Format	Removes formatting from the selected text.
B	Bold	Makes the selected text bold.
<i>I</i>	Italic	Makes the selected text italic.
<u>U</u>	Underline	Underlines the selected text.
A B C	Strikethrough	Strikes through the selected text.
	Text color	Makes the selected text the specified color.
	Background Color	Sets the background of the selected text to the specified color.

Icon	Name	Function
	Numbering	Makes the selected paragraphs into a numbered list.
	Bullets	Makes the selected paragraphs into a bulleted list.
	Decrease Indent	Decreases the indent of the selected paragraph.
	Increase Indent	Increases the indent of the selected paragraph.
	Insert/Edit link	Inserts a hyperlink at the cursor location or edits the selected hyperlink. A user can jump to the associated anchor by clicking this hyperlink.
	Remove link	Removes the selected hyperlink.
	Anchor	<p>Inserts a text anchor at the cursor location. A user can jump to this anchor by clicking the associated hyperlink.</p> <p>Note: If you select text and then click the Anchor button, the anchor you create will replace the selected text.</p> <p>Anchors display as broken image icons in the HTML Editor. These icons are not visible in the HTML Viewer.</p> <p>The broken image icon is a place holder that marks the image location. Right-click the broken image icon to delete the anchor or to edit its properties.</p>
	Insert image	Inserts an image at the cursor location. Forms Designer stores the images in the SYSATTACHMENTS table.
	Insert attachment	Inserts a link to an attachment at the cursor location. Forms Designer stores the attachments in the SYSATTACHMENTS table.
	Insert record	Inserts a link to a record at the cursor location.
	Align Left	Left-aligns the selected paragraphs.
	Center	Center the selected paragraphs.
	Align Right	Right-aligns the selected paragraphs.
	Justify	Fully justifies the selected paragraphs.
	Insert Table	<p>Inserts a table at the cursor location.</p> <p>Note: Right-click the table to access the columns, rows, cells and editing commands.</p>
	Insert horizontal rule	Inserts a horizontal bar at the cursor location.

Icon	Name	Function
	Insert special character	Inserts a specified character at the cursor location.
Font	Font drop-down list	Sets the selected text to the specified font.
Size	Font Size drop-down list	Sets the selected text to the specified font size.

HTML Editor keyboard shortcuts

The HTML Editor has the following keyboard shortcuts.

Shortcut	Action
Ctrl+A	Select all
Ctrl+B	Set the selected text to bold
Ctrl+F	Open a Find dialog box you can use to search the editor's content
Alt+F1	Open Help on Field for the Web client Note: After closing the pop-up help of an HTML Editor field, you need to press Tab in Internet Explorer or Shift+Tab in Firefox to move the focus back to the text input area of the HTML Editor.
Ctrl+H	Open Help on Field in the Windows client
Ctrl+I	Set the selected text to italic
Ctrl+U	Set the selected text to underlined
Ctrl+V	Paste
Ctrl+X	Cut
Ctrl+Y	Redo
Ctrl+Z	Revert

Add a subform to a form

Applies to User Roles:

System Administrator

Subforms enable you to create a form once and then use it in other forms. You construct a subform in the same way as you would a standard form. You display a subform on another form by calling it with a subform widget embedded in the form.

Limitation: If a subform contains only a table, make sure that the table and subform have the same height. This way, a vertical scroll bar will appear for the table if needed when the total height of all table rows exceeds the specified table height. If you specify a table height greater than the subform height, the table rows may overflow outside the subform and overlap other fields on the form. If a subform contains multiple widgets, make sure that the subform height is set to be large enough to hold all the widgets; otherwise overlaps may also occur.

To add a subform to a form:

Follow these steps to add a subform to a form.

1. Click **Tailoring > Forms Designer** in the System Navigator, or type `fd` on the Service Manager command line and then press **Enter**.

2. Type a form name and click **Search**.

Service Manager opens a list of forms. Click the form you want to update to open it.

3. Click **Design** to open design mode.

4. Add a subform widget, or select an existing subform.

The properties for the comfill appear in the properties view.

5. Specify the database field to access in the Value text box of the Input property.

6. Specify the name of the form you want to display when the user hovers over the comfill in the Subform Format field.

7. Specify the virtual join field to associate with the comfill in the Subform Input field.

8. Mark the Subform Enabled checkbox.

9. Click **Web Preview** to see what the form will look like in the Web client.

10. Click **OK** to see what the form will look like in the Windows client.

11. Click **OK** to save your changes.

12. Open the form using Database Manager to test your changes:

a. Click **Tailoring > Database Manager**

b. Click **Search**.

Service Manager opens a list of forms.

c. Click the form you just created to open it.

Sort the data in a subform

Applies to User Roles:

System Administrator

You can configure the default sort order of the data that is displayed in a subform. To do this, configure the following two drop-down lists in the Link form of the subform:

- **Sort Field(To/Target)**
Sets the field by which the subform is sorted
- **Sort Order(To/Target)**
Sets the order in which the field is sorted

Note:

- If multiple links are defined for a field in the Link file of the subform, you must use the first link to define the default sort field and sort order.
- You cannot sort array fields in Service Manager 9.32. Therefore, you should not configure an array field as the default sort field.
- You may experience slow performance if you configure a long text field (for example, the **Description** field) as the default sort field.

Using pop-ups

You can display a form as a pop-up by embedding the comfill widget in the form.

Pop-up forms enable you to create a form once, and then use it in other forms. Construct a pop-up form in the same way as you would a standard form.

Forms displayed as pop-ups are read-only; users cannot interact with them. If you want your users to be able to interact with the form, display the form as a standalone or another form instead of as a pop-up.

- Pop-ups only display one record. If the virtual join returns zero or multiple records, the pop-up form will display an error message.
- Only a single pop-up can be active at a time.

The following widgets are supported for pop-ups. Use a regular subform instead of a pop-up subform for any widget not on this list. If you use a non-supported widget, the pop-up will display an error message.

- Checkbox
- Date
- Group/Frame
- Image
- Label
- Table
- Text
- TextArea
- Wrap Label

Note: For the Web client, the allowed maximum width of a pop-up form is 500 pixels. When designing a pop-up form for the Web client, make sure that the pop-up form is not too wide; otherwise, it may not appear correctly on the Web client.

Add a pop-up to a form

Applies to User Roles:

System Administrator

To add a pop-up to a form:

1. Click **Tailoring > Forms Designer** in the System Navigator, or type `fd` on the Service Manager command line and then press **Enter**.

2. Type a form name and click **Search**.

Service Manager opens a list of forms. Click the form you want to update to open it.

3. Click **Design** to open design mode.
4. Add a comfill widget, or select an existing comfill.

The properties for the comfill appear in the properties view.

5. Specify the name of the form you want to display when the user hovers over the comfill in the Pop-up Subform Format field.
6. Specify source field name to associate with the comfill in the pop-up Subform Input field.
Note: This field should exist on the link record.
7. Click the **Pop-up Subform Enabled** box.

8. Click **Web Preview** to see what the form will look like in the Web client.

9. Click **OK** to see what the form will look like in the Windows client.

10. Click **OK** to save your changes.

11. Open the form using Database Manager to test your changes:

- a. Click **Tailoring > Database Manager**

- b. Click **Search**.

Service Manager opens a list of forms.

- c. Click the form you just modified to open it.

When the user hovers over the comfill control that you placed on the form, if the virtual join returns a single record, Service Manager will display it in a pop-up subform.

Add a dynamic form to a form

Applies to User Roles:

System Administrator

To add a dynamic form to a form:

Refer to `plugins\com.hp.ov.sm.client.common_<VERSION>\src\resources\schema\dynamicForm.xsd` for the format for the XML that describes the dynamic form.

1. Click **Tailoring > Forms Designer** in the System Navigator, or type `fd` on the Service Manager command line and then press **Enter**.

2. Type a form name and click **Search**.

Service Manager opens a list of forms. Click the form you want to update to open it.

3. Click **Design** to open design mode.

4. Drag the Dynamic form icon to the desired location.

5. Click the dynamic form field to select it.

6. Type the name of the form in the **Value** text box of the **Name** Property.

7. Type the name of the field in which to store the XML in the **Value** text box of the **Input** property

8. Specify the height, width, and other properties of the dynamic form, as desired.

Note: Forms designer sets the widget size set to the maximum size the dynamic form (once stretched) can occupy. You can resize the widget, but if you set the size too small, the dynamic form will appear truncated.

9. Click **Web Preview** to see what the form will look like in the Web client.

10. Click **Ok** to see what the form will look like in the Windows client.

11. Click **Ok** to save your changes.

12. To see the dynamic form, open the form on which it is located with Database Manager.

Each time you save changes in Forms Designer, exit the form and reload it in Database Manager.

Best practice

To see the XML and the dynamic form at the same time for testing purposes, create a form with a regular text input field and a dynamic form widget for the same data field. Since the Web client does not allow you to use the same field name more than once on a form, alias the field, making the input property for the dynamic forms widget the alias and the input property for the text file the field itself. Be sure to make one of these read-only, so that they don't interfere with each other. Put your XML into the text field. You can use the magnify option to open a large text box to edit the XML.

Once you have created an acceptable dynamic form, you can generate the XML with a RAD or JavaScript application.

Building a graph diagram

The graph control enables you to display and interact with a graph that consists of a set of nodes connected by edges that represent a network of interrelated objects. The following is a summary of the diagram capabilities:

- Supports hierarchical graphs. Since it is not feasible to draw all possible nodes on one diagram, the graph represents a subset of these as a single node. The user can then drill down to see the node or edge of interest.
- Supports inter-edges. An inter-edge is a virtual edge between two group nodes that represents the many edges that actually run between the two groups.
- Supports advanced layout algorithms that minimize edge crossover.
- Supports a client/server model. The application partitions between the client and the server.
- Supports the ability to bind compound rendering methods to nodes and edges. For example, a node may include an icon, two labels, and a circle around it.
- Supports user definable actions on nodes and edges by double-clicking and/or right mouse click.
- Supports scaling.
- Supports scrolling and scaling.
- Supports printing and output in common file formats.
- Supports modifications to the diagram such as adding, moving, and deleting nodes and edges.

Components of the graph

The graph diagram provides a comprehensive visualization solution that enables the ability to easily visualize relationships between different components. An extensive library of graphics display the relationship in a variety of views. You can customize the views to fit your organization, including using your own graphics for display. You can view, add, or update relationships using the graphical interface. The main components of the graph diagram are:

Graph

A mathematical structure used to model relationships between objects from a certain collection.

Node

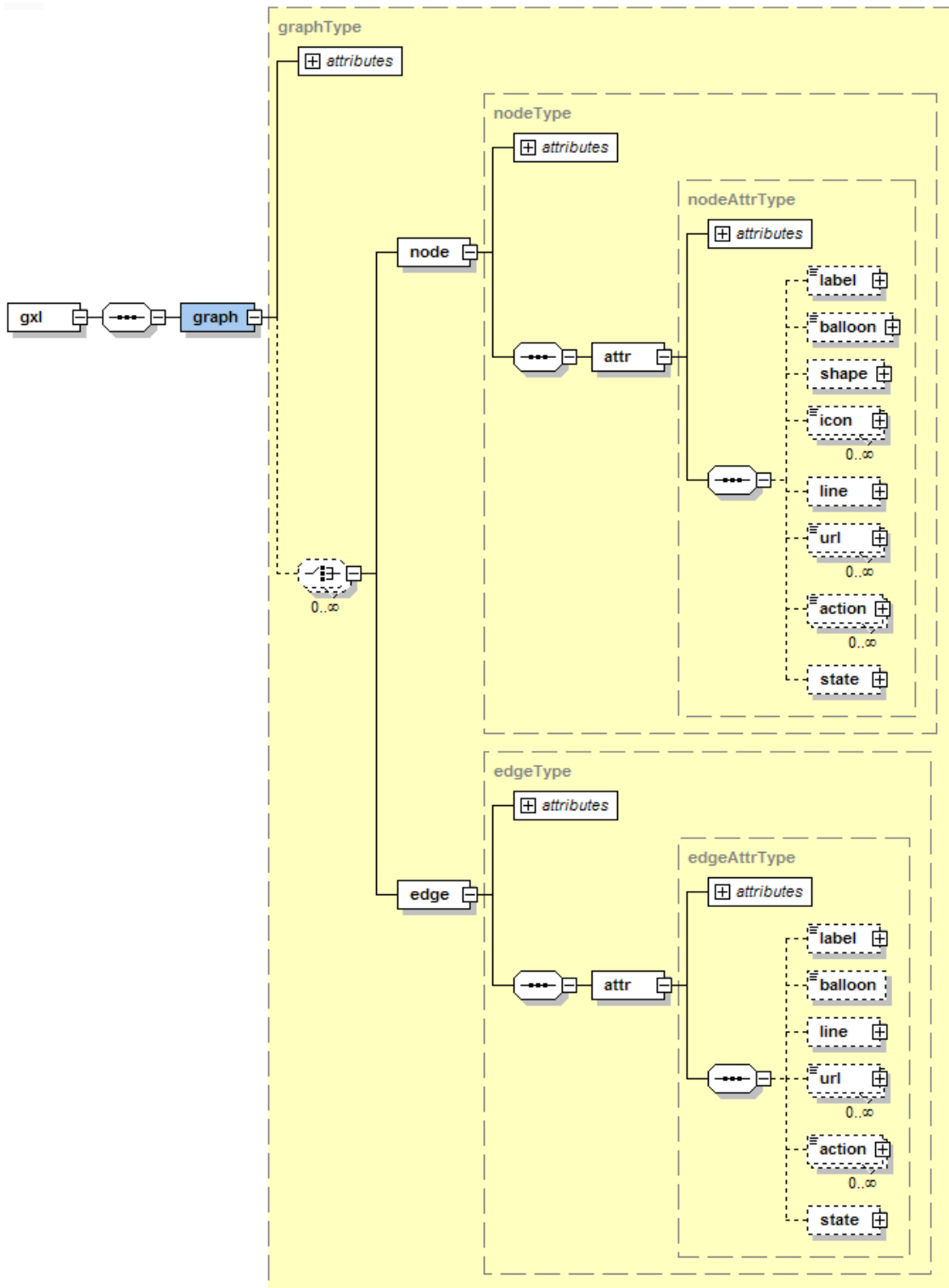
Representation of an entity. Nodes are defined by an ID and at a minimum require that either a bounding shape or an icon be defined so they are visible in the graph. Nodes can be joined by an edge (which also has an ID) by specifying the IDs of the joined nodes as the "to" and "from" attribute values of the node edge. Line styles and colors are optional.

Edge

Represents a relationship between two nodes. An edge is usually depicted as a line connecting its end nodes. An arrowhead at the target end of the line indicates the destination of a directed edge.

Defining the graph in XML format

The XML used to define a graph for Service Manager is an extension of a graph definition standard GXL. The GXL standard defines the basic structure for defining nodes and edges, but is lacking in specifics about graph visualization. The primary extension is the addition of an *attr* element for conveying visualization specific properties of the nodes and edges. The diagram below illustrates the primary elements involved in the definition. For more detail about the proper structure of the XML, see the link to the schema in the related topics.



W3C schema for graph diagram XML

This is the schema for validating a well formed graph definition. The schema is available as `dv/gx1-1.-
-dvd.xsd` in the war file.

```

<?xml version="1.0" encoding="UTF-8"?>
<!--W3C Schema generated by XML Spy v4.4 U (http://www.xmlspy.com)-->
<xs:schema xmlns="http://hp.com/schema/gxl-1.0-dv.xsd" xmlns:xs="
http://www.w3.org/2001/XMLSchema" targetNamespace="http://hp.com/schema/gxl-1.0-dv.xsd"
elementFormDefault="qualified">
  <xs:element name="gxl">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="graph" type="graphType"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="nodeAttrType">
    <xs:sequence>
      <xs:element name="label" type="labelType" minOccurs="0"/>
      <xs:element name="balloon" type="labelType" minOccurs="0"/>
      <xs:element name="shape" type="shapeType" minOccurs="0"/>
      <xs:element name="icon" type="iconType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="line" type="lineType" minOccurs="0"/>
      <xs:element name="url" type="urlType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="action" type="actionType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="state" type="nodeStateType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required" fixed="Viz"/>
  </xs:complexType>
  <xs:complexType name="edgeAttrType">
    <xs:sequence>
      <xs:element name="label" type="labelType" minOccurs="0"/>
      <xs:element name="balloon" type="xs:string" minOccurs="0"/>
      <xs:element name="line" type="edgeLineType" minOccurs="0"/>
      <xs:element name="url" type="urlType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="action" type="actionType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="state" type="stateType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required" fixed="Viz"/>
  </xs:complexType>
  <xs:complexType name="shapeType">
    <xs:attribute name="type" type="shapetypeEnum" use="required"/>
    <xs:attribute name="x" type="xs:short" use="optional"/>
    <xs:attribute name="y" type="xs:short" use="optional"/>
    <xs:attribute name="width" type="xs:short" use="optional"/>
    <xs:attribute name="height" type="xs:short" use="optional"/>
    <xs:attribute name="color" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:complexType name="edgeType">
    <xs:sequence>
      <xs:element name="attr" type="edgeAttrType"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:ID" use="required"/>
    <xs:attribute name="from" type="xs:string" use="required"/>
    <xs:attribute name="to" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="graphType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="node" type="nodeType"/>
      <xs:element name="edge" type="edgeType"/>
    </xs:choice>
    <xs:attribute name="id" type="xs:NMTOKEN" use="required"/>
  </xs:complexType>

```

Valid values for graph elements and attributes

These are the valid values for the elements and attributes.

Element/attribute name	Type
action	String specifying an action acted on by the associated action handler.
action/balloon	String specifying tooltip to display if action is shown in a menu.
action/forcemenu	If set to true, indicates the pop-up should be display when only one labeled action is defined.
action/label	String specifying a label to display if action is shown in a menu.
action/type	String specifying the type of action invoked by the action handler.
balloon	String specifying the balloon text to show for the item.
edge/from	String specifying the ID of the originating node.
edge/id	String specifying a unique ID for the edge.
edge/to	String specifying the ID of the target node.
icon	String specifying relative or absolute URL or relative resource path.
icon/group	String specifying a display group name.
icon/position	String specifying icon position from one of the possible values relative to the node center. The default is C. N = North or above NE = Northeast or upper right E = East or right SE = Southeast or lower right S = South or below SW = Southwest or lower left W = West or left NW = Northwest or upper left C = Center
label/font	String specifying font name per Java Font class.
label/style	Integer specifying font style per Java Font class.
label/size	Integer specifying font size per Java Font class.
line/color	String specifying color name or Hex RGB color value prefixed with #. The default is black.

Element/attribute name	Type
line/end	Integer specifying the type of line end. The default is 0. NO_ARROW = 0 SOURCE_ARROW = 1 TARGET_ARROW = 2 SOURCE_AND_TARGET_ARROW = 3
line/highlightcolor	String specifying highlight color name or Hex RGB color value prefixed with #. The default is none.
line/style	Integer specifying line style. The default is 0. SOLID = 0 LONG_DASH = 1 SHORT_DASH = 2 DOT = 3 DASH_DOT = 4 DASH_DOT_DOT = 5
line/weight	Integer specifying the line weight. The default is 2.
node/id	String specifying a unique id for the node.
shape/color	String specifying fill-color name or Hex RGB color value prefixed with #. The default is none.
shape/height	Integer specifying height.
shape/type	Integer specifying the type of shape. The default is -1. NONE=-1 RECTANGLE = 0 ELLIPSE = 1 DIAMOND = 2 ROUNDED_RECT = 3
shape/width	Integer specifying width.
shape/x	Integer specifying x coordinate.
shape/y	Integer specifying y coordinate.
state/expanded	If set to true, indicates the node is currently expanded.
state/hidden	If set to true, indicates the node or edge is currently hidden.
state/opened	If set to true, indicates the node has been opened by reading the associated XMLADD action or URL.
state/root	If set to true, indicates the node is the root node for the graph.
url	String specifying a relative or absolute URL path
url/balloon	String specifying tooltip to display if URL is shown in a menu.

Element/attribute name	Type
url/label	String specifying a label to display if URL is shown in a menu.
url/urltype	String specifying the type of URL. The default is HTML. HTML = new HTML page XML = new XML defining graph XML_ADD = XML content to add to current graph

Referencing images in the XML for the graph control

When you are building an application that includes a graph diagram, you can reference an image that is attached to a record in Service Manager. The graph uses the attached image to display the icon for that node. You can also display portrait icons that are attached to contact records in the graph diagram.

This is an example of a schema definition for an icon.

```
<xs:complexType name="iconType" mixed="true">  
  <xs:sequence>  
    <xs:element name="path" type="xs:string" minOccurs="0"/>  
    <xs:element name="label" type="labelType" minOccurs="0"/>  
    <xs:element name="balloon" type="labelType" minOccurs="0"/>  
    <xs:element name="url" type="urlType" minOccurs="0" maxOccurs="unbounded"/>  
  </xs:sequence>  
  <xs:attribute name="group" type="xs:string" use="optional"/>  
  <xs:attribute name="position" type="positionType" use="optional"/>  
</xs:complexType>
```

The syntax to reference an image from the database and display it in the graph is:

scattach://{type}:{uid}:{attachmentname}:{table}:{record}, where *type* equals 'img' or 'file'.

The unique identifier for each image is located on the corresponding SYSATTACHMENTS record.

The following is an example of how to reference an image that is an attachment to a record.

```
scattach://file:45c0dfd600245044034b76c8:max_manager_small.gif:contacts:MANAGER,  
MAX
```

The following is an example of how to reference a portrait image on a contacts record.

```
scattach://img:45afb84a002d42c403030d48:MANAGER, MAX:contacts:MANAGER, MAX
```

Here is an example of a complete and valid XML statement that references an image. This particular statement references an image that is attached to Max Manager's contacts record. This displays as an

icon for the node in the graph diagram.

```
<?xml version="1.0" encoding="UTF-8"?>
<gxl>
  <graph id="">
    <node id="MANAGER, MAX">
      <attr name="Viz">
        <label font="Times New Roman" size="12" style="1">MANAGER, MAX</label>
        <balloon>Max Manager</balloon>
        <icon>scattach://file:45c0dfd600245044034b76c8:max_manager_small.gif:contacts:MANAGER, MAX</icon>
        <action label="Expand" type="XMLADD">XMLADD</action>
        <action label="Action 1" type="Action 1">Action 1</action>
        <action label="Action 2" type="myaction2">Action 2</action>
      </attr>
    </node>
  </graph>
</gxl>
```

Icons with larger position values appear on top of smaller position values unless the images specify the same value for the position attribute. In that case, the icons are rendered in document order so that icons later in the document for the same node overlay those preceding them. The origin for all icons is the center. When you specify a group attribute for an icon, it includes items that contain duplicate group attributes.

Defining actions on nodes and edges in the XML for the graph control

When you are building an application that includes a graph diagram, you can define the actions on nodes and edges.

Following is the pertinent part of the schema definition for an action on a node or an edge. The *type* attribute is required and can be any value if a custom action handler is provided. If the default action handler is used, then the possible values are the same as the *urltype* attribute for the *url* element. If a *label* attribute is present, the action appears on the popup action menu when more than one labeled action is available. Providing a value of true with the *forcemenu* attribute causes the popup to appear even when there is only a single possible action.












```
<xs:complexType name="actionType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="type" type="xs:string" use="required"/>
      <xs:attribute name="label" type="xs:string"/>
      <xs:attribute name="balloon" type="xs:string"/>
      <xs:attribute name="forcemenu" type="xs:boolean"/>
      <xs:attribute name="target" type="xs:string"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

The following is an example of a complete and valid XML statement that defines an expand action on the first node and two additional actions on each node and edge. It also references the images that are attached to the contacts records of Max Manager and Steve Supervisor. These display as icons for each node in the graph diagram.

```
<?xml version="1.0" encoding="UTF-8"?>
<gxl>
  <graph id="">
    <node id="MANAGER, MAX">
      <attr name="Viz">
        <label font="Times New Roman" size="12" style="1">MANAGER, MAX</label>
        <balloon>Max Manager</balloon>
        <icon>scattach://file:45c0dfd600245044034b76c8:max_manager_small.gif.contacts:MANAGER, MAX</icon>
        <action label="Expand" type="XMLADD">XMLADD</action>
        <action label="Action 1" type="Action 1">Action 1</action>
        <action label="Action 2" type="myaction2">Action 2</action>
      </attr>
    </node>
    <node id="SUPERVISOR, STEVE">
      <attr name="Viz">
        <label font="Times New Roman" size="12" style="1">SUPERVISOR, STEVE</label>
        <balloon>Steve Supervisor</balloon>
        <icon>scattach://file:45c10d0b00116014030be8e0:steve_supervisor.gif.contacts:SUPERVISOR, STEVE</icon>
        <action label="Action 1" type="Action 1">Action 1</action>
        <action label="Action 2" type="myaction2">Action 2</action>
      </attr>
    </node>
    <edge from="SUPERVISOR, STEVE" id="SUPERVISOR, STEVE - MANAGER, MAX" to="MANAGER, MAX">
      <attr name="Viz">
        <line color="#000000" style="0" weight="2"/>
        <label font="Times New Roman" size="12" style="1">Manages</label>
        <action label="Edge Action 1" type="Edge Action 1">Edge Action 1</action>
        <action label="Edge Action 2" type="myaction2">Edge Action 2</action>
      </attr>
    </edge>
  </graph>
</gxl>
```


Graph toolbar icons

The following table describes the icons available for use on the graph diagram:

Name	Icon	Description
Zoom In		Decreases the size of the current visualization diagram.
Zoom Out		Enlarges the size of the current visualization diagram.
Fit Content		Fits the content of the diagram to the view.
Select Mode		Highlights an item in the visualization diagram.
Window Area Mode		Clicking and dragging the cursor to highlight one or more items will zoom in and fit the selection to the current view.
View Drag Mode		Clicking the mouse and dragging the cursor will pan the diagram within the current viewable area.
Drill Down Mode		Drills down further into a node and its relationships.
Show Overview		Turns on the overview window displaying an overview of the entire graph contents highlighting what is displayed in the working window.
Show Node Labels		Shows or hides the labels on nodes.
Show Edge Labels		Shows or hides the labels on edges.
Print Preview		Displays a print preview of the visualization diagram in the current focus. You can then select Print from the Print Preview dialog.
Select Layout		Filter the view of the graph using one of the available layouts.

Functions and displayevent definitions for actions in the graph diagram

These functions enable users to interact with a graph that represents a network of interrelated objects. Both RAD and JavaScript functions are supported.

Graph event

The following are the functions and displayevent definition that are required to get and display more information in a graph diagram. When a user requests more information in the graph diagram, the applications generate additional XML that is routed through the server to the client for re-rendering.

Displayevent

OnGraphEvent (32105)

RAD functions

get.graph.id()

get.graphnode.id()

add.graphnodes(xml)

JavaScript functions

get_graph_id()

get_graphnode_id()

add_graphnodes(String strXML)

Graph action

The following are the functions and displayevent definition that are required to activate an action that is defined for a node. When a user selects an action from the pop-up menu for a node, the application can access the Graph ID, Action, and Target by using the functions below. This information can be used in the displayevent to call additional application logic.

Displayevent

OnGraphAction (32100)

RAD functions

get.graph.action()

get.graph.target()

JavaScript functions

get_graph_action()

get_graph_target()

Activate an action on a node or edge in a graph

Applies to User Roles:

System Administrator

To activate an action on a node or edge in a graph:

1. Click once on a node or edge inside of a graph diagram.
A pop-up menu opens with a list of available actions.
2. Click on a node or edge action from the pop-up menu.
The action and target execute.

Note: Right- or left-click any of the toolbar controls to activate an action.

Set the focus on the graph

Applies to User Roles:

System Administrator

To set the focus on the graph:

1. Type `fd` on the Service Manager command line and press Enter to start Forms Designer.
2. Type a form name and click **Search** or create a new form.
3. Once in design mode, click the **Graph** control in the form.
This highlights the Graph control and enables you to edit its properties.
4. In the Properties view, set the Input property value to a database field or variable. For example, `$L.graph`.
This sets the focus on the graph and enables the end user to perform an action on the graph (such as view a node detail) and then return to the tab with the graph.

Forms Designer best practices

Forms Designer can produce portable forms that render successfully in the Windows and Web clients if you follow a few design suggestions. To ensure all forms are portable, test each new or upgraded form with both clients.

Hint: You can cut or copy and paste existing objects onto other forms.

Printing forms: printing is designed to work nicely with label/inputs; however, it will NOT work with arbitrary positioning of widgets in the form. If you want to print a table, make it a table - do not simulate one with "cleverly" placed labels.

Using pop-up subforms can save screen real estate, and can be helpful for performance because the query to obtain the subform information is not made until the data is needed for display.

Form design

The right edge of any form should be at grid unit 156. The bottom of any form should be at grid unit 42. Forms which follow this guideline will fit in the default fonts at 640x480 and 800x600. Additionally, these forms provide enough room along the edges for a scroll bar if required.

When placing controls on forms, make all character dimensions (coordinates for placement on the screen, height, width) even numbers, because the space required to display one character is 2x2. When the `ArrayLength` of a field is set to be a number greater than 0 and the field is situated at an odd coordinate, it will automatically shift to an even coordinate.

Important: Each field needs a unique name property. You can change the value in the name property, but do not delete it.

Things to keep in mind when designing forms.

- Provide logical navigation buttons in all forms. For example, the Back, Cancel, Submit buttons.
- Use a Submit or Search button where appropriate instead of relying solely on the ENTER key.
- Stack wide elements like text areas used for descriptions. Multiple wide fields can make a form too wide. Stacking is easier to read than wide elements placed side-by-side.
- Avoid creating wide forms. Recent surveys show that the majority of users do not use monitor settings greater than 1024x768.
- Avoid three-column layouts. Scrolling vertically is more acceptable to most people than scrolling horizontally.
- Avoid adding too many fields, widgets, and notebook tabs to a single form. Simplify forms by breaking them up into smaller forms that link to other forms through a logical flow. Too much information is overwhelming to most users and results in a less productive application. Too many controls on a form can severely degrade performance on the Web client.
- Avoid placing elements too close to each other. Forms are easier to read and use when the text and fields are spaced out. When users increase font sizes this becomes more apparent. This can translate into overlapping form elements, text, and containers in the Web and Eclipse clients.
- Printing works well with label/inputs. It does NOT work with arbitrary positioning of widgets in the form. If you want to print a table, make it a table. Do not simulate one with cleverly placed labels.
- When you design a QBE list form that contains a table, do not enclose the table in a Frame or Group; otherwise the QBE list form will not function properly.

Form layouts

Follow these guidelines to generate forms that display well on a variety of platforms and with different screen resolutions and fonts.

The size of Service Manager forms and the objects on them are defined in terms of grid units. For instance, a combo box may be defined to be 36 units wide and 2 units tall. Objects are also placed on forms using the same units. For example, the same combo box may have its upper left corner at a location 5 units to the right of the form edge and 4 units beneath the top of the form.

The size of a grid unit varies depending upon the currently selected Service Manager font. The grid unit is always defined as being half as wide as the lower case *e* in the current font, and half as tall as the lower case *e* in the current font. Thus, in a font whose letter *e* is 8 pixels wide by 12 pixels tall, the grid unit is 4 pixels wide and 6 pixels tall.

Changing the system's font changes the size of a form on the screen. Just as importantly, it may change the relative shape of the objects therein. If a screen's grid unit goes from 4x4 to 6x8, then objects on the screen becomes 50% taller, and doubles in width. Thus, the screen appears to stretch and have different proportions than it did originally.

An important point to recognize is that Windows true-type fonts are non-deterministic. Each video driver manufacturer can bundle its own hardware mapping of common fonts, and most modern video cards do so. Many manufacturers improve upon the base Microsoft Windows definition of what constitutes a particular font. Thus, the letter *e* in Arial 8 pt. bold when displayed at 640x480 on one video card may have a different metric than the same letter displayed at the same resolution on a different video card.

Use easy to understand names and descriptions for buttons and links. Acronyms and cryptic language makes it is difficult or impossible to know what is going to happen if you click the link or press the button.

Fonts

Using font styles such as Bold or Italic affects the spacing of the characters on a form. If labels are truncated in forms, select a different font, style, or size. You can also increase the width of the form to accommodate a different font, style, or size by specifying an increased width percentage in Service Manager preferences.

Web client forms

The Service Manager Web tier offers automatic support for existing applications and their forms. The Web tier generates dynamic HTML that approximates the exact layout of forms as you define them with Forms Designer. The default application forms are portable from the Windows client to the Web client with no required modification.

If you upgrade an existing system, you will find that Service Manager clients automatically support and display your customized forms. However, there may be cases when further form modification is necessary to correct cosmetic issues that appear when you view the form with the Web client. The following table describes common form revisions that might be required.

Area	Correction
Overlapped objects	Ensure that form objects do not overlap each other. The Windows client makes slight adjustments to correct for overlapped objects, but these design issues are exposed on a web browser.
Dynamic resizing	The Windows client dynamically resizes objects such as text areas and notebooks when you resize the window. The Web client does not resize most objects and does not support elastic properties as you resize the browser window. Therefore, ensure that you assign a default initial size.
Graphics and images	Size any graphics and images to fit conservatively in the form. The Windows client supports scaled images used as buttons. However, the Web client displays these images in their native size. The button grows to accommodate the size of the image.
Dynamic View Dependency (DVD) conditions	Hidden data with DVD conditions may be exposed if the form permits the user to make changes.

Collapsible sections

In the Web client, groups of items can be displayed in collapsible sections. To display such items, use the Group control in Forms Designer, and check the Collapse Enabled property.

When designing forms with collapsible sections, use the following guidelines to avoid formatting problems when the form is displayed in the Web client.

Important guidelines

- All groups on the form should be marked as Collapsible-enabled.
- All widgets must be placed in a collapsible group box; no widgets should remain floating independently on the form.
- Do not stack or mix collapsible group boxes with non-collapsible group boxes.

Sizing graphics

When creating or placing images in the Service Manager GUI, create the image in the size needed. Do not create oversized or undersized images and expect Service Manager to resize them properly. There are several important reasons for this.

- When scaling images, the quality of the image distorts.
- The size of the image is not changed and thus may be a performance consideration.
- The Web client has no way of resizing the image. For example, in the Service Manager GUI, if you placed a large image in a button, the GUI and the Eclipse client size it properly based on the button container. The Web client does not do this and has no way of scaling the image to the size of the parent container. Thus the image appears as its original size in the Web client. The bigger the original size, the bigger it appears in the Web client. On the Web this can also be a major performance issue because large images require more bandwidth and cause slow browser performance.

Form naming conventions

Form names incorporate the application or feature identification as a prefix in the form name. Look for these prefixes to identify updated or new forms.

Application	Form prefix	Related table
Problem Management	PM	knownerror rootcause rootcausetasks
Service Level Management	sla	sla
Service Desk	SM	incidents
Incident Management	IM	probsummary

Building accessible forms

You can use the Forms Designer to make your system more accessible. In addition to the accessible attributes, you can customize forms for easier readability and information manipulation.

Attribute name	Definition
accessiblename option	Give a form element a name compatible with a screen reader.
accessibledescription option	Give a form element a description compatible with a screen reader.

Accessible Web client forms

The accessible mode of the Web client allows users that require different levels of accessibility to apply personal preferences to improve their user experience. The accessible Web client also enables accessibility tools, such as screen readers, to work with Service Manager. The accessible Web client omits the graphical workflow feature, and thread navigation links, which are the tabs that identify open forms in the Windows client.

If you are designing forms for accessible use, these are the most important design requirements:

- High-contrast color graphics
- Larger default fonts
- Larger form spacing (more white space)
- Simplified navigation (fewer buttons, objects and icons)
- Browser settings must be able to control
 - Resizing fonts
 - Foreground and background color selection

A visually impaired user might want a well-designed form that reads left to right with labels announcing the name of the subsequent form object and tables with row labels that read horizontally (not by column). This user might also want to specify a black background with white text in a 14-point bold font, instead of the default color and font combinations.

Putting the HTML Editor on accessible forms

The HTML Editor must be the last widget in its form for reverse tabbing to work properly. Widgets placed under it cannot be tabbed.

The Accessible Web client ignores the tab stop information set in the Form definition. Instead, the tabbing always moves from left to right and top to bottom so that the tabbing order matches the order in which the widgets are described by a screen reader.

For complex editing requiring 508 compliance, edit the text in a 508 compliant HTML Editor and then paste it into the HTML Editor.

Dynamic View Dependencies

Steps taken to ensure better data entry and form usability are critical to a database. This includes visual enhancements and form modifications based on user input. Dynamic View Dependencies (DVD) is a feature of Forms Designer that enables you to display objects and their attributes on a form depending on user interaction.

DVD enables you to create control or layout objects with properties that are evaluated when the form appears to control the appearance of these objects on the form.

You can use dynamic view dependencies to:

- Dynamically determine the properties of a field based on the DVD condition.
- Make a field visible only when another field contains a certain value.
- Dynamically determine the choices in a drop-down list based on the choice in another combo box.
- Dynamically set the font or color of a field based on the value of another field.
- Dynamically set the font or color of a field based on its value.

Note: DVD select statements can only be used on text fields.

You create dynamic view dependencies by specifying conditional statements for properties of a control. The three types of conditional statements for a property are described in the following table.

Conditional Statements	Syntax	Example
Simple expression	FIELD [OPERATOR VALUE]	<ul style="list-style-type: none">• [\$invisible] = 0

Conditional Statements	Syntax	Example
<p>(Field comparisons):</p> <ul style="list-style-type: none"> • Enable you to specify comparisons of equality or inequality. • Evaluate to “true” or “false” at runtime. 	<ul style="list-style-type: none"> • FIELD: Any valid input field or variable enclosed in brackets. • OPERATOR: =, >, <, <>, or # • VALUE: Any quoted string or number <p>Note: You can use or &to concatenate multiple expressions.</p> <p>For more information, see "Field comparisons" on page 251 and "Use field comparisons" on page 251.</p>	<ul style="list-style-type: none"> • [<code>invisible</code>] <> 1 • [<code>L.one.click</code>]<>true &[<code>quantity</code>]>1&[<code>orderonbehalf.marker</code>]<>true
<p>Value:Display expression (Field value matching)</p> <ul style="list-style-type: none"> • Enable you to specify matching output values for various input values. • Return a single value or a list of values when evaluated at runtime. 	<p>FIELD ? ValueList : DisplayList</p> <ul style="list-style-type: none"> • FIELD: Any valid input field or variable enclosed in brackets. • ValueList and DisplayList: Zero or more comma separated values. <p>For more information, see "Field value matching" on page 252 and "Use field value matching" on page 252.</p>	<p>[<code>severity</code>]? "sev1","sev2","sev3" : 1,7,2,13</p> <p>This example returns a value as follows:</p> <ul style="list-style-type: none"> • 1 if <code>severity</code>="sev1" • 7 if <code>severity</code>="sev2" • 2 if <code>severity</code>="sev3" • 13 if <code>severity</code> is none of the listed "sev" values
<p>Function call (Dynamic functions):</p> <ul style="list-style-type: none"> • Enable you to call a function that evaluates a result dynamically. 	<p>FUNCTION(ParaList)</p> <ul style="list-style-type: none"> • FUNCTION: A valid function name (for example, <code>select</code>) • ParaList: Zero or more comma separated VALUES <p>For more information, see "Dynamic functions" on page 253 and "Use dynamic functions" on page 254.</p>	<p><code>select("subcategory", "subcategory", "category", [<code>category</code>])</code></p> <p>This example returns a list of subcategory names from the subcategory table, whose category equals the <code>category</code> value.</p>

Service Manager includes a form, **dep.g**, which demonstrates Dynamic View Dependencies.

Use form dep.g to demonstrate Dynamic View Dependencies

The form dep.g has been prepared to demonstrate Dynamic View Dependencies. You can interact with this form to see the DVD examples this form uses and the dynamic effects they create.

To understand DVD examples on the dep.g form:

1. Click **Tailoring > Forms Designer**.
2. Type dep.g in the Form field, and click **Search**.

The dep.g form opens.

3. Select the **Instructions** tab for instructions on how to use three types of DVD conditional statements. For more information, see ["Dynamic View Dependencies" on page 241](#).
4. Follow the instructions and interact with the different fields to observe how the View properties change dynamically.
5. Click **Design**.
6. Inspect Properties View to see what values are assigned to create the effects. The following tables explain those properties that create the dynamic effects on different tabs of the form.

Labels and Fields tab:

Control Name	Property: Value	Explanation
Providers > Caption text box (Text22)	<ul style="list-style-type: none">Visible: 1Input: \$caption	<ul style="list-style-type: none">This text box is always visible.The value you type in this text box (disabled, big, bold, or italic) is assigned to the \$caption variable, which is referenced in other conditional statements on the same screen.
Providers > Severity text box (Text25)	<ul style="list-style-type: none">Visible: 1Input: \$severity	<ul style="list-style-type: none">This text box is always visible.The value you type in this text box (sev1, sev2, sev3, or any other value) is assigned to the \$severity variable, which

Control Name	Property: Value	Explanation
		is referenced in other conditional statements on the same screen.
Providers > Disabled check box (CheckBox28)	<ul style="list-style-type: none"> o Visible: 1 o Input: \$disabled 	<ul style="list-style-type: none"> o This check box is always visible. o When you select or clear this check box, 1 or 0 is assigned to the \$disabled variable, which is referenced in other conditional statements on the same screen.
Providers > Invisible check box (CheckBox30)	<ul style="list-style-type: none"> o Visible: 1 o Input: \$invisible 	<ul style="list-style-type: none"> o This check box is always visible. o When you select or clear this check box, 1 or 0 is assigned to the \$invisible variable, which is referenced in other conditional statements on the same screen.
Dependents > Caption label (Label33)	<ul style="list-style-type: none"> o Visible Condition: [\$invisible] = 0 o Caption Condition: [\$caption] o Foreground Color Condition: [\$severity] 	<ul style="list-style-type: none"> o This label is visible only when the Providers > Disabled check box (CheckBox28) is not selected. o The value you type in the Providers > Caption text box (Text22) is automatically populated as the caption of this label. o The font color of the label caption changes with a color ID number (1, 2, 3 and so on) you type in the Providers > Severity text box.
Dependents > Expression label (Label35)	<ul style="list-style-type: none"> o Visible Condition: [\$invisible] = 0 o Foreground Color Condition: [\$severity] ? "sev1","sev2","sev3" : 1, 7, 2, 13 o Bold Condition: [\$caption]="bold" o Italic Condition: 	<ul style="list-style-type: none"> o This label is visible only when the Providers > Disabled check box (CheckBox28) is not selected. o The font color of the label caption changes with the value you type in the Providers > Severity text box (Text25): color 1 for sev1, color 7 for sev2, color 2 for sev3, and color 13 for any other entries. o The label caption is bolded if you type bold in the Providers > Caption text box

Control Name	Property: Value	Explanation
	<ul style="list-style-type: none"> [\$caption]="italic" ○ Font Increase Condition: [\$caption] ? "big" : 5, 0 	<p>(Text22).</p> <ul style="list-style-type: none"> ○ The label caption is in italic if you type italic in the Providers > Caption text box (Text22). ○ The font size of the label caption is increased by 5 only when you type big in the Providers > Caption text box (Text22).
<p>Dependents > Disable Checkbox text box (Text37)</p>	<ul style="list-style-type: none"> ○ Visible Condition: [\$invisible] = 0 ○ Read-Only Condition: [\$disabled] ○ Input: \$x 	<ul style="list-style-type: none"> ○ This text box is visible only when the Providers > Disabled check box (CheckBox28) is not selected. ○ This text box is read-only when the Providers > Disabled check box (CheckBox28) is selected. ○ The value you type in this text box is assigned to the \$x variable, which is also the Input of the Disable Caption text box (see below).
<p>Dependents > Disable Caption text box (Text39)</p>	<ul style="list-style-type: none"> ○ Visible Condition: [\$invisible] <> 1 ○ Input: \$x ○ Read-Only Condition: [\$caption] = "disabled" 	<ul style="list-style-type: none"> ○ This text box is visible only when the Providers > Disabled check box (CheckBox28) is not selected. ○ The input of this text box is \$x, which is same as the value in the Dependents > Disable Checkbox text box (Text37). ○ This text box becomes read-only when you type disabled in the Providers > Caption text box (Text22).
<p>Dependents > Visible label (Label41)</p>	<ul style="list-style-type: none"> ○ Visible: 1 ○ Caption Condition: [\$invisible] ? 0, 1 : "Yes", "No" 	<ul style="list-style-type: none"> ○ This label is always visible. ○ The caption of this label is Yes when the Providers > Disabled check box (CheckBox28) is not selected, and No when the check box is selected.

Select Function tab:

Control Name	Property: Value	Explanation
Categories combo box (ComboBox51)	<ul style="list-style-type: none"> ○ Input: \$category ○ Value List Condition: select("name", "category", "category", *) 	<ul style="list-style-type: none"> ○ The value you select in this combo box is assigned to the \$category variable, which is referenced in the Value List Condition property of the Subcategories combo box. ○ This select function returns all category names from the category table (that is, the category field can be any value (*)).
Subcategories combo box (ComboBox53)	<ul style="list-style-type: none"> ○ Input: \$x ○ Value List Condition: select("subcategory", "subcategory", "category", [\$category]) 	<ul style="list-style-type: none"> ○ The value you select in this combo box is assigned to the \$x variable (which is not referenced in any other conditional statements on the same screen though). ○ This select function returns, from the subcategory table, the subcategories of a category you specify in the Category combo box (\$category).
Operator combo box (ComboBox57)	<ul style="list-style-type: none"> ○ Input: \$operator ○ Value List Condition: select("name", "operator", "name", *) 	<ul style="list-style-type: none"> ○ The value you select in this combo box is assigned to the \$operator variable. ○ This select function returns the login names of all records in the operator table (that is, the login name can be any value (*)).
Capabilities text area (MultiText59)	<ul style="list-style-type: none"> ○ Caption Condition: select("cap.exec", "operator", "name", [\$operator]) 	This select function returns the capability words (the cap.exec field in the operator table) of the operator that you select in the Operator combo box into the Capabilities text area.

Buttons tab:

Control Name	Property: Value	Explanation
Ok radio button (Radio70)	<ul style="list-style-type: none"> ○ Input: \$buttonID ○ Value: Ok 	<ul style="list-style-type: none"> ○ The input value of this radio button is assigned to the \$buttonID variable. ○ When selected, this radio button has an input value of Ok.
Cancel radio button	<ul style="list-style-type: none"> ○ Input: \$buttonID ○ Value: Cancel 	<ul style="list-style-type: none"> ○ The input value of this radio button is assigned to the \$buttonID variable. ○ When selected, this radio button has an

Control Name	Property: Value	Explanation
(Radio72)		input value of Cancel .
Design radio button (Radio73)	<ul style="list-style-type: none"> Input: \$buttonID Value: Design 	<ul style="list-style-type: none"> The input value of this radio button is assigned to the \$buttonID variable. When selected, this radio button has an input value of Design.
None radio button (Radio74)	<ul style="list-style-type: none"> Input: \$buttonID Value: None 	<ul style="list-style-type: none"> The input value of this radio button is assigned to the \$buttonID variable. When selected, this radio button has an input value of None.
Button (Button71)	<ul style="list-style-type: none"> Caption Condition: [\$buttonID] Button ID Condition: [\$buttonID] ? "Ok", "Cancel", "Delete", "Design" : 3, 5, 4, 9, 50 	<ul style="list-style-type: none"> The caption of this button dynamically changes to the value of the \$buttonID variable (Ok, Cancel, Design, or None). The ID of this button changes dynamically with the \$buttonID variable value as follows: <ul style="list-style-type: none"> 3 when the \$buttonID value is Ok; 5 when the \$buttonID value is Cancel; 4 when the \$buttonID value is Delete; 9 when the \$buttonID value is Design; 50 when the \$buttonID value is none of the above-listed values <p>When you press the button, it functions as the Ok, Cancel, or Design button accordingly.</p>

Bitmaps tab:

Control Name	Property: Value	Explanation
Provider > Ticket Severity combo box (ComboBox79)	<ul style="list-style-type: none"> Input: \$ticketSeverity Value List: ok;warning;alert Display List: 	<ul style="list-style-type: none"> The value you select in this combo box is assigned to the \$ticketSeverity variable. The available selections are: ok, warning, and alert. The available selections are displayed as: Low,

Control Name	Property: Value	Explanation
	Low;Medium;High	Medium, and High.
Dependents > chart (Chart81)	<ul style="list-style-type: none"> ○ Caption Condition: [\$ticketSeverity] ? "ok", "warning", "alert" : "1","5","10","0" ○ Color List=green;yellow;red ○ Color Scale=1;5;10 	<p>Note: The Properties pane of this chart does not currently display the complete property information. You can open the dep.g form in non-Design mode and click Format Detail from the More Actions menu to view the complete property information of this chart.</p> <ul style="list-style-type: none"> ○ A value displays in the column of the chart as follows: <ul style="list-style-type: none"> • 1 when \$ticketSeverity is "ok" (that is, when you select Low in the Ticket Severity combo box); • 5 when \$ticketSeverity is "warning" (that is, when you select Medium in the Ticket Severity combo box); • 10 when \$ticketSeverity is "alert" (that is, when you select High in the Ticket Severity combo box); • 0 when \$ticketSeverity is none of the above-listed values (that is, when the Ticket Severity combo box is empty). ○ A list of colors is applied to the chart column: green, yellow, and red. ○ The color of the chart column is determined as follows: <ul style="list-style-type: none"> • Green when the column value is 1; • Yellow when the column value is 5; • Red when the column value is 10.
Dependents > image (Graph82)	<ul style="list-style-type: none"> ○ Image Condition: [\$ticketSeverity] 	This image dynamically changes with the option you select in the Ticket Severity combo box: Low (ok.gif), Medium (warning.gif), or High (alert.gif).
Dependents > marquee	<ul style="list-style-type: none"> ○ Caption Condition: [\$ticketSeverity] ? 	The caption of this marquee dynamically changes with the option you select in the Ticket Severity

Control Name	Property: Value	Explanation
(Marquee83)	"ok","warning","alert" : "Severity is low ...","Severity is medium ...","Severity is HIGH!!"	combo box: <ul style="list-style-type: none"> Severity is low ... when you select Low (ok); Severity is medium ...when you select Medium (warning); Severity is HIGH!! when you select High (alert).

Circular Dependencies tab

Control Name	Property: Value	Explanation
Hello check box (CheckBox88)	<ul style="list-style-type: none"> Input: \$hello Selected Condition: [\$world] 	<ul style="list-style-type: none"> The input value of this check box (0 or 1) is assigned to the \$hello variable. This check box is selected or deselected when the value of the \$world variable is 1 or 0 (that is, when the World check box is selected or deselected).
World check box (CheckBox89)	<ul style="list-style-type: none"> Input: \$world Selected Condition: [\$hello] 	<ul style="list-style-type: none"> The input value of this check box (0 or 1) is assigned to the \$world variable. This check box is selected or deselected when the value of the \$hello variable is 1 or 0 (that is, when the Hello check box is selected or deselected).
I am my number one fan! check box (CheckBox91)	<ul style="list-style-type: none"> Input: \$self Selected Condition: [\$self] 	<ul style="list-style-type: none"> The input value of this check box (0 or 1) is assigned to the \$self variable. This check box is selected or deselected when the value of the \$self variable is 1 or 0 (that is, select or deselect of this check box is independent of any other controls).

Note: The fields in dep.g use variables because dep.g is not associated to a dbdict. You can use field names on forms that do have an associated dbdict.

Setting Dynamic View Dependencies properties

You can set up a number of properties for dynamic evaluation using the Properties window of Forms Designer. For example, the Caption property that you set for a Label specifies the text to display on the

screen. Under the “Caption” property there is a property named “Caption Condition.” This associated property enables you to specify a condition evaluated dynamically to compute a caption for the label. These two properties are paired for the purpose of DVD conditions. There are several other property pairings as well.

Several Forms Designer controls have “property” and “property condition” pairs. In each case, you can specify a special conditional statement that specifies the run-time value for the property.

DVD conditional property pairings and examples of their supported types of controls are listed in the following table.

Property	Property Condition	Examples of supported control types
Caption	Caption Condition	Button, Check Box, Combo Box, Label, Marquee, Radio Button, Text, and Text Area
Value List	Value List Condition	Combo Box
Display List	Display List Condition	Combo Box
Visible	Visible Condition	Button, Check Box, Combo Box, Label, Marquee, Radio Button, Text, and Text Area
Foreground Color	Foreground Color Condition	Label
Background Color	Background Color Condition	Label
Bold	Bold Condition	Label
Italic	Italic Condition	Label
Font Increase	Font Increase Condition	Label
Read-Only	Read-Only Condition	Check Box, Combo Box, Radio Button, Text, and Text Area
Button ID	Button ID Condition	Button
Image File	Image Condition	Button
-	Selected Condition	Check Box
Mandatory	Mandatory Condition	Combo Box, Radio Button, Text, and Text Area
Enabled	Enabled Condition	Button

Using Dynamic View Dependencies vs. Data Policy

Overuse of DVD on forms can result in performance problems. For example:

- Using the select function or a \$ variable that is not on the form requires an interaction with the server. On connections with slow network response the use of these types of DVD statements can significantly impact performance.
- Using Focus In/Out or Data Changed Events also causes a server interaction. During design, the response time may seem adequate. However on a network with a half second response, it can be cumbersome.

Data Policy operates at the table level and achieves many of the same results without the complexity and without taxing system resources. For example, one common use of DVD is to set Visible or Readonly conditions, both of which can be set using Data Policy. With Data Policy, the conditions are evaluated on the server before the form is sent, thus avoiding a network interaction.

Field comparisons

Field comparisons use a statement to create a conditional property for a control on a form.

Syntax: `FIELD OPERATOR VALUE`

- *FIELD* is any variable or field name surrounded by brackets, for example, [`$x`], [`name`], and so forth.
- *OPERATOR* can be =, >, <, <>, or #.
- *VALUE* is any quoted string or number.

Note: Field comparisons can also use complex conditions or statements to create a conditional property for a control on a form. You can use | or & to concatenate multiple expressions. For example:

```
[$L.one.click]<>true & [$quantity]>1& [$orderonbehalf.marker]<>true.
```

Important: In order for a property to depend on a variable (or record field), the variable must be assigned as an input to an object on the same screen. For examples, see ["Use form dep.g to demonstrate Dynamic View Dependencies" on page 243](#).

Use field comparisons

Here are some examples of using field comparisons.

- Making a certain field visible only when the category of a record equals "network."
 - Assume that the category can be selected via a drop-down list whose Input is defined as `$category`.
You can specify the visibility dependency by assigning the following to the Visible Condition property of the field:
`[$category] = "network"`
 - Instead of a variable, a drop-down list can use a record field as input. If the field is named "category," set up the "Visible Condition" as follows:
`[category] = "network"`

- Adding a DVD visible condition to a field:
`[assignment]#"help"`

- Specifying an inequality comparison:
`[$category] <> "network"` (that is, category is not network)

- Using complex conditions or statements to create a conditional property for a control on a form.
 - `[$category]="network"&[$subcategory]="modem"`
 - `[$category]="network" | [$category]="DEFAULT"`
Complex conditions are true only if both parts of the condition are true.

Field value matching

Field value matching enables you to specify matching output values to various input values.

Syntax: `FIELD ? ValueList : DisplayList`

FIELD is any variable or file field surrounded by brackets, for example, `[$x]`, `[name]`, and so forth.

ValueList and *DisplayList* are lists of comma-separated values

Use field value matching

Setting the color of a label to change based on the value of a severity field

The severity is specified in an edit field whose input is `$severity`. You set the Color Condition of the label as follows:

```
[$severity]? "sev1","sev2","sev3" : 1,7,2
```

This statement tells the label to do the following:

- Use a color value of 1 (red) when `$severity="sev1"`
- Use a color value of 7 (yellow) when `$severity="sev2"`
- Use a color value of 2 (green) when `$severity="sev3"`

Specify a value to use if no match is found

```
[$severity]? "sev1","sev2","sev3" : 1,7,2, 0
```

This tells the system to use color 0 (black) if `$severity` is none of the listed "sev" values.

Note: The `$severity` field must have been assigned as input to at least one field on the screen.

Dynamic functions

The dynamic property condition statement enables you to call a function that evaluates a result dynamically.

Syntax: `FunctionName(PARM_0, ..., PARM_n)`

The parameters for the function can be any string or number. One parameter may also be a variable or record field.

The dynamic functions include:

- **lang_preference(lang)**

This function returns the current language table selection. It takes as a parameter the current language value.

You can use it to save new language settings in the user preferences for future reference. It forces the user interface to update its language table and repaint as necessary.

- **select(return_field, file_name, select_field, select_field_value)**

This function works against many records to grab a scalar value from each one to make a list. You can use it to select just one record and use an array from within that record (for example, the `cap.exec` array field of the `operator` table) to make a display list.

This function returns the value of a field (`return_field`) of a record that is from a table (`file_name`) and has a field (`select_field`) value equal to a specified value (`select_field_value`). The

`select_field_value` can be a variable or a wildcard character (an asterisk). The following are examples:

- `select("subcategory", "subcategory", "category", [$category])`
- `select("name", "operator", "name", *)`
- `select("cap.exec", "operator", "name", [$operator])`

Use dynamic functions

Use the following to select all the login IDs from a department to make a list:

```
select("name","operator","department",[$dept])
```

Use the following to select all the approvers for a certain group:

```
select("approvers","cm3groups","name",[$cm3group])
```

Format Control

Format Control allows the System Administrator to apply the following special processes to Service Manager files through individual forms:

- Validate fields.
- Establish user privileges.
- Display alternate record list forms.
- Do Calculations and Validations based on other fields in this file or other files.
- Call RAD routines.
- Define additional options and menu items.
- Automatically update or insert data in other parts of the database.

Format Control records can attach to any form or file within Service Manager and do not require special programming skills to implement. You define routines in Format Control that are user interactive or transparent and perform when a record opens or when a user adds, updates, or deletes a record from the database. Format Control is easy to apply and change.

While Format Control is a convenient utility, it should not be overused. Excessive reliance on Format Control to modify your system results in reduced system performance. If you intend to heavily customize your system, we recommend you find other, more efficient ways to implement your changes such as Data Policy or Data Validation.

Format Control processes

There are eight major functional processes in Format Control that define the actions to be taken on a record. Each of these processes has a separate form within the Format Control Utility.

Main information — This form is the entry point in Format Control and has several functions.

- Initializes fields or variables that are later used in the Format Control record. Initialization expressions are the first operation performed for each evaluation of a Format Control record.
- Initially displays a value in a field when the record itself opens.
- Names special record list and initial query forms to use, sets up default sort sequences, and runs scripts.

Forms — This section in Format Control has two main functions:

- Specify alternate forms to display a list of records.
Note: This option cannot be used when the Record List (split screen) functionality is enabled.
- Specify alternate forms to display a single record.

The formatcontrol option allows you to display the information using different Service Manager forms based on conditions evaluated at run time. You can specify either QBE Forms to display a list of records or Alternate Forms to display a record. You may want to specify different lists according to the user role or capabilities.

By specifying alternate forms, whenever the condition is met, the user will have an option available called "Alternate Forms" where the user can select the form to use to display the information. Additional forms are useful to display different views such as financial or security information when the condition specified is met.

Queries — This process enables you to extract information from a file other than the primary file in order to perform calculations and validations and to report on information from more than one file.

Calculations — This process enables you to perform calculations on currently available fields or variables. The fields needed for calculations may be variables, fields in the primary file, or fields in any other secondary files that may have been queried.

JavaScript — This process enables you to call JavaScripts from Format Control.

Validations — This process enables you to set up a logical expression for checking data in fields or variables on the form. The validation expression you set up must evaluate to true upon the desired edit function for it to be successful. If the expression does not evaluate to true, a validation message opens, and the specified operation fails.

Subroutines — This process enables you to call RAD routines from Format Control.

Additional options — This process enables you to define menu options you want available to users on any form associated with the particular Format Control record. You can use Format Control to set up a menu option called Validity Lookup. This option performs a validity check on the field of focus in the form for which validity specifications have been defined. Invalid fields are highlighted and a message is displayed in the status bar giving details on the type of error incurred.

Note: This feature calls RAD subroutines and is available in Database Manager only.

Privileges — This process enables you to use security to control database options available to the user. If the contents of a field evaluate to true at processing time, the corresponding button is available to the user.

Accessing Format Control

You can access the Format Control utility in Service Manager as follows:

- **Access Format Control from Forms Designer:**

Note: The best method for accessing Format Control is through Forms Designer. Accessing the record from the form with the same name as the Format Control record reduces your chances of accessing the wrong record.

1. Click **Tailoring > Forms Designer**.
2. Type the name of the form you want to view in the Form field of the Forms Designer dialog box.
3. Click **Search** to find the form.
4. Click **Format Control** from the More Actions menu. The Format Control record opens for the selected form.

- **Access Format Control from Database Manager:**

Note: To access the Format Control Additional Options for Change Management, you must select Administration Mode or you will be using Change Management instead of Database Manager.

Click **Tailoring > Database Manager**.

- **To access Format Control directly:**

Note: Accessing Format Control directly without going through Forms Designer enables you to search for an existing record or create a new record.

Click **Tailoring > Format Control**.

Open a Format Control record

Applies to User Roles:

System Administrator

To open a Format Control record:

1. Open Format Control using one of the following options:
 - Click **Tailoring > Format Control**.
 - Type `fc` in the Service Manager command line and click **Execute command** or press **Enter**.

Service Manager opens a blank Format Control Maintenance form.

2. Type the name of the form or leave all fields blank, and then click **Search**.

Service Manager opens a list of active forms. If only one form matches the query, Service Manager will display that form.

3. Click a form on the list to open it. Service Manager opens the selected form, showing the Main Information screen.

Add a Format Control record

Applies to User Roles:

System Administrator

To add a Format Control record:

1. Access Format Control from Forms Designer.

Service Manager creates a new Format Control record for the form. Service Manager automatically places the form's name in the Name field. By default, a value is automatically entered in the System field (a required field) and the Use Default Sort box is selected.

Note: When you access Format Control in this manner, the name of the new record defaults to the same name as the form used to access it. In the rare cases where you need a different name, override the default name.

2. Add the desired data.
3. Click **New** to add the record to the database.

Using expressions in Format Control

To create Format Control expressions you will need to be familiar with system functions, boolean (logical) fields and the file variable.

Use the following definitions when Creating Format Control expressions:

Primary File:

The file attached to a form. This is referred to as \$file in a Format Control record. Additional files are defined as \$file1, \$file2, etc. in the order of their entry in the secondary files section of the Additional File Queries screen of the Format Control record.

Semicolon:

Semicolons(;) separate statements to allow them to be entered on one line.

Variable:

A value that resides in memory only, rather than in a database record. Variable names always start with \$. To ensure that a variable contains the proper value, initialize it in the Initializations Section of the Format Control record.

Statements:

Service Manager supports some C-like statements.

- if then else
- do while
- for do

Format Control system functions

Many of the Service Manager system functions are used to create expressions in Format Control. The following is a list of the most commonly used functions:

Function	Description
denull()	Compresses an array by deleting all trailing NULL entries and returns the compressed array.
index()	Returns the index or position number for a specific element value in an array or character in a string. If the target value is not in the array or string, it returns 0 (zero).
lng()	Returns the number of elements in an array or structure and the number of characters in a string.
nullsub()	Replaces a null value with a known value, eliminating the possibility that expressions will return null or unknown results.
operator ()	Returns the logon id of the current operator.
str()	Converts a non-string data type into a string.
substr()	Extracts a substring from a string.
tod()	Returns the current date and time.
val()	Converts a field into a numeric, logical, or date/time value.

Format Control Boolean (logical) fields

In the forms for Additional File Queries, Calculations, Validations, and Subroutines, you will find some or all of the following fields:

Conditional Field	Format Control processing area	Description
add	Additional File Queries, Calculations, Validations, JavaScript	Conditional expressions used in this field are evaluated before a record is added.
update	Additional File Queries,	Conditional expressions used in this field are evaluated before a record is updated.

Conditional Field	Format Control processing area	Description
	Calculations, Validations, JavaScript	
delete	Additional File Queries, Calculations, Validations, JavaScript	Conditional expressions used in this field are evaluated before a record is deleted.
display	Additional File Queries, Calculations, Validations, JavaScript	Conditional expressions used in this field are evaluated when a screen/format is displayed or when the Format Control record is used with reports.
initial	Additional File Queries, Calculations, Validations, JavaScript	Conditional expressions used in this field are evaluated once, before a record is displayed the first time.
before	Subroutines	This field determines when the Conditional expressions in the add, update, and delete fields are evaluated. If the condition is true, the expression will be evaluated before the record is added, updated, or deleted. If the condition is false, the evaluated after the record is added, updated, or deleted. You cannot use a variable in this field.

These fields must evaluate to true or false only. A value of true (or an expression that evaluates to true) in any of these fields causes the Format Control functions to execute. These functions execute before the edit is performed. If the fields are blank, the calculation or validation are not performed for this table. For example, a value of true in the add field causes Format Control functions to be performed before a record is added. If the other fields (update, delete, display) are blank, the functions are not performed when the record is updated, deleted, or displayed.

Normally, these fields contain true, false, blank, or they can contain a variable that is calculated in the Calculations process before being used in another process. When using either of these methods, use either the short or long version of Format Control to enter an expression that is evaluated to true or false at execution time.

This is an example of an expression that is frequently used:

```
index ("SysAdmin", $lo.ucapex)>0
```

This expression searches the user's capability array in the Operator Record to see if it has SysAdmin capabilities. The function executes if this evaluates to true, indicating that the operator has SysAdmin listed as a capability and therefore is authorized for the function.

Format Control file variable

The current database record (the record the user is opening for modifying) in Format Control is always identified by the variable \$file. Elements of the current record being acted upon by Format Control associate with \$file by expressions. For example, <field.name > in \$file might identify a particular field in the current record to which a Format Control process is attached. The \$file variable does NOT reference a file, which may use several forms to input data.

Important: In order for a property to depend on a variable or record field, the variable must be assigned as an input in Format Control.

Turn on menu forms by using Format Control

Applies to User Roles:

System Administrator

To turn on menu forms using Format Control:

1. Open Format Control using one of the following options:
 - Click **Tailoring > Format Control**.
 - Type `fc` in the Service Manager command line and click **Execute command** or press **Enter**.

Service Manager opens a blank Format Control Maintenance form.

2. Type `login.DEFAULT` and then click **Search**.
3. Service Manager opens the login default form showing the Main Information screen.
4. Add `$G.show.menu.forms="true"` to the end of the **Initialization Expressions**.

Sequential numbering for Format Control

Sequential numbering is a Subroutine process of Format Control that automatically adds identifying numbers to database records as they are created. Sequential numbering is used for Configuration Management, incident records, or employee records.

Sequential numbers are defined for appropriate fields in any data record. You create alphanumeric strings (prefixes and suffixes) that identify records by task or item.

Sequential numbering enables you to:

- Increment/decrement numbers in sequence.
- Reset values at definable intervals.
- Define values for increment/decrement sequence.
- Define the length of the number.
- Append prefixes and suffixes (characters) to sequential numbers.

The `getnumb.fc` application is called from the Subroutines process of Format Control to establish the sequential numbering sequence.

Sequential number file

The Subroutines process of Format Control creates a Sequential Number File record when the application `getnumb.fc` is called. The parameters passed to the application from within Subroutines define the values in the number file fields. These values determine how the sequential number opens and what other information it contains. You can define number file values either directly in the number file or via parameters passed from Format Control. However, if you use the number file to change values also defined in the Format Control subroutines process, they are overridden by the Format Control values.

Data types of sequential numbers

There are two data types of sequential numbers:

- Simple numbers. For example, 1, 2, 3.
- Complex character strings. For example, (DEV00001/WS).

Note: If you want to use the prefix and suffix capabilities of sequential numbering, the field being incremented or decremented must be defined in the database dictionary as type character. In all other cases, the field must be of type number.

Create simple sequential numbers

Applies to User Roles:

System Administrator

This example illustrates the process of setting up sequential numbering using Format Control.

To create simple sequential numbers:

1. ["Create the form for simple sequential numbers" below.](#)
2. ["Create a database file for simple sequential numbers" below.](#)
3. ["Create a Format Control record for simple sequential numbers" on the next page.](#)
4. ["Open the number file" on page 265.](#)
5. ["Add a numbered record to the database file for simple sequential numbers" on page 265.](#)

Create the form for simple sequential numbers

Applies to User Roles:

System Administrator

Create a form using the Form Wizard in Forms Designer called employees. The form will record simple employee data, such as name, address, and telephone number. Be sure to include a field labeled Employee ID with an input value of employee.id.

Create a database file for simple sequential numbers

Applies to User Roles:

System Administrator

To create a database file for simple sequential numbers:

1. Click **Tailoring > Forms Designer**.
2. Type employees in the form field, and click **Search**.

Service Manager opens the employees form.
3. Click **Create File** in the More Actions menu.
4. Type employees in the filename field.
5. Click **OK** to create the new data file.

6. Change the Employee ID field of the employee file to a number data type.
 - a. Select the employee.id field with the cursor.
 - b. Click **Edit**.
 - c. Click **number** in the Type field menu.

Important: A data type of number is required for all simple sequential numbering.

Create a Format Control record for simple sequential numbers

Applies to User Roles:

System Administrator

To create a Format Control record for simple sequential numbers:

1. Click **Tailoring > Format Control**
2. Type employees in the Name field.
3. Click **New**.
4. Click **Subroutines** in the More Actions menu or click the **Subroutines** button.
5. Click **Show Expanded Form** in the More Actions menu.

Use the following values to complete the Subroutines process:

Application Name	Add	Before	Error Message
getnumb.fc	true	true	Sequential number did not process correctly!

6.

Names	Definition	Values
record	The record in which the number will be placed. In this example, the employees data file.	\$file
name	The name to appear in the Class field of the Number file generated by the Format Control record.	employees
prompt	The field within the record receiving the sequential number.	employee.id

7. Click **OK** to save the record.

Open the number file

Applies to User Roles:

System Administrator

To open the number file:

1. Click **Tailoring > Database Manager**.
2. Type number in the Form field of the Database Manager dialogue box.
3. Click **Search**.
4. When the blank Sequential Number File form opens, click **Search** to open a record list.
5. Select employees from the list of records. Service Manager opens the Sequential Number File for the employees file.

Add a numbered record to the database file for simple sequential numbers

Applies to User Roles:

System Administrator

To add a numbered record to the database file for simple sequential numbers:

1. Click **Tailoring > Database Manager**.
2. Type employees in the Form field.
3. Click **Search**.
4. Add data to the blank Employee Database record, but leave the Employee ID field blank.
5. Click **Add**.

Service Manager adds the sequential number to the Employee ID field before the record is added to the database.

Create sequential number prefixes and suffixes

Applies to User Roles:

System Administrator

Sequential numbering is a subroutine process of Format Control that automatically adds identifying numbers to database records as they are created for Configuration Management, incident records, or employee records. Prefixes and suffixes are used to embed information in the sequential number and provide additional levels of identification. For example, each device in the database might be described by a logical name in the form of DEV(number)/(device type). For workstations, the device type might be WS. In this case, DEV is the prefix for the sequential number and /WS is the suffix.

For this example, see the task topics linked to below that will allow you to add a subroutine call to the device.workstation form to assign a unique device number to all records added to the file from this form.

To create sequential number prefixes and suffixes:

1. ["Create a Format Control record with prefixes and suffixes" below.](#)
2. ["Add a record with prefixes and suffixes" on the next page.](#)

Create a Format Control record with prefixes and suffixes

Applies to User Roles:

System Administrator

Sequential numbering is a subroutine process of Format Control that automatically adds identifying numbers to database records as they are created for Configuration Management, incident records, or employee records. Prefixes and suffixes are used to embed information in the sequential number and provide additional levels of identification. For example, each device in the database might be described by a logical name in the form of DEV(number)/(device type). For workstations, the device type might be WS. In this case, DEV is the prefix for the sequential number and /WS is the suffix.

To create a format control record with prefixes and suffixes:

1. Click **Tailoring > Format Control**.
2. Type device.workstation in the Name field of the Format Control Initializations form.
3. Click **Search**.
4. Click **Subroutines** in the More Actions menu or click the **Subroutines** button.

Use the following values to complete the Subroutines process:

Application Name	Add	Before	Error Message
getnumb.fc	true	true	Sequential number did not process correctly!

5.

Names	Definition	Values
record	The record in which the number is placed. In this example, the device.workstation record.	\$file
name	The sequential number class to be used.	devices
prompt	The field within the record receiving the sequential number.	logical.name
string1	The string (prefix) to be added to the front of the sequential number when creating a new number.	DEV
query	The string (suffix) to be added to the end of the sequential number when creating a new record.	/WS
text	Defines the sequential number type (number or string). Since characters occur in the number, this must be string.	string
number1	Determines the length of the sequential number.	val("4", 1)

Add a record with prefixes and suffixes

Applies to User Roles:

System Administrator

Sequential numbering is a subroutine process of Format Control that automatically adds identifying numbers to database records as they are created for Configuration Management, incident records, or employee records. Prefixes and suffixes are used to embed information in the sequential number and provide additional levels of identification. For example, each device in the database might be described by a logical name in the form of DEV(number)/(device type). For workstations, the device type might be WS. In this case, DEV is the prefix for the sequential number and /WS is the suffix..

To add a record with prefixes and suffixes:

1. Click **Tailoring > Database Manager**.
2. Type device.workstation in the Form field.
3. Click **Search**.

4. Add data to the blank Workstation record.
5. Click **Add**.

Service Manager adds a complex sequential number to the Asset field before the record is added to the database.

Notice that the complex sequential number DEV0002/WS appears in the Asset field (logical.name). Add another record, and the sequential number advances to DEV0003/WS.

Array maintenance

Two types of arrays exist within Service Manager: simple arrays, consisting of fields of similar data types; and arrays of structures composed of related fields of similar or dissimilar data types. A series of applications exists in Service Manager that can be called from the Additional Options process of Format Control to perform various maintenance functions on array structures. A single application called from the Subroutines process allows the user to sort simple arrays automatically when adding or updating records.

Array structure maintenance functions

When using functions, such as copy, move, delete, and insert in the More Actions menu, these functions act against the entire array structure element, not against a single field within the structure.

The array structure editor reopens the field in question in an expanded, dynamic array field. The form that opens when starting the function is the same form that opens during execution. The routines that perform these functions are available collectively and individually to any RAD routine or Format Control Additional Options definition.

The following general rules apply to array structure maintenance:

- None of the routines perform update functions to the database. To save the changes, you must select the applicable update function within the application that called the maintenance functions.
- The applications are cursor dependent. For example, the insert function inserts a blank element at the cursor position within the array structure. If the cursor is placed in any field within an element of the array structure, the function is performed against that entire element.
- If the functions are executed against a non-array structure field, the procedure normally fails and either issues an invalid data-type message or produces unpredictable results.

The array structure maintenance RAD applications add buttons to the form that enable users to perform tasks, such as moving, copying, deleting or sorting lines in array structures.

They include:

- `as.copy`
- `as.delete`
- `as.get.name`
- `as.insert`
- `as.move`
- `as.options`
- `as.sort`

You can call the array structure maintenance RAD applications from the Additional Options Format Control process. You will need a thorough knowledge of Forms Designer and the use of subforms in the creation of array structures.

For more information on each application, a detailed explanation of how to define Additional Options, and how to pass data to these subroutines, see the related topics.

Note: RAD users should follow normal subroutine call procedures to execute these applications.

Add array structure maintenance options to a format control record

Applies to User Roles:

System Administrator

In this example we add the RAD application `as.options` to the format control record for the `servicecontract` form. The same general instructions work for any form.

To add array structure maintenance options to a format control record:

1. Click **Tailoring > Format Control**.
2. Type `servicecontract` in the Name field and press **Enter**.
3. If a Format Control record does not already exist for the form, click **New** to create one.
4. Click the **Add Options** tab.

5. Type the following values into the appropriate fields. You can click **Show Expanded Form** from the More Actions menu if you prefer to work in it.

Field	Value
Option	2
Description	Array Options
Condition for Option	true
Reset on Return	true
Application	as.options
Error Message	Could not reset
Names	file name
Values	\$file servicecontract

6. Click **Save**. Service Manager updates the Format Control record.

Test the array structure maintenance options I added

Applies to User Roles:

System Administrator

In this example, we test the option added to Format Control record of the servicecontract form. The same general instructions work for testing any Format Control record.

To test the array structure maintenance options you added:

1. Click **Tailoring > Database Manager**.
2. Type servicecontract in the Form field and press **Enter**.
3. Select servicecontract from the list. Service Manager displays a blank Service Contract form.
4. Click **Search**.
5. Select a record from the list.

6. Click the **Named User** tab and insert your cursor in the array.
7. Click **Array Options** in the More Actions menu. The System Tray in the Service Contract form displays the buttons associated with the `as.options` subroutine.
 - Click **Insert** to insert a blank line in an array structure above the cursor position.
 - Click **Delete** to delete the line in which the cursor is placed.
 - Click **Copy** to copy the entry from the line selected by the cursor.

The original element is left in place when the copy is inserted into the array structure.

- Click **Move** to copy the entry from the line selected by the cursor.

The original element is deleted when the copy is inserted into the array structure.

The System Tray display these buttons to complete the Copy and Move functions:

- **Insert** inserts the copied element into the array structure above the line selected by the cursor.
- **Replace** replaces the element selected by the cursor with the copied element.

8. Test the functionality of the buttons by inserting, copying, moving, and deleting named users.
9. Click **End** to restore the normal tray buttons.
10. Click **OK** or **Save** to save any changes you have made to the Service Agreement record.

You must use **OK** or **Save** to update your record. Pressing **Enter** will NOT save any changes you have made.

Sorting simple arrays

The `sort.array` application is a utility called from Format Control to sort simple arrays in either ascending or descending order. Supported array data types are:

- number
- character
- data/time

The `sort.array` utility is called from the Subroutines process of Format Control. Use this utility to sort simple arrays in ascending or descending order

Sort simple arrays

Applies to User Roles:

System Administrator

The sort.array application is a utility called from Format Control to sort simple arrays in either ascending or descending order. Supported array data types are:

- number
- character
- data/time

To sort simple arrays, follow the procedures in these task topics:

1. ["Create the form for sorting simple arrays" below.](#)
2. ["Create the Format Control record for sorting simple arrays" on the next page.](#)
3. ["Create a data record for sorting simple arrays" on the next page.](#)

Create the form for sorting simple arrays

Applies to User Roles:

System Administrator

To create a form for sorting simple arrays, use the Form Wizard in Forms Designer called demo.sort. It has three simple arrays of different data types: numbers, characters, and dates. Provide a Name field for the record and create a file in the database dictionary called demosort.

Use the following values to construct your form:

Field Name	Input Value
Name	name
Numbers	number
Characters	character
Dates	date

Create the Format Control record for sorting simple arrays

Applies to User Roles:

System Administrator

To create the Format Control record for sorting simple arrays, you will call the `sort.array` application from the Subroutines process of Format Control as follows:

1. Click **Tailoring > Format Control**
2. Type `demo.sort` in the Name field.
3. Click **New**.
4. Click **Subroutines** in the More Actions menu or click the **Subroutines** button.
5. Click **Show Expanded Form** in the More Actions menu.

Use the following values to complete the Subroutines process:

Application Name	Add	Update	Before	Error Message
<code>sort.array</code>	true	true	true	Could not sort the array.

6.

Names	Definition	Values
file	When calling <code>sort.array</code> from Format Control, always pass <code>\$file</code> to this parameter.	<code>\$file</code>
name	The name of the input field (array) in <code>\$file</code> you wish to sort.	number
boolean1	The parameter that controls the sort order. A value of true sorts in ascending order. If you pass false, the sort order is descending. The default is true.	val ("false", 4) *

For a detailed discussion of the `val` function, refer to System Language Help.

7. Click **OK** to save your record.

Create a data record for sorting simple arrays

Applies to User Roles:

System Administrator

Once you have created the form and Format Control for sorting simple arrays, you are ready to create the data record for sorting simple arrays.

To create a data record for sorting simple arrays:

1. Click **Tailoring > Database Manager**.
2. Type demo.sort in the Form field of the Database Manager dialogue box.
3. Click **Search**.
4. Type sort1 in the Name field of your test form.
5. Enter data in the arrays in random order. For example, numbers in the Numbers array and characters in the Characters array.
6. Click **Add** to add the record to the database and sort the array.

Note: Since Update evaluates to true in the Format Control record for this form, you may also sort an array by clicking **Save**.

The Numbers array is sorted in descending order.

7. Repeat the process for the other arrays by changing the name parameter in the Format Control record to reflect the array being sorted.
8. Remove the boolean1 parameter, and the sort order defaults to true. The array is sorted in ascending order.

Determine parameters without using the RAD Editor

Applies to User Roles:

System Administrator

You can determine the parameters for any application in Service Manager without using the RAD Editor. Use Database Manager to display the input value of each field in a particular application parameter panel. This value is the parameter name used in the Subroutines process of Format Control.

To determine parameters without using the RAD Editor:

1. Click **Tailoring > Database Manager**.
2. Type format in the Form field, and click **Search**.

3. Click **Format** in the record list.
4. Type the name of the application whose parameter values you want to view (for example, getnumb.fc) in the Format Name field of the blank Format form.
5. Click **Search**.

The parameter Name opens in the Input field of the form and the function appears in the Label field.

6. Scroll down to view all the parameters listed.
7. Click **OK** and exit this record without saving it.

Invoke auditing from Format Control

Applies to User Roles:

System Administrator

The topics below comprise an example of how to use Database record auditing. The steps in these topics must be followed in order. If you skip a step, Database record auditing may not work.

1. ["Open the audit specifications table" on the next page.](#)
2. ["Add an audit specifications record" on page 279.](#)
3. ["Define an audit specifications entry" on page 279.](#)
4. ["Invoke auditing" on page 281.](#)
5. ["Set up event triggers" on page 282.](#)
6. ["Add lookup functionality to Format Control" on page 284.](#)
7. ["Test audit lookup functionality" on the next page.](#)

To invoke auditing from Format Control, perform the following steps:

1. Create an Audit Specifications record for a particular file.
2. Open the Format Control record associated with the form and the file for which you created the audit specifications record in step 1.

3. Select the **Save Copy** option.
4. Click **Subroutines** to display the Subroutines form.
The Subroutines panel opens.
5. Enter the desired format control. For this example, enter:

Field	Enter
Add	true
Upd	true
Before	true
Application	audit.compare
Error Message	Audit Processing could not complete.
Names	file second.file
Values	\$file0 \$file

Open the audit specifications table

Applies to User Roles:

System Administrator

Use one of the following methods to open the audit specifications table:

- From the System Navigator, click **Tailoring > Audit** and then double-click **Audit Specifications**.
- From the Command line, type `audspec` from the command line and press **Enter** or click **Execute Command**.
- From the System Navigator, click **System Definition > Tables > auditspecs > forms** and then double-click **auditspecs**.

Test audit lookup functionality

Applies to User Roles:

System Administrator

Some files may not be configured for Audit Lookup. In that case, you must add Audit Lookup functionality before Audit Lookup will work.

HP Service Manager invokes Audit Processing using the definitions provided in the Format Control, when a record is added or updated in a file for which auditing has been set up.

Note: Before using Audit Lookup, be sure that the Audit Specifications include all fields that need auditing.

To test Audit Lookup functionality:

1. Add a record to the contacts file using the values in the table below.

Field	Value
Contact Name	MILLER, JOHN
Employee ID	NEW00003
Last Name	Miller
First Name	John
Company	advantage
Email	john.miller@Advantage.com
Dept Name	Documentation

Note: Service Manager displays the following message: *“Contact Information record added.”* The audit process is invoked. Service Manager does not open an audit record until a user changes the current field values.

Service Manager has recorded all fields specified in the Audit Specifications record. All Old values are NULL or contain no data because there is no previous version of the record; this is a new record.

Note: Audit Log entries are created in response to any changes made to the current values in this record. These audit records are displayed in the **audit.summ** form, when accessed using the Format Control option created earlier.

If you check Audit Lookup now, a message informing you that there are no audit records.

2. To complete the example, modify the record created in the previous steps by entering new values in the fields of the form that were specified as Unique A-D fields in the audit specifications file. For this example, change the Employee ID to NEW00004.

3. Click **Save** to commit the changes to the database.

Service Manager displays the following message: *"Record updated in the contacts file. Audit Record successfully recorded and added."* when the record is saved and an Audit Log is created.

4. Open the More Actions menu and select **Audit Lookup**.

A record list opens.

Note: The updates you see may vary, depending on how you have your triggers set up for the file.

5. Select the desired record from the list.

Note: Only those fields which were modified are recorded. Service Manager displays both the Old and the New versions of each modified field in the audit log record. If none of the fields defined in the Audit Specifications Table have been modified, then Service Manager does not generate an audit log entry for that database dictionary file.

6. For further information regarding the modifications to the **contacts** file, click **Show detail**.

The same record is re-displayed in an expanded form audit.g), which shows more detailed information for the arrayed fields, as well as scrollable fields.

7. Click **End** to return to the summary form.

8. From the summary form, click the **Prev** and **Next** buttons to review any additional audit records in the log for this contacts record.

9. Click **End** to return to the Contacts record.

The topics below comprise an example of how to use Database record auditing. Follow the steps in this order. If you skip a step, the example will not work.

To use Audit Lookup:

1. ["Open the audit specifications table" on page 276.](#)
2. ["Add an audit specifications record" on the next page.](#)
3. ["Define an audit specifications entry" on the next page.](#)

4. ["Invoke auditing" on page 281.](#)
5. ["Set up event triggers" on page 282.](#)
6. ["Add lookup functionality to Format Control" on page 284.](#)
7. ["Test audit lookup functionality" on page 276](#)

Add an audit specifications record

Applies to User Roles:

System Administrator

To add an audit specification record:

1. Click **Tailoring > Audit > Audit Specifications.**
2. Type a file name in the **Filename** field.

In this example, type **contacts**.

3. Enter unique keys in the Unique A, B, C, or D field.

The Unique key for the contacts file is contact.name. In this example, type **contact.name** in the Unique A field and leave Unique B through D as NULL.

4. Click **Add.**

When Service Manager records an Audit Log record, for example, the contact name Teresa Chan, the Filename field is recorded in the Audit Log as contacts and the Unique A field is recorded as TERESA CHAN.

The contacts Audit Log record then has a unique association with a contacts record. You can generate new Audit Logs based on the data found in the previous log record for a specific device.

Note: A *device* is also referred to as a *Configuration Item (CI)*.

Define an audit specifications entry

Applies to User Roles:

System Administrator

The system validates Filename and Field Name values every time you update an existing record or add a new record. The Audit utility safeguard system prevents records with misspelled and incorrect file names or field names from being processed. Such errors potentially could cause faulty communication within the database.

Note: The name of the file and one field name of this example have been entered incorrectly to illustrate error-correction process built into the Audit utility.

To enter data in the Audit Specifications file:

1. Click **Tailoring > Audit > Audit Specifications**.

A blank Audit Specifications Table form opens.

2. To select the file to create the specifications for.

Enter information in the **Filename** field and **Unique A** through **Unique D** fields, as necessary, to parallel the Unique key in the source file with the Audit Log.

Note: see the related topics and read the topic "Audit specifications file description" for more information on input fields.

For this example, enter `contacts` in the **Filename** field, and then click **Search**.

Caution: Making changes to the **contacts** file to cause it to invoke auditing will cause two audit records to be generated when a file is updated. To prevent this, use a backup, and restore the original file when finished with the example, or create a different file to practice on.

3. Define those fields you want Service Manager to audit and any aliases. If a field name is invalid, a list will open up allowing you to copy a valid name and use it to replace the invalid name.
For this example, enter:

Field Name	Alias
contact.name	Contact Name
user.id	Employee ID
first.name	First Name
last.name	Last Name
dept.name	Department
email	Email Address

Field Name	Alias
location	Location

4. Click **Add**. to retain this record and commit it to the database.

The topics below comprise an example of how to use Database record auditing. Follow the steps in this order. If you skip a step, the example will not work.

1. ["Open the audit specifications table" on page 276.](#)
2. ["Add an audit specifications record" on page 279.](#)
3. ["Define an audit specifications entry" on page 279](#)
4. ["Invoke auditing" below.](#)
5. ["Set up event triggers" on the next page](#)
6. ["Add lookup functionality to Format Control" on page 284.](#)
7. ["Test audit lookup functionality" on page 276.](#)

Invoke auditing

Applies to User Roles:

System Administrator

You can invoke auditing from Format Control. If you are using the Change Management application, you can invoke auditing from Database Manager.

Select one of the following options to invoke auditing:

- ["Invoke auditing from Format Control" on page 275](#)
- [Invoke auditing from the file in Database Manager](#)

Invoke auditing for joindefs tables

Applies to User Roles:

System Administrator

To use the database record auditing functionality, you need to invoke auditing for joindefs tables, as described in the following tasks.

Task 1: Temporarily disable the auditspecs.check Subroutine on the auditspecs Format Control record.

1. Log in to the Windows client.
2. In Format Control, open the **auditspecs** record (Name: **auditspecs**).
3. Click **Subroutines**, and from the More Actions menu select **Show Expanded Form**.
4. For each of the conditions Add, Update, and Before, append the following expression: **and false**.

The conditions for each should be: **true and false**.

5. Click **Save**.

Note: Re-enable the Subroutine after performing Task 2.

Task 2: Add an Audit Specifications record for joinbizservice and each of the other "join" files being used.

1. Click **Tailoring > Audit > Audit Specifications**.
2. Enter field data as follows:

Field	Value
Filename	joinbizservice
Unique A	logical.name
Field Name	logical.name (and any other fields you want in the list)

3. Click **Add**.
4. Repeat for each additional "join" table desired.

Set up event triggers

Applies to User Roles:

System Administrator

Event triggers are part of the Service Manager auditing function. A trigger can be set up to invoke the auditing application, `audit.compare`.

Note: If an audit is needed for Configuration Management, do not use triggers. Instead, use Format Control to invoke the `audit.compare` application.

To set up the `audit.compare` trigger:

1. Open the triggers form in Database Manager.
2. Enter the name of the new trigger. For this example, type: `example.trigger.audit.update`.

Note: Each trigger performs one action; therefore, when naming the trigger, you may want to incorporate the trigger type (when the trigger is to execute) into the name of the trigger. For example, `trigger.audit.add` could be the name of a trigger that executes whenever a record is added.

3. Enter the name of the file you want Service Manager to audit in the **Table Name** field. For this example, type: `contacts`.
4. Select the type of trigger, according to the number legend displayed beside the field. For this example, select: 4
5. Enter the application name in the **Application** field. For this example, type: `trigger.invoke.auditor`.
6. Click **Add** when ready to add the record to the file.

Service Manager displays the following message: *"Trigger record added."*

The topics below comprise an example of how to use Database record auditing. Follow the steps in this order. If you skip a step, the example will not work.

1. ["Open the audit specifications table" on page 276](#)
2. ["Add an audit specifications record" on page 279](#)
3. ["Define an audit specifications entry" on page 279](#)
4. ["Invoke auditing" on page 281](#)
5. ["Set up event triggers" on the previous page](#)

6. ["Add lookup functionality to Format Control" below](#)
7. ["Test audit lookup functionality" on page 276](#)

Add lookup functionality to Format Control

Applies to User Roles:

System Administrator

The topics below comprise an example of how to use Database record auditing. The steps in these topics must be followed in order. If you skip a step, Database record auditing may not work.

1. ["Open the audit specifications table" on page 276.](#)
2. ["Add an audit specifications record" on page 279.](#)
3. ["Define an audit specifications entry" on page 279.](#)
4. ["Invoke auditing" on page 281.](#)
5. ["Set up event triggers" on page 282.](#)
6. ["Add lookup functionality to Format Control" above.](#)
7. ["Test audit lookup functionality" on page 276.](#)

Note: Before using Audit Lookup, be sure that the Audit Specifications include all fields that need auditing.

To add Lookup Functionality, perform the following steps:

1. Access the Format Control record associated with the form or file.
2. Select the **Save Copy**.
3. Click **Additional Options**.
4. Select **Additional Options**.
The Additional Options panel opens.
5. Activate Lookup Functionality by adding the following specifications:

Field	Contents
Option	1
Desc	Audit Lookup
Condition	true
Application	audit.lookup
Comment	Audit Lookup
Message	Could not call Audit Lookup application.
Names	file
Values	\$file

Special processing considerations: Incident Management

In Incident Management, each action taken on an incident record has the option of adding an additional page; therefore, update and delete operations are not truly performed. Even though there is a Save button, the application is really adding a new record to the `problem` table. When you close an incident, a page can be added to the file with a status of closed. Consequently, when using Format Control on Incident Management, the add field must evaluate to true.

Format Control records are only executed on the initial, browse, open, update, and close forms for Incident Management.

Format Control can be placed on the following forms, or by using the master Format Control called `probsummary`.

- `IM.open.incident`
- `IM.update.incident`
- `IM.close.incident`
- `IM.browse`

Special processing considerations: Change Management

Change Management enables you to define three types of Format Control records:

- Detail
- Master
- Approval

The detail and master records execute during standard add, update, and close processing. The approval record executes during Change Management approval processing.

Change Management uses the four processing functions as follows:

- Add
Processed at open time.
- Update
Processed at update time (update, reopen).
- Delete
Processed at close time.
- Display
Processed when an item is displayed.

The Format Control components executes after the process invokes but before the record permanently updates. For instance, the add function executes after the user clicks on the Open button in either the Request or Task structure, but before the Request/Task saves to the database. The display function executes after a record is selected from the record list but before the record opens.

You can execute both a master Format Control record and a detail Format Control record for each Request/Task process. The master Format Control record executes before the detail Format Control Record. If any of the Format Control components fail for any reason, the user always returns to the previously open screen with the appropriate error messages.

Special processing considerations: master Format Control record

Change Management supports a master Format Control record that enables you to define in one record the Format Control statements that apply to all Change Management Request Phases. The name of the master Format Control record is cm3r (cm3t for Tasks). The options on this Format Control record execute during all Request/Task processing (including background processing) except approval processing. The master format control record processes before the detail Format Control record.

Special processing considerations: detail Format Control record

To enforce processing rules that are unique to a Phase, define a format control record that is the same name as the default View of that Phase. The add, update, delete, and display features process just as they are with detail Format Control definitions.

Special processing considerations: approval Format Control record

The Change Management approval process checks for a Format Control record that associates with the approvals view of the Phase.

- The add features are processed when a Request is approved.
- The update features are processed when a Request is disapproved.
- The delete features are processed when a Request is unapproved.

The master Format Control record does not execute during approval processing.

Format Control and eventout records

When incidents are opened, updated or closed by Event Services, a record is written to the eventout file. This record contains information from the incident (described in the output eventmap record for the event) that is passed to an external process via the SCAuto/IPAS external interface. You can elect to write to the eventout file when Service Desk operators open and close records so that the information is passed to the external interface.

The `aces.write` application creates a character string of fields from a structure and writes them to eventout. An Event Registration record identifies the event type and the name of the Event Map records used to define which fields will be selected from the record. The application should be called as a Format Control Subroutine passing two parameters:

- The record from which data will be mapped.
- The Event Type, as defined in the Event Register.

To write to eventout on incident close, the Format Control record is attached to the `problem.equipment.close` form using the following values:

Field	Value
Application	axces.write
axces.write	true

Name	Value
record	\$file
name	pmc

Note: This procedure is not specific to Incident Management. You can write eventout records for other applications, such as Configuration Management or Change Management.

Format Control error messages

There are several error messages you receive if you enter an incorrect expression into a Format Control record. These errors take several forms:

- If any syntax errors are present when the system attempts to parse (evaluate) an expression field (for example, Calculation, Initialization) in a Format Control record, the following message opens in the Status Bar at the bottom of the screen when you click on the Back button or attempt to select another function from the More Actions menu:
Field contains an invalid expression.
In the case of incorrect syntax in a Calculation expression, the system does not let you exit the Format Control record until the error is resolved. If you cannot resolve it, delete the line and re-enter the desired input when the correct syntax is determined.
In the case of incorrect syntax in an Initialization Expression, the system does not let you access additional Format Control functions (Subroutines, Additional Options, etc.) from the More Actions menu.
- The system will accept a properly constructed Initialization Expression in a Format Control record that may not be accepted by the called application at run time. When an application cannot recognize an Initialization Expression, the Format Control function fails, and Service Manager displays this message: *Cannot evaluate initialization expression #:1*
- Format Control accepts properly constructed but conflicting expressions within a record.
For example:
Initialization expression: \$x="ibm"
Calculation expression: \$x+=1
When Format Control attempts to perform this calculation on a non-numeric field at run time, the

system displays the following error message:
Wrong or mismatched type in increment or decrement

Common routines called from Format Control

The following is a list of RAD routines that have a specific function within Format Control:

- as.copy
- as.delete
- as.get.name
- as.insert
- as.move
- as.options
- as.sort
- axes.write
- database
- fill.fc
- fingerprint
- getnumb.fc
- marquee.publish
- marquee.send
- message.fc
- post.fc
- publish
- sort.array

- `query.stored`
- `validate.fields`

Links

One of the advantages of a relational database is the elimination of redundant information. You accomplish this by storing information about a particular subject in one place or file that has links to other subjects. Links are a combination of data and link definitions, sets of conditions containing the relationships for linked information. Use links in Incident Management and Configuration Management environments to relate information in one file to information in another.

Link records define the relationship among sets of data. They are used extensively throughout the system and the main topics include the following:

- Understanding links — general linking within the system
- Link maintenance — using links effectively
- Virtual joins — are used to display read-only data from another file.
- Posting — copy data from a source record to a target record.

Understanding links

These topics discuss linking within the system. The topics demonstrate how to:

- Access link definitions
- Add a new link
- Modify an existing link
- Use advanced link editing features
- Link dependencies with the Service Desk environment

Advanced link editing features

Simple links define the relationships between a specific field in a source record and a specific field in a target record. The Find and Fill options perform in a straightforward manner. A link query is built based upon a search argument, which contains the value of a field in the source file and the name of a field in the target file. These fields need to be Unique key fields for simple linking.

The system provides extended flexibility to define complex link expressions. For example, you can:

- Define a specific query that uses more than one field to form the link query.
- Specify the record list format that should be used when the link query finds more than one record.
- Define a variable, rather than a specific name, to be used as the target file.
- Manipulate the value of fields by using link expressions.
- Specify that particular fields are copied from the target file to specific fields in the source file during the Fill operation without a requirement for identical field names.
- Include multiple non-keyed fields on the link line structure of a keyed field, enabling those fields to be copied to the source form along with the data retrieved from the keyed field query.

Note: Advanced link features are not available when using Virtual Join.

Specifying a link query

If you want to control the search criteria used by Find and Fill, define a specific link query.

There can be instances when the link is dependent on more than the simple value of fields in the records. For example, links from an incident document could be dependent on the device type. Link Maintenance allows specification of expressions that are evaluated by the Fill and Find operations before the link query is built.

Whenever the link relationship varies according to data in the source record, using expressions and a variable for the target file is the most efficient method for establishing a link.

Expressions process in the order they are listed on the form.

Note: Be aware that if the \$query variable is used in link records, each \$query value is valid only in the current thread. This means a \$query value might be overwritten by executing a second link query. To avoid this issue, you can use \$L.query instead of \$query in these cases.

Copying fields by name during fill operations

Normally, the Fill operation copies fields of the same name from the target file to the source file using a function called project.

For example, if you open an incident document and place the cursor on the `logical.name` input field and then choose the Fill option, the system searches the device file for a record with the same `logical.name`. The system then copies fields with the same name from the device file into the incident document.

The incident document contains information about the device that copied from like-named fields in the device record.

As long as the field names are the same, and the information being copied is from an equal or higher level, the project function performs normally. When the field names are dissimilar, or if the information to be copied is from structure fields to scalar fields, you must use a different method.

Scalar/non-scalar field links

A simple link cannot copy information from a non-scalar field (composed of more than one data element of the same type) to a scalar field (composed of a single data element). For example, information from a record in the incident file cannot be projected into a `probsummary` record. Even though the field names are the same, the fields in the incident record are not scalar fields. They exist in one of the three structures that make up the `problem` table descriptor. The fully-qualified name of the assignment field in the incident record is really `header,assignment`.

The system can project the `logical.name` value in the incident file into the incident record's `logical.name` field. It cannot project the incident file's non-scalar `middle,logical.name` field into the scalar `logical.name` field in the incident file. To copy non-scalar information to a scalar field, the Fill operation copies each field using instructions in the link record.

Keeping changes

There is no Update option when editing a link line entry. A copy of the link line entry is made when it is selected. That copy is compared to the current line entry when you exit using Back, Close Application, Next Entry or Previous Entry.

Remember that when you modify a link line entry, the change does not write to the database until you exit from the link record and confirm the update action.

Link dependencies within Service Desk

The Service Desk applications use two special link records. They perform the conversion of `problem` records to `probsummary` records and the conversion of inventory files to the entity and attribute files.

- The `build.inventory.files` link record converts information in the various inventory files to the Configuration Management device file and their associated attribute files. This link record cannot be used for any other purpose since the `$file` variable has special meaning in the conversion application.
- The `build.problem.summary2` copies information from the problem record in the rootcause table to the incident record in the probsummary table whenever the appropriate problem record is updated. This link record uses the `$pfile` variable in place of the normal `$File` variable when building expressions. It processes expressions after copying the fields, rather than before as in normal link processing.

Document Engine master link record

For each object accessed through the Document Engine, a master link record combines with the form-dependent link record. The master link record is valid for the entire file, and always has the same name as the Object/filename. For example, if the filename is contacts, the master link record is the contacts link record. This eliminates redundant links across many records for ease of maintenance and for consistency.

Add a new link file

Applies to User Roles:

System Administrator

To add a new link file:

1. Click **Tailoring > Tailoring Tools > Links**.
2. In the Name field, type the name of the form or file. If desired, type a descriptive name or phrase in the Description field.
3. Click **New** to create the new link record.
The record opens for editing.
4. Type the relationships in the appropriate columns.
5. Click **Save** to add the new link record.

Modify an existing link

User Role: System Administrator

To modify an existing link.

1. Click **Tailoring > Tailoring Tools > Links**.
2. Type optional search criteria, and then click **Search**.
3. Double-click a link to view or change it. If necessary, press Ctrl+H to view help for each field. When the link record opens, the following options are available from the More Actions menu:

Option	Value
Insert Line	Opens a window to prompt for the number of lines to insert, then inserts them above the cursor position.
Delete Line	Opens a window to prompt for the number of lines to delete, then deletes them beginning with the line the cursor is on.
Select Line	Allows advanced link processing.
Check Field	When the cursor is positioned on a Source Field Name or Target Field Name, prompts for a file name and then checks the database dictionary of the file to determine whether a field of that name exists. If invalid, allows selection of a valid field.

4. If you make changes, click **Save**.

Test a link

Applies to User Roles:

System Administrator

To test a link:

Note: The locations form is used for this example.

1. Click **Tailoring > Database Manager**.
2. Type **location** in the Form field, and then click **Search**.

The location file opens.

3. Type **P** in the parent Location field.
4. Click **Find**. A list of records with a link definition for this file opens.
5. The Parent field is linked to the Location Full Name field in the Link file for *location*.

Note: Link records are case-sensitive.

Delete a link

Applies to User Roles:

System Administrator

To delete a link:

1. Click **Tailoring > Tailoring Tools > Links**.
2. In the **Name** field, type a link name, and then click **Search..**

The link record opens.

3. Click **Delete** in the toolbar.

You are prompted to confirm that you want to delete the link record.

4. Click **Yes** to confirm that you want to delete the link record.

You are returned to the link record prompt screen.

Access advanced link maintenance

Applies to User Roles:

System Administrator

When you access a link record, the Source Field and Target Format/File Names copy to the link structure form. New fields are available and new option keys are enabled.

To access advanced link maintenance:

1. Click **Tailoring > Tailoring Tools > Links**.
2. Type an optional link name in the **Name** field, and then click **Search**.

A list of all link records opens.

3. Click the link to edit.

For example, click the locations link record.

4. Position the cursor on the line you want to edit.
5. Click **More** or the More Actions icon and then choose **Select Line**.

Copy information from structure fields to scalar fields

Applies to User Roles:

System Administrator

To copy information from structure fields to scalar fields (fields composed of a single data element):

1. Open a link record.
 - a. Click **Tailoring > Tailoring Tools > Links**.
 - b. In the Name field, type the name of the link record, and then click **Search**.
 - c. Select a record from the list.
2. Place the cursor on the line containing the link.
3. Click **Select line** from the More Actions menu.
4. Switch the fields by typing the structure field in Source Field (Fill To) and the scalar field in Target Field (Fill From). The contents of the structure fields are copied to the scalar fields.

Link maintenance

One of the advantages of a relational database is the elimination of redundant information. You accomplish this by storing information about a particular subject in one place or file that has links to other subjects. Links are a combination of data and link definitions, sets of conditions containing the

relationships for linked information. Use links in Incident Management and Configuration Management environments to relate information in one file to information in another.

Data relationships and the link file

Link maintenance involves establishing relationships between data so that information residing in one file can form the link query that selects, displays, or copies information from another file.

A separate link definition can occur for each form.

When a field is linked to more than one table, the order in which the entries appear in the link record determines the order in which Service Manager searches them. First, Service Manager searches the table at the top of the list. If no records are found to satisfy the link query, it searches the tables lower in the list, from top to bottom. It stops searching when the query is satisfied, or it finishes searching all tables in the list.

Types of links

Find, Fill, and virtual join links are the basis for the most powerful features of Service Manager.

- The Find function uses the value of the field in which the cursor is placed to query for information based on data stored in existing records. Security parameters restrict a user's access to the requested data. If information is found, a record list of matching configuration item (CI) records opens. If no information is found, a negative message opens.
- The Fill function selects information from existing records and copies it into the current record. For example, incident documents are populated with device information when a logical name (logical.name) is specified.
- The virtual join combines information from many files and presents the results on a single form. At the service desk, a vendor's telephone number can display in an incident document without occupying any space in the actual record. The information presented does not reside in long-term memory and is another way of viewing Configuration Management data.

Linking ensures current and consistent data across all incident documents. A change to the original configuration item (CI) record automatically updates the incident document. Each time an incident document opens or updates, the applicable fields displaying linked data pick up the most recent Configuration Management data.

Important: Queries for Find, Fill and virtual joins are performed only when run on fields that have been set up as unique keys in the database dictionary record for the target files. These unique key definitions

allow data to be stored using certain markers. Link queries need to locate these markers in order to retrieve the applicable data.

Search the Help for more information on setting up unique keys and other definitions in the database dictionary record.

Find functionality

Using information in the link record, Find (us.link) locates and displays information in another record (or records) based on the contents of the current record.

In an Incident Management example, the source record could contain *pc* in the field labeled Affected CI (logical.name field). The link record relates the logical.name field in the SOURCE record to the logical.name field in the vendor (TARGET) file.

All device records containing logical.name fields beginning with *pc* are selected and displayed in a record list for the user to select.

The user selects the specific device sought and opens the applicable device record.

Find uses the field name of the text box where the cursor currently is, in order to determine what relationship to establish. To select the device records with names that begin with *pc*, the cursor must be in the CI Name field that contains *pc*.

It builds the following query:

```
logical.name field in device file begins with "pc"
```

That translates in to the following link query that executes against the vendor file:

```
device#"pc"
```

Note: Using advanced features in Link Maintenance, you can apply rules to use a more complex link query for data selection. Refer to Using Advanced Link Editing Features Help for more information.

The Find option is available throughout the system. It is subject to security restrictions within the application. For example, Database Manager controls the Find option through the use of Format Control. In Incident Management, the profile determines whether Find is available for the current user and how it behaves.

The selected data records can be manipulated in accordance with security restrictions. Changes made to these records DO NOT modify the source record unless the user selects from an available list of options that allow updated information to be posted to the specific configuration item record. For more information on posting, refer to Format Control and Display Options Help.

Fill functionality

Using information in the link file, Fill (us.link) locates information in another record and copies it into the current record.

Fill does the following:

- Starts on a source record.
- Accesses a record in a secondary (target) file based on data in the source record (usually the contents of the input field selected by the cursor).
- Copies (fills) data from that target record back to the source record.

The link record for the probsummary file is used in Incident Management.

In the following example, Fill populates fields in an incident document based on the value of the field labeled CI Name.

Rather than selecting the device record for display, fields in the *device* (TARGET) file that have the same name as the fields in the *problem* (SOURCE) file project into the source record.

Fields on the source form populate automatically with data directly from the applicable device record. For example, CI Name (logical.name) on the incident record fills with the value of Configuration Item (logical.name) from the device record. The Type field on the incident form fills with the value of Type from the device record, and the value for the Category field copies from the device record to the incident record.

Through this transfer, the TARGET record is unchanged but the SOURCE record modifies to reflect the values of fields in the TARGET record.

The resulting modified SOURCE record is not written to the database until some action instructs the system to do so. For example, open or update the incident.

The Fill option uses the value in the field where the cursor is positioned to determine which link relationship to use. This example assumes that the cursor is in the CI Name (logical.name) input field when you select the Fill option.

Use Fill when it is necessary to store information in the source record so that it can be changed or used as a link to other information. If you simply need to display the information in a form, use Virtual Join.

Fill functionality with multi-select

The Fill Selected option is available from record lists resulting from a fill on an array field. When one or more records from a list are selected, the user can choose the Fill Selected option to fill the array with only the selected records.

- A multi-select fill will not overwrite the existing content of an array. The records selected will be inserted into the array from the element where the fill was initiated.
- The multi-select capability for a record list is determined by the field that the fill is being performed from.
 - If the field is a scalar the capability is disabled.
 - If the field is an array the capability is enabled.
- The multi-select functionality is not enabled in system-generated record lists. The System Administrator needs to create a form with a table for the capability to be available.
- When a Fill All or Fill Selected is performed, a message indicating how many records were filled displays in the message area.
- The post expressions defined in a link line are executed as each record is filled into an array when performing a Fill All or Fill Selected.

Turn off multi-select functionality

The Fill Selected option is available by default whenever a fill is performed on an array field. System Administrators can disable this option by setting the variable `$fill.mult` to false in the expressions of the link line for the field where the fill is being performed.

Note: The Multiple Selection property in Forms Designer has no effect on enabling or disabling the multi-select capability when performing a fill from an array field.

Virtual joining functionality

The Virtual Join function allows a single form to display information from many files. You cannot modify virtually joined information but it can link with other files using Find and Fill.

There are special requirements for virtual joins when specifying link information:

- The TARGET Format/File Name value in the link record must contain a file name not a form name.
- The target field must be a non-concatenated key in the target file.
- The target field must be the first instance of the key in the database dictionary's key array.
- The target field must be a scalar field (non-array).
- Virtual Join only executes simple links.

While the appearance of virtually joined information is the same as if it exists in the open record, the information is stored in one place and displayed on demand to another.

Note: Use virtual join in any form except record list forms. Use virtual join information as a source field to other information using Find or Fill. Nested virtual joins are not supported.

Us.link

This section describes the Universal Services- Link (us.link) application.

The us.link application replaces the old applications find, fill and fill.recurse for several reasons:

- To take advantage of the current Service Manager environment in order to speed up transactions.
- To have a common rule base behind each of these applications.
- To make the find/fill process easier to debug.

The us.link is responsible for selecting the correct records from the correct file and then passing control to either us.find or us.fill.

The find, fill, and fill.recurse applications are now single panels calling the us.link routine. No changes are necessary to existing code that calls one of these routines.

Variables used in links

The variables used in links are listed below. The application "us.link" is in charge of executing the link expressions using the values set in those variables. If values for the logical variables are not specified, they default to false. Unless noted, the special variables used by prior applications are also used by us.link.

Variable	Value	Use
\$fill.date.calendar	true or false	Enables a pop-up calendar window, and is used for date/time fields. Note: \$fill.date.calendar is no longer needed when using 6.2 or later clients, which allow you to define a date/time field on a form as a "calendar"-type widget.
\$fill.display.add	true or false	Determines whether you can add a record if the current link query returns no records.
\$fill.display	true or false	Determines whether the record you are filling from opens before you copy data from it into your source record.
\$fill.exact	true or false	Forces the project panel to adhere strictly to data types and index levels during the project.
\$find.skip	true or false	Tells the application to skip the current entry whenever a "find" is performed on the field.
\$fill.list	An array list of values. For example: \$fill.list={"a", "b", "c"}	Enables you to fill from an array list of values. Target file and field names are not required when using this option. For example, \$fill.list={"a","b","c"} will execute a pop-up window where the user can select one of the array values, a, b, or c.
\$fill.recurse.msg	A message. For example: \$fill.recurse.msg="Hello world"	Defines the message to appear on a recursive link (when \$fill.recurse=true).
\$fill.option.copy	true or false	This option is no longer necessary when using us.link.
\$fill.option.skip	true or false	Enables the skip option to skip the current link entry and move on when displaying the results of a recursive fill.
\$fill.recurse	true or false	Determines whether to move to the next entry in the link record (perform a recursive fill).
\$fill.replace	true or false	Governs whether a field that contains data should be overwritten with new data (including NULL). Only used for the project portion of a fill, not fields specified on the fill to/fill from section of the link line.
\$fill.search.format	Form to use when performing a fill. For example: \$fill.search.format="company"	Defines the search screen to use when performing a fill. If this variable is set in the link expression, and the field that you are filling is NULL, the indicated search screen

Variable	Value	Use
		opens. The search created here continues the standard fill process.
\$fill.skip	true or false	Tells the application to skip the current entry and continue based upon the value of \$fill.recurse for that entry.
\$fill.structure	Specific link. For example: \$fill.structure={1,"rad"}	<p>Enables you to perform a find from a field that is part of a structured array and to use fill to move information into that specific element of the structured array when:</p> <ul style="list-style-type: none"> - The field name to fill from/post to is part of the structured array. - The field name to fill from/post to is not used in any other structure of the database dictionary record. - The name of the structure must be the same as the name of the array of which it is a part. <p>To use find/fill on a structured array, you must set up the variable \$fill.structure in the expressions of the specific link line. \$fill.structure is an array of two elements. The first is the index of the field (that is used as the source field) in the structure and is of type number. The second is the name of the structure and is of type character.</p> <p>When using the fill function on a structured array, you can only modify the specific line of the structure being accessed. Instead of the actual field name, use the index number of the field within the structure. This includes the source field.</p>
\$project.first	true or false	Governs whether a project should be done before moving any of the fields defined in the list of fill to/fill from fields.
\$fill.mult	true or false	Determines whether or not you can fill information from multiple records. True enables fill information from multiple records. The default value is true.

Calling us.link

Use the following parameters to call the us.link application:

Note: Fill and fill.recurse use different parameter names for the same fields.

Parameter Name	Description	Default
record	Source Record (required)	None
name	Field name to find from/link to	Current field
string1	Format Name	Current format
second.record	Link Record (optional)	None
prompt	Action (“find” or “fill”)	“find”
boolean1	Background flag	False
all.null	Exit when closing the current thread	None
index	Cursor line	None
cond.input	Skip Exact Find Note: This parameter is used only for reference fields (that is, fields that have a referenced table defined in the data policy). By default, the Find function of a reference field performs an exact match search instead of using the query defined in the link record. If you want to use the query defined in the link record for a reference field, pass this parameter as true when calling the us.link application.	False

Note: The only required parameter for us.link is the source record.

Skipping query writing

Service Manager performs the “Fill” action based on the query defined in the link line to filter data against the target table. Prior to Service Manager 9.41, you usually defined the query by ID.

As of Service Manager 9.41, a **Skip Query Rewriting** option is available on the Edit Link Line form. This option is added as part of the logical name solution, which adds the **display.name** field to the **device** table to store CI names.

Note: For more information about the logical name solution, see the [Service Manager Logical Name Solution](#) white paper.

Because CI names are now stored in the **display.name** field instead of the **logical.name** field, Service Manager by default automatically rewrites your existing link queries at runtime to address the impact of the logical name solution (the **Skip Query Rewriting** option is disabled by default). If the **Skip Query Rewriting** option is enabled, the system does not perform the additional operations that it does when the option is not enabled.

To access this option, open a link record in Database Manager, highlight a link line and then select the **Select Line** option from the **More** menu. By default, this option is disabled.

Guidelines on editing link queries for a reference field

When writing a link query for a reference field, you need to observe different guidelines depending on whether this option is enabled or disabled.

Note: The following descriptions use **affected.item** as an example reference field.

- If the **Skip Query Rewriting** option is not enabled, update an expression such as “logical.name#affected.item in \$File” to the following:

```
logical.name#\"+affected.item in $File+\\"
```

- If the **Skip Query Rewriting** option is enabled, you must write a query like the following:

```
if (not (null(get.display.value($File, "affected.item")))) then $query+=(" and display.name#\"+get.display.value($File, "affected.item")+\"")
```

Additionally, as a system administrator, you need to understand how the Fill function works for a reference field:

- The “affected.item in \$File” expression will be null when the user types something in the format, and the system gets what the user has typed by the “get.display.value” function.
- If the **Skip Query Rewriting** option is not enabled, the system assigns what the user has typed to “affected.item in \$File” when the user clicks the Fill button, and then rewrites at runtime the

evaluated “\$query” from, for example, “logical.name# DISPLAY VALUE” to “display.name# DISPLAY VALUE”. This makes the original link query still work for the reference field.

Accessing \$File/dates

\$File is used in Fill and Find link line queries. You can modify the \$File directly using the link expressions and the changes are automatically saved in the source record. When performing a fill on a date field, the system first checks to see if that date field is in the link record. If the date field is in the link record, the link expressions perform rather than just fill the current date and time. This way it is possible to fill with something other than tod() (such as date(tod()) or tod() + '7 00:00:00').

Find from and Fill to a \$ variable

The us.link application makes it possible to perform a find from a field that has a \$ variable as an input and to use fill to move information into a \$ variable field. To use a variable as a source field, place the variable name in the Source Field column of a link record. To fill to a specific variable, place that variable on the source field(fill to/post from) column on the specific link line.

Note: Use \$file in virtual join link line queries and \$File in fill and find link line queries.

Posting does not allow use of \$ variables at this time.

Find from and Fill to an array structure

The us.link application makes it possible to perform a find from a field that is part of an array structure and to use fill to move information into that specific element of the array structure. This can only be performed under the following circumstances:

1. The field name to fill from/post to is part of the array structure.
2. The field name to fill from/post to is not used in any other structure of the database dictionary record.
3. The name of the structure must be the same as the name of the array of which it is a part.

To use find/fill on an array structure, you must set up the variable \$fill.structure in the expressions of the specific link line. \$fill.structure is an array of two elements. The first is the index of the field (that is used as the source field) in the structure and is of type number. The second is the name of the structure and is of type character.

When using the fill function on an array structure, you can only modify the specific line of the structure being accessed. Instead of the actual field name, use the index number of the field within the structure. This includes the source field.

The `$fill.display` and `$fill.display.add` functionality

When using `$fill.display`, you can modify the data using the standard rules that apply to that database using Format Control.

- Click **OK** to save the record and use the new version of the record to perform the Fill function, based on the associated link line.
- Click **Cancel** to cancel any updates. It will not fill information back into the source record.

When `$fill.display.add` is set to true and the `us.link` performs the standard link query and returns no records, the form defined in the link line opens and the user can add a record. Any information defined in the Fill to or Fill from fields copies into this record.

- Click **Add** to add the record while leaving the user on this form. At this point the rules for `$fill.display` are followed.
- Click **OK** to add the record and perform the Fill function back to the source record using the newly-added record.

Access the link record

Applies to User Roles:

System Administrator

This examples uses the `problem` table.

Consider the case where there is a record against a user's pc. The incident record requires information regarding the serial number, location of the pc, and its vendor/supplier, as well as a description of the incident.

The required information for documenting an incident is often already in the database. To find and display the information with the incident document, you must tell the system:

- What value to use as a search argument
- Where to look for the matching information

At least two separate records store the required information for the incident document. Therefore, you must define at least two relationships in the link record: one to device, one to vendor/supplier, and one to any other applicable table.

The system stores this information about each configuration item in Configuration Management and references it according to several categories of information. The device record contains the logical name, contact, and location names. The vendor/supplier record contains the vendor/supplier ID, location, and vendor/supplier phone number.

There is an implied order within each link that defines these relationships. The device record contains the name of the vendor/supplier but not the phone number of the vendor/supplier, and the location record stores the contact name instead of the device record.

By defining the first relationship between the incident document and the device record, you can retrieve the data necessary to form the queries that retrieve data through the second and third relationships.

The link record defines the relationship between the incident document and information in the device, vendor/supplier, and other records. Although you can have different link records for each form in Incident Management, the `problem` and `probsummary` records always exist. These records provide a basis and example for other Incident Management link records.

Note: If you are using IR Expert, consider building links from one of the IR fields (like `action`) to itself. This allows a simple Find command to locate relevant incidents before a record opens.

1. Click **Tailoring > Tailoring Tools > Links**.
2. In the Name field, type **problem**, and then click **Search**.
3. Select a record from the list.

The column labeled **Source Field Name** contains the names of fields in the current record (in this case `problem`). The columns under the **Target File Name** contain the file names and corresponding field names that define the relationship. When the link runs, the contents of the source name in the `problem` record search the target table for other information to populate the form.

The configuration items link to the incident document by the field labeled CI Name (`logical.name`) field. The first link to the device record retrieves the serial number and location names. Subsequent links use that information to form their own relationships.

A link to the vendor/supplier record from the field labeled Service Provider (on any linked Incident Management form) provides the vendor/supplier phone number for the incident document.

Configuration Management data stored on Configuration Items (CIs), vendor, and other records can link to several forms in Incident Management in the same way multiple relationships for the same field exist. The link record specifies these link relationships. Links between data and forms are established in

the order in which they appear in the list. A query for a specific field can link to more than one configuration item record in the process of searching for the necessary data.

Virtual joins

Virtual joins allow you to display data fields from several different database files on a single form. Virtual joins do not allow data entry into the joined fields; they only display information from other files into a format for reference purposes.

Virtual joins can be used on any format. They are established when data from a record is displayed without the need to use the Find and Fill functions of the link utility.

You create a virtual join by adding a subform to any form, and displaying the desired data there.

Understanding subforms

Subforms allow the creation of multi-part forms, where sections of the form can be reproduced in other forms. These modular forms enable the simultaneous display of data from numerous database records via virtual joins. Subforms are also useful in constructing multiple views of the same subform data, for example, Incident Management views.

You construct subforms the same way as standard forms, with the exception that you create them to appear within another, larger, form. Subforms can be micro versions of larger forms, containing specific key information that is valuable for display on other forms. For example, Configuration Management subforms that appear on incident record forms.

You can create a subform on any form as long as it is properly placed using Forms Designer. Data from a record can open in a subform automatically or through the use of the Find and Fill function keys of the Link utility.

Example: Creating a virtual join

Applies to User Roles:

System Administrator

The tasks listed below illustrate the steps involved in retrieving data from a file called sales and displaying data in a form called orders.

The forms and subforms that you need to build to create a virtual join include:

- **sales:** You create the main sales form, and then you create a database dictionary file so data can be entered through the form and recorded to the database. The file in the database associated with this form is the target file from which data will be retrieved to display in the orders file.
- **sales1 subform:** When you create the sales1 subform, it is a copied and modified version of the original form (sales). It is already associated with the sales database dictionary record.
- **orders:** When you create the orders form, you will add fields for customer name, contact, phone number, and order amount.

Note: Code must be a single Unique key (not concatenated) in the target file. Also, the source and target fields must be scalar (non-array).

To complete building the necessary forms and creating the virtual join in order to retrieve data from the sales file and display data in the order forms, perform the following tasks:

1. ["Build the sales form" below](#)
2. ["Create the sales file" on the next page](#)
3. ["Create the sales record list" on page 313](#)
4. ["Add data to the sales file" on page 314](#)
5. ["Create the sales1 subform" on page 314](#)
6. ["Create the orders form" on page 315](#)
7. ["Build the virtual join into the form" on page 316](#)
8. ["Build the link" on page 318](#)
9. ["Verify that the sales1 form works" on page 319](#)

Build the sales form

Applies to User Roles:

System Administrator

When you create the main sales form, you need to create a database dictionary file so data can be entered through the form and recorded to the database.

1. Click **Tailoring > Forms Designer**.
2. Type **sales** in the Form field, and then click **New**.
3. Click **No** at the Forms Wizard prompt. A new blank form opens.
4. Using the Label tool, add a title across the top. For example, **Sales Personnel Information**.
5. Create two tabs by selecting the Notebook button. Name the front tab **Name/Code** and the back tab **Manager/Commission** by changing the Caption property.
6. Select the Name/Code tab and do the following:
 - a. Add a Text box and place a Label called **Sales Code**.
 - b. Place a Text box next to the Label with the input value **code**.
 - c. Add a Text box and place a Label called **Seller Name**.
 - d. Place a Text box next to the Label with the input value **name**.
 - e. Add a Combo Box. Place a Label called **Location** and add a list of at least four city names in both the **Displaylist** and **Valuelist** fields.
 - f. Place a Combo Box next to the Label with the input value **commission**.
 - g. Add a Text box and place a Label called **Sales Commission %**.
 - h. Place a Text box next to the Label with the input value **commission**.
7. Select the Manager/Commission tab and do the following:
 - a. Add a Text box and place a Label called **Manager**.
 - b. Place a Text box next to the Label with the input value **manager**.
8. Click **OK** to confirm the new form.
9. Click **OK** again to exit Forms Designer and save your new form.

Create the sales file

Applies to User Roles:

System Administrator

To create the sales file:

1. Log in to the Service Manager Windows client.
2. Click **Tailoring > Forms Designer**.
3. In the Form field, type **sales**, and then click **Search**.
4. Go to the Detail menu to create a file. Either right-click in the sales form or select the Detail menu button in the top right corner, and then select **Create File**.
5. Type **sales** as the filename, and then click **OK**.

If the file name is already prepopulated with "sales," click **OK** to confirm.

This creates a valid "sales" file. The database dictionary file (dbdict record) generates automatically and displays in the System Definition Utility, with the exception of the No Nulls key for the location field. It becomes the target file for your subform when it appears on other forms.

6. Click the Fields and Keys tab to see field definitions and keys. It contains one key on field "code," which is a type "unique."
7. In the Keys section, click **New**.

The General section is displayed.

8. In the Type field of the General section, select **No nulls**, and then click **Add**.
9. Select **location**, and then click **OK**.
10. Click **Save**.

A warning appears, stating that the pending changes in the keys definitions for this dbdict record require a tableregen.

11. Click **OK**, and then close the System Definition Utility window.

The main sales data form and file are now complete.

Create the sales record list

Applies to User Roles:

System Administrator

To create the sales record list:

1. Click **Tailoring > Forms Designer**.
2. In the Form field, type **sales.qbe**, and then click **New**.
3. Click **No** at the Forms Wizard prompt.
4. Draw a table on the design space.
5. Type **Code** and **Location** in the Columns field of the Table properties window. These are the two database dictionary keys for the sales file.
6. Type **code** in the Code column input field.
7. Type **location** in the Location column input field.
8. Click **OK** to finish.

Add data to the sales file

Applies to User Roles:

System Administrator

To add data to the sales file:

1. Click **Tailoring > Forms Designer**.
2. In the **Form** field, type **sales**, and then click **Search**.
3. Click **More** or the More Actions icon and then select **Database Manager**.
4. Create at least two new sales personnel records. Make sure to fill in the fields on both tabs.
5. Click **OK** to return to Forms Designer.

Create the sales1 subform

Applies to User Roles:

System Administrator

When you create the sales1 subform, it is a copied and modified version of the original form (sales). It is already associated with the sales database dictionary record.

To create the sales1 subform:

1. Click **Tailoring > Forms Designer**.
2. In the Form field, type **sales**, and then click **Search**.
3. Click **Copy/Rename** from the More Actions menu.
4. Make a copy of the sales form.
 - a. In the New Name field, type **sales1**.
 - b. Click **Copy** in the Format field.
 - c. Click **OK** to confirm the creation of the duplicate form.
5. Click **Design**.
6. Cut the Seller Name label and field, as well as the Sales Commission % label and field from the notebook tabs, and then paste them somewhere on the form.
7. Remove everything from the form except Seller Name and Sales Commission %.
8. Click **OK** to confirm changes to sales1.

Create the orders form

Applies to User Roles:

System Administrator

To create the orders form:

1. Click **Tailoring > Forms Designer**.
2. In the Form field, type **orders**, and then click **New**.

If you have another form named orders, return to the Form field and rename this new form.
3. Click **No** at the Forms Wizard prompt.
4. Select the Frame tool and do the following:
 - a. Draw a group on the screen and give it a label name **Customer Order**.
 - b. Draw a frame on the screen for the Customer Order fields.

c. Add fields and label the form. The inputs for the text box fields are as follows:

- **customer.name**
- **contact**
- **phone.number**
- **order.amount**

5. Click **OK**.

Build the virtual join into the form

Applies to User Roles:

System Administrator

To build the virtual join into the form:

1. Open the orders form (created in ["Create the orders form" on the previous page](#)) in Forms Designer.
2. Click **Design**.
3. Select the Group tool and Frame tool to create a frame on the screen where you want the seller data to appear. Give it the caption/title **Seller Data**.
4. Select the Label tool and create a label named **Seller Code**.
5. Select the Comfill tool and create a fillable input field. This enables you to choose from a list of seller codes.
6. In the Property box, do the following:
 - a. Type **code** in the input value and **9** (fill) in the ButtonID field.
 - b. Click **Y** to confirm the addition.
7. Click **OK** to save changes to the form.

Subform placement

1. Select the Subform tool.
2. Position the cursor on the screen where you want to begin your virtual join, and then create a square where you want the virtually joined subform to appear.
3. In the Property box for the subform, type **code** in the Input field.

The input field value (code) is the same as the previous input field value, because a link needs to be established with the target file. The Virtual Join tool uses the link with the code key on the sales file to pull in the information requested by the sales1 form, namely values for the name and commission fields.

4. Type **sales1** in the Form field to indicate which form should appear in the subform area.
5. Click **Yes** in the Virtual Join field of the Property box to activate the virtual join functions on the subform.
6. Click **OK** to save changes to the orders form.

Note: You cannot see the subform at this point. The subform data fields do not appear until you enter data through the Database Manager.

7. From the orders form, click **More** or the More Actions icon and then select **Create file**.

The Database Dictionary (dbdict) utility opens.

8. Create an orders file in the database dictionary.
 - a. Type **orders** as the filename, and then click **OK**.

If the name is prepopulated with orders, click **OK** to confirm.

This creates a valid "orders" file. The dbdict record for it is now displayed in the System Definition Utility. In the Fields and Keys tab, you can look at field definitions and keys. It contains one key on field "customer.name," which is type "unique."

- b. In the orders file, ensure there is a line for the code field at the bottom of the list. If there is not, in the Fields section click **New Field**, and then create a new character field called "code."
 - c. Click **Save** to save your changes.
9. Close the System Navigator to leave the System Navigator utility and return to Forms Designer.

Build the link

Applies to User Roles:

System Administrator

To build the link record:

1. Click **Tailoring > Forms Designer**.
2. Type **orders** in the Form field, and then click **Search**.

The orders form opens.

3. Click **More** or the More Actions icon and then select **Link**.

The link record for the orders file opens.

4. Create a link for the code field. Link the (Seller Code) code field on the order file to the (Seller Code) code field on the sales file.
5. Click **Save**.
6. Click **Back** to exit the link record and return to Forms Designer.

Note: Use \$file in virtual join link line queries and \$File in fill and find link line queries.

Use the virtual join

Applies to User Roles:

System Administrator

Once you save the record, you see the virtual join information whenever you select, add, or update the record.

To create a new orders record using the orders form.

1. Click **Tailoring > Database Manager**.
2. Type **orders** in the Form field, and then click **Search**.

The "orders" form opens.

3. Type the name of a customer in the **Customer Name** field.

4. Type a contact for that customer in the **Contact** field.
5. Type a number for the **Phone** field.
6. Type any number for the **Order Amount** value.
7. In the **Seller Code** field, select a seller code from the record list of seller codes.
8. Click **Add** to save the new orders record.

The new orders record appears with the sales file data virtually joined in the subform at the bottom.

The Seller name and commission fields are not stored with the record in the order file. They are only referenced and are noneditable. To make changes or updates to the seller data, use the sales form.

Note: If you delete this record, Seller name and commission do not delete because they are stored in the sales file.

Verify that the sales1 form works

Applies to User Roles:

System Administrator

To verify that the sales1 form works:

1. Click **Tailoring > Database Manager**.
2. Open the sales1 form.
3. Click **Search** and select a record from the record list.

The first *sales* form inputs sales personnel data that is retrieved through the *sales1* subform. The *sales1* subform is virtually joined to the orders form.

Entity Relationship Diagram creation utility

An Entity Relationship Diagram (ERD) is a graphic description of the logical structure of a database. Database administrators often rely on conceptual and physical data models to understand the relationships among tables, records, and data, and to produce reports.

The Entity Relationship Diagram (ERD) Creation utility enables you to generate Database Definition Language (DDL) statements that you can import into any tool that is capable of producing this type of

conceptual data model. For example, Erwin Data Modeler can create a visual Service Manager data model when you import a DDL file created by the ERD Creation utility. If you have administrative rights, you can customize the files and links used to create the component DDL statements.

Checklist: Creating an ERD record for an application

Ensure that you consider the following items prior to creating an ERD record for an application.

1. Identify all files that are related to the application name.
2. Identify the master link records for all files listed on the Files tab. Master link records are the link records with the same name as the application files.
3. Identify all link records that are associated with any file listed. The ERD Creation utility associates these link records by the format name.

What is the required input for ERD Create?

The ERD Create utility uses a variety of input to generate DDL statements. The following list describes the basic input requirements.

- The list of files specified in the erdcreate record and all fields in these files.
- The relationships among these files determined by the link records specified in the erdcreate record.
- If specified, the relationships among the files specified by joindef and erddef records.
- If specified, the relationships among the files specified by the Match Fields section of Data Policy.
- If specified, the relationships manually defined by the database administrator.

Create an ERD definition

Applies to User Roles:

System Administrator

An Entity Relationship Diagram (ERD) is a graphic description of the logical structure of a database. Database administrators often rely on conceptual and physical data models to understand the relationships among tables, records, and data, and to produce reports.

To create an ERD definition:

1. From the System Navigator, click **Tailoring > Database Manager**.
2. In the Form field, type **erdddef** and click **Search**.
3. From the Associated Forms section, double-click the **erdddef.g** form.
4. Enter the **First Filename**, **Second Filename**, and **Relationship Type**.
5. In the **Field Names from First Filename** text box, enter the fields for which you want to create the join.
6. Enter the fields for which you want to create the join in the **Field Names from Second Filename** text box.
In this example, the contact.name field in the probsummary table contains the same data as the contact.name field in the contacts table.
7. Click **Add**.

For additional information about relationship types, see **Create a join** in the related topics.

Create an ERD record

Applies to User Roles:

System Administrator

An Entity Relationship Diagram (ERD) is a graphic description of the logical structure of a database. Database administrators often rely on conceptual and physical data models to understand the relationships among tables, records, and data, and to produce reports.

To create an ERD record:

1. Click **Tailoring > Tailoring Tools > ERD Create Records**.
2. Type the **Name** (usually the application name) for the ERD Create record.
3. Click **More** or the More Actions icon and then select **Add Files for Module**.

Note: The Add Files process will fail if the name of the ERD record is not associated with one or more Data Policy records.

4. Click **More** or the More Actions icon and then select Add Master Links.

5. Click **More** or the More Actions icon and then select Add all Related Links.
6. Click **Add**.
7. Click **OK**.

Note: You can add relationships manually from the Manual Relationships tab. Use this tab to define relationships that do not occur in any of the automatically discovered categories. Each relationship that you create manually also appears in the ERD.

Modify an ERD record

Applies to User Roles:

System Administrator

An Entity Relationship Diagram (ERD) is a graphic description of the logical structure of a database. Database administrators often rely on conceptual and physical data models to understand the relationships among tables, records, and data, and to produce reports.

To modify an ERD record using the Entity Relationship Diagram (ERD) Creation utility:

1. Click **Tailoring > Tailoring Tools > ERD Create Records**.
2. To locate an existing ERD record, click **Search**.
3. Select a record from the record list.
4. Click **More** or the More Actions icon, and choose **Add Files for Module**.
5. From the More Actions menu, choose **Add Master Links**.
6. From the More Actions menu, and choose **Add all Related Links**.
7. Click **Save**.
8. Click **OK**.

Notes:

- You can add relationships manually from the Manual Relationships tab. Use this tab to define relationships that do not occur in any of the automatically discovered categories. Each relationship that you create manually also appears in the ERD record.

- The ERD record creation process will fail if the name of the ERD record is not associated with one or more Data Policy records.

Create manual relationships

Applies to User Roles:

System Administrator

An Entity Relationship Diagram (ERD) is a graphic description of the logical structure of a database. Database administrators often rely on conceptual and physical data models to understand the relationships among tables, records, and data, and to produce reports.

To create manual relationships in the ERD definition:

1. Click **Tailoring > Tailoring Tools > ERD Create Records**.
2. Click **Search** to display a list of existing records.
3. Select an application name, and then select the **Manual Relationships** tab.
Note: The Manual Relationships tab enables you to define any relationship that is not automatically detected by Service Manager through link, erddef, and datadict records. You must identify the two files that are related and the specific fields that create the relationship.
4. Type a user-defined unique **Name** for the relationship.
5. Type the name of a source file where the relationship originates in the **From File** text box.
6. Type the name of a field in this file in the **From Field** text box.
7. Type the name of a target file in the **To File** text box.
8. Type the name of a field in this file in the **To Field** text box.
9. Click **Save**.
10. Click **OK**.

What are Master link records?

Master link records are link records that have the same name as the associated file. For example, the Master link record for the contacts file is the contacts link record.

What are related link records?

Related link records are form records that relate to the associated files and link records that relate to these form records. The ERD Create utility collects related link records by finding all forms associated with the selected files, and then locating any link records associated with these forms.

Associate a data policy record with an application

Applies to User Roles:

System Administrator

Every data policy record does not automatically have associated applications. You may need to complete Step 4 to ensure there is a link between the data policy record and the appropriate application. When the ERD Creation Utility runs for a named application, the utility looks for all data policy records with links to that application to generate the list of associated files.

To associate a data policy record with an application:

1. Click **Tailoring > Data Policy**.
2. Click **Search** to generate a list of existing data policy records.
3. When you select a data policy record from the list, the fields in this record appear in the Data Policy form.
4. The General tab shows the current applications associated with the data policy record. To add another application, insert the cursor in a blank **Modules** row. Click the drop-down list to choose a Service Manager application.
5. Click **Save**.
6. Click **OK**.

Checklist: Generating a DDL file

Ensure that you consider the following items prior to generating a DDL file for an application.

1. Ensure that there is one or more data policy record associations for the application.
2. Create a new ERD record that lists all related files and fields.
3. Define any additional manual relationships.
4. Use the ERD Create utility to generate a DDL file that contains DDL statements.
5. Open the DDL file to import the DDL statements into a database modeling tool.

DDL mode

The DDL mode describes how the ERD Create utility creates foreign key definitions. You can select from one of two modes, Full mode or Simple mode.

Full mode

Full mode creates one foreign key definition for each unique field or file relationship. In Full mode, you can have multiple links between two files.

Simple mode

Simple mode creates one foreign key definition for each file-to-file relationship within each relationship type.

Generate a DDL file

Applies to User Roles:

System Administrator

Database administrators often rely on conceptual and physical data models to understand the relationships among tables, records, and data, and to produce reports. If you have administrative rights, you can customize the files and links used to create the component Database Definition Language (DDL) statements.

To generate DDL statements in a file using the Entity Relationship Diagram (ERD) Creation utility:

1. Click **Tailoring > Tailoring Tools > ERD Create Records**.
2. Determine the correct export mode for the DDL statements.
 - **Full mode:** Creates one foreign key definition for each unique field or file relationship. You can have multiple links between two files.

- **Simple mode:** Creates one foreign key definition for each file-to-file relationship within each relationship type.
3. From the ERD Creator Definition form, click the More Actions menu.
 4. Select one of the following:
 - **Create/Export DDL > Full.**
 - **Create/Export DDL > Simple.**

The output of the ERD Create utility automatically resides at the root level of the Service Manager client installation folder. The file name is the DDL file name with a .sql file extension.

Example: ..\Program Files\HP\ Service Manager x.xx\ Service Manager Client\ddl_name.sql.

Import a DDL file into a database modeling tool

To import a DDL file into a database modeling tool, this example uses Erwin Data Modeler. Other tools may have different steps.

1. From the Erwin Data Modeler, click **File > Open**.
2. Change the file type to **SQL DDL**.
3. Browse to locate the DDL file.

By default, Service Manager stores these files at the root level of the client installation folder. For example: ..\Program Files\HP\Service Manager x.xx\Client\.

4. Click the file to select, and then click **Open**.

The Erwin Data Modeler displays the Reverse Engineer - Select Template wizard.

5. Modify the settings in the Target Database section, as follows:
 - Select the appropriate Database.
 - Select the appropriate Version.
6. Click **Next**.
7. Select the appropriate values.
8. Click **Next**.

The resulting Entity Relationship Diagram opens. You can use the Zoom tool to view the content of each entity in the diagram.

Posting

Posting is the process of copying data from a source record to a target record. The purpose of posting is to update similar fields in other records without having to open those records to modify each field. The relationship between the two records is normally based on a value defined in one (or more) of the source record input fields. Posting is the opposite of the Fill function, which copies data from the target record to the source record. You can use posting to update Service Manager tables from another part of Service Manager.

Posting does the following:

- Starts on a source record.
- Accesses a record in a secondary (target) table, based on data in that record.
- Copies (posts) data from the source to the target, and then performs some action (for example, add, update, or open) on the target record.

The posting process enables you to control the process in several ways:

- Define the target table.
- Select the fields to copy.
- Determine whether to prompt the user for add/update confirmation.
- Decide which application to invoke for the posting process.
- Define manual or automatic posting.

Note: Do not post to the probsummary table or to incident records. The record being updated may be locked by another user and the post will not occur. In addition, the pm.eval.alerts application will create its own \$file variable, which conflicts with the \$file variable that post.fc is using. Updating the probsummary table triggers an application that cleans out the \$file variable, the same variable that is used by the Format Control routines.

Posting link records

The posting process requires you to create a standard Service Manager link record. The link record establishes the connection between similar fields in the source and target files. When posting, it is best to create a new link record dedicated to the posting process, to avoid conflicts with link records defining common fill relationships.

To refer to fields in the source record, use the `$File` file variable. To refer to fields in the target record, use the `$Filet` file variable.

Important: The case must match the example, with the initial 'F' upper case and the rest lower case.

Posting link line definition file

The link line definition file defines a single field as the reference field for the posting routine and lists the names of other fields linked for the posting process. When posting is invoked, the system compares the value for the reference field in the source file with the value for the linked field in the target file. If the values are the same, the system compares the listed fields for changes and posts the new data to the target file.

The fields used for the link line definition have the same definition or use for posting as they do for find and fill with the following exceptions:

- The source field array lists fields in the source record and the target field array lists fields in the target record.
- You can define parameters in the Expressions field to control the posting process flow.

Note: Reference or manipulate input fields with the `$File` file variable. Any modifications made to this record variable are passed back to the calling application.

Important: Processing statements may modify the contents of the original record.

Types of posting available in Format Control

There are two types of posting available in Format Control:

- Automatic posting, set up via Format Control subroutines.
- User-controlled posting, set up via the Format Control additional options for use with Database Manager only.

Confirming the posting routine

A confirmation function is available in Posting that posts the new contact information to the target file, but waits for user approval before updating the record. The target file is open to the user, who can edit any of the information before confirming the update. For example, when the Posting routine from Change Management creates a new Contact record, only the fields specified in the Link record populate. At some point, the record must be complete. When the confirmation feature is active, the new Contact record opens to the user when Posting occurs. The user can complete the contact information form, save the record, and then return to the request without leaving Change Management.

Note: This feature is available in both the automatic and the manual Posting modes.

Posting variables

The following variables are available during the posting process and can be used to control the Posting process flow:

\$File

The source record file variable.

\$Filet

The target record file variable.

\$post.msg

The message issued to the operator when the posting routine successfully completes. The default message is: The posting routine is complete.

\$post.confirm

A logical field that controls whether or not the user is prompted to confirm the Posting action. The default is false.

\$post.confirm.fmt

The name of the form used to display the target record for Posting confirmation.

\$post.appl

The name of the RAD routine used for posting. If an application is not defined to this variable, and the

`$post.confirm` option is true, the confirmation is a simple add or update of the target record. No other processing is performed against the record.

`$post.names`

An array that defines the names of the parameter panel input fields to which data will be passed. This field is needed only when an application is defined to the `$post.appl` parameter.

`$post.values`

An array that defines the values passed to the called application. There is a one-to-one relationship between this array and the `$post.names` array.

Automatic updates using Format Control subroutines

You can use posting to update Service Manager tables from any other part of Service Manager. For example, new contact information from a request can post directly to related fields in the contacts file without any user interaction. If you are opening a request for a new contact, posting creates a new Contact record and populates the common fields defined in the Link record. Posting from Request Management can occur when new requests are opened.

In this example, `ocmq.open.newemployee.quote.g` is our Source Format (accessing the `ocmq` file) and `contacts` is our Target form (accessing the `contacts` file). Both files contain such contact information as contact name, company, first name, last name, and department. The relationship between the two files is based on a single common field—the `employee.id` field in this example (Employee Number). When Posting initiates from a request, the system attempts to find a Contact record in which the value of the `user.id` field matches the value of the `employee.id` field in the request. If a match is found, the system updates the Contact record. If no match is found (as in the case of a request from a new person), the system creates a new Contact record based on the values in the request.

Posting should NOT be performed on the `probsummary` table when set up using a subroutine in Format Control as this causes the system to clear the variable, `$file`.

Use the procedures in the following help server topics to create automatic updates using Format Control subroutines:

1. [Identify the field input values for posting](#)
2. [Create the link record for posting](#)
3. [Create the Format Control record for automatic posting](#)
4. [Open a request for posting](#)

Identify field input values for posting

Applies to User Roles:

System Administrator

To create a Link record for Posting, you must identify the common fields in the Source and Target files. This can be accomplished from within Forms Designer. For this example, individual fields are selected in the design mode.

To identify common fields in the Source and Target files:

1. In Forms Designer open the form `ocmq.open.newemployee.quote`.
2. Click **Design**.
3. Select a field and check the Input value in the Property window.

Note: This method displays all input values, including those not shown in Fill boxes.

Important: Be certain not to alter the form when checking field input values. Click **Cancel** to exit the Design mode without changing the form.

Create the link record for posting

Applies to User Roles:

System Administrator

Create a new Link record for the form you are going to use. For example, if a Link record does not already exist, create the Link record `ocmq.post.contacts` for the new form `ocmq.open.newemployee.quote` for the Posting process. This avoids any conflicts involving links established in a current record for the Fill function.

1. Click **Tailoring > Tailoring Tools > Links**.
2. In the Name field, type the name of the Link record you want to create. For this example, type **ocmq.post.contacts**.
3. Click **New**.
4. Type the following values in the first line of the new Link record.

Field	Value
Field Name (SOURCE)	employee.id
Format Name (TARGET)	contacts
Field Name (TARGET)	user.id
Comment	POST

Important: You must type POST in upper case letters in the Comment field for the Posting process to work.

The new Link record opens.

- Place the cursor in the employee.id line and click **Select Line** from the More Actions menu.

The Link line definition form opens.

- Type the following values for the Source and Target fields.

Source Field	Target Field
employee.id	user.id
requested.for	contact.name
employee.id	full.name
request.phone	contact.phone
request.dept	corp.structure

- Click **Back**. A prompt asks if you want to save the Link record you have created.

- Click **Yes** to save the record.

Create the Format Control record for automatic posting

The Subroutines process of Format Control defines Automatic Posting. The application called is post.fc. Posting takes place before the request is added or updated in the database.

Important: As posting may add a new record to the target file, you must fill in all fields that are part of a Unique or No Nulls key using the Source Field / Target Field table, otherwise posting fails.

1. Click **Tailoring > Format Control**.
2. In the **Name** field of the Format Control Initializations form, type **ocmq.open.newemployee.quote**.
3. Click **Search**.
4. If no record exists, click **New**.

The Format Control record for ocmq.open.newemployee.quote opens.

5. Click **More** or the More Actions icon and then select Subroutines. You can also click **Subroutines** on the form.
6. Click **More** or the More Actions icon and then select Show Expanded Form.

The long version of the Subroutines form opens.

7. Type the following values in the first available slot.

Names	Values
file	\$file
name	ocmq.post.contacts
prompt	emoloyee.id
Error Message	Could not post to contacts file.
Application Name	post.fc
Comments	POST
Add	true
Before	true

Note: The name parameter passes the name of the Link record you created, and the prompt parameter passes the source field for the Link relationship.

8. Click **Back**. A prompt asks if you want to save the changes to the Format Control record.
9. Click **OK** to save the record.

Open a request for posting

Applies to User Roles:

System Administrator

To test the posting routine you have set up, you must create a new request. We will update the contacts file by changing the values in the First Name and SM Contact Name fields of the request, and then add a title.

1. Click **Request Management > Quotes > Create New Quote**.
2. Select **Human Resources > New Employee Setup**.
3. A summary form of the items you requested displays with total cost. You can choose to:
 - Cancel your request.
 - Add Items to your request.
 - Submit Request.
4. Click **Submit Request**. The Human Resources - New Employee Setup Request form opens.
5. Place the cursor in the **SM Contact Name** field of the Employee Information section, and then click **Fill**.
6. Place the cursor in the **Contact Name** field, and click **Search**. A record list of contacts opens. The contact information comes from the contacts file.
7. Select a name from the record list of contacts.

Service Manager populates the Employee Information section with information from the contacts file.

8. Change the **First Name** and **SM Contact Name** in the Human Resources - New Employee Setup Request form.
9. Add a **Title**.
10. Fill in the required fields.
11. Copy the **Employee Number** to use it in the contacts file later.

12. Click **Save & Exit**.
13. If errors occur, read the posting messages in the Messages view.

To verify changes made to the contacts file during the posting process.

1. Click **System Administration > Base System Configuration > Contacts**.
2. Paste the Employee Number copied earlier into the **Employee ID** field, and then click **Search**.

The Contact Information form opens. You can see the contact information you updated in the request, along with the title you added for the employee.

Manual posting using Format Control additional options

In some cases, you may want to give users the option to manually control posting. The additional options process of Format Control defines user options for posting data and require user interaction to complete the routine.

Manual posting is desirable whenever you do not want changes to post automatically. You might want to do this to avoid posting temporary changes to permanent records. For example, if the contact is calling from a different phone, you could add the temporary number to the change request without posting to it in the contacts record.

Additional options are only available for Database Manager. To set up manual posting within other Service Manager applications, set up a new display option definition to call the post.fc application.

In this example, we will reuse our posting link record from Database Manager and define additional options for a hardware change request form. The user will be able to click on a button to post updates to contact information. As in the previous example, the source file is cm3r, and the target file is contacts.

Note: HP does not recommend accessing Change Management forms and records using Database Manager. We do so in this example only to demonstrate additional options capabilities for manual posting.

Perform the steps in the following help server topics to manually control posting using Format Control:

1. [Create the link record for posting](#)
2. [Create the Format Control record for manual posting](#)
3. [Modify a hardware change request for posting](#)

Create the link record for posting

Applies to User Roles:

System Administrator

Create a new Link record for the form you are going to use. For example, if a Link record does not already exist, create the Link record `ocmq.post.contacts` for the new form `ocmq.open.newemployee.quote` for the Posting process. This avoids any conflicts involving links established in a current record for the Fill function.

1. Click **Tailoring > Tailoring Tools > Links**.
2. In the Name field, type the name of the Link record you want to create. For this example, type **ocmq.post.contacts**.
3. Click **New**.
4. Type the following values in the first line of the new Link record.

Field	Value
Field Name (SOURCE)	employee.id
Format Name (TARGET)	contacts
Field Name (TARGET)	user.id
Comment	POST

Important: You must type POST in upper case letters in the Comment field for the Posting process to work.

The new Link record opens.

5. Place the cursor in the `employee.id` line and click **Select Line** from the More Actions menu.

The Link line definition form opens.

6. Type the following values for the Source and Target fields.

Source Field	Target Field
employee.id	user.id
requested.for	contact.name
employee.id	full.name
request.phone	contact.phone
request.dept	corp.structure

7. Click **Back**. A prompt asks if you want to save the Link record you have created.
8. Click **Yes** to save the record.

Create the Format Control record for manual posting

Applies to User Roles:

System Administrator

Manual posting is user-controlled posting, set up via Format Control Additional Options, for use with Database Manager.

1. Click **Tailoring > Format Control**.
2. In the **Name** field of the blank Format Control record, type the name of the form. For this example, type cm3r.hardware. Click **Search**

Note: If the Format Control record does not already exist, click **New** to create a new Format Control record.

3. Click **More** or the More Actions menu, and choose **Additional Options**. You can also click **Addl Options** in the form.
4. Click **More** or the More Actions menu, and then choose **Show Expanded Form**.

The long version of the Additional Options form opens.

5. Type the following field values in the first available slot.

Field	Value
Option	1
Condition for Option	true
Description	Post to Contacts
Description ID	
Comments	POST
Application	post.fc
Error Message	Could not post to contacts file.
Reset on Return	true

Names	Values
file	\$file
name	cm3r.hardware.post
prompt	requested.by

6. After entering the second parameter name and value (name/cm3r.hardware.post), click **Save** to save the updated Format Control record.

A scroll bar appears next to the Names array, so you can scroll down and enter the third Names value (prompt).

7. To enter the third Values value (requested.by), position the cursor on the first Value entry(\$file) and press the Page Down key.
8. Click **OK** to save your changes.

Modify a hardware change request for posting

Applies to User Roles:

System Administrator

To test the manual Posting routine you set up, you must access a hardware change request and modify one or more fields in the Initiator section that was defined in the link record. Then, you will invoke the Additional Options to verify that the contacts file data changes do reflect the changes in your change request.

We will do this in the Database Manager Administrator mode, because Format Control Additional Options are not available in Change Management.

1. Click **Tailoring > Database Manager**.
2. Select the **Administration Mode** checkbox.

Note: You must click Administration Mode or you will be using Change Management instead of Database Manager. Format Control Additional Options are not available in Change Management.

3. Open the **cm3r.hardware** form.
4. Click **Search**.

A list of change request records opens.

5. Retrieve any existing Hardware change request.
6. Verify that the existing Initiated By information matches an existing contacts file record. To do so, position the cursor on the **Name** field, and then click **Find**.

The related Contact Information record opens.

7. If no contacts record match exists, do the following:
 - a. Clear the existing **Name** field, and then use the Fill button to fill a valid contact value into the change request record.
 - b. Update the **Department** and **Phone** fields.
 - c. Click **Post to Contacts** from the More Actions menu to initiate posting to the contacts file.

Note: The option name depends on the value entered in the Command field of the Additional Options definition. Therefore, you may see something other than Post to Contacts in your More Actions menu.

- d. Place the cursor in the **Change Initiator Name** field, and then click **Find**. The updated Contact record opens.

The updated departmental data from the change request opens in the Corp. Structure field at the bottom of the Business Information tab, and the updated phone data opens in the Work phone field in the Contact Numbers tab.

8. Click **Save**.

Use the confirmation function in the posting routine

Applies to User Roles:

System Administrator

Activate the confirmation function by setting the flag `$post.confirm` to true in the Expressions field of the source file Link line definition record.

1. Click **Tailoring > Tailoring Tools > Links**.
2. Type the name of the source file Link record in the **Name** field of the Link Manager dialogue box. In this example, use the name of the Link record you created, `cm3r.hardware.post`.
3. Click **Search**. The source file Link record opens.
4. Place the cursor in the `requested.by` field of the Link record, and then click **Select Line** from the More Actions menu. The Link line definition form opens.
5. Type `$post.confirm=true` in the Expressions field.
6. Click **Back**. A prompt box opens, asking if you want to save the changes to the Link record.
7. Click **Yes**.

Do manual posting with confirmation

Applies to User Roles:

System Administrator

Use the same procedures for evoking the confirmation process as for establishing User Options in manual posting. When you activate the Posting routine, the confirmation process opens the target form and file (contacts) and provides a new set of toolbar buttons.

1. Click **Tailoring > Database Manager**, and then select the **Administration Mode** checkbox.

Note: You must click Administration Mode or you will be using Change Management instead of Database Manager. Format Control Additional Options are not available in Change Management.

2. Open the `cm3r.hardware` form.
3. Retrieve any existing Hardware change request.

4. Verify that the existing Change Initiator matches an existing contacts file record. To do so, position the cursor on the Change Initiator Name field, and then click **Find**. The related Contact Information record opens on form contacts.g.
5. If no contacts record match exists, clear the existing Change Initiator Name field, and then use the Fill button to fill a valid contact value into the change request record.
6. Update the Department and Phone fields.
7. Click **Post to Contacts** from the More Actions menu to initiate posting to the contacts file.

Note: The option name depends on the value entered in the Command field of the Additional Options definition. Therefore, you may see something other than Post to Contacts in your More Actions menu.

8. The Contact record (target file) for the contact specified in the change request form opens with a confirmation request.

Edit fields other than those selected for the Posting routine.

9. Click **Save** to save ALL changes, from both the change request and also changes entered directly in the contact record, to the database.
10. Click **Cancel** to ignore any changes you made in the record. The Posting process is NOT complete and you return to the change request form.

Display functions

The system permits control of user interaction with the following utilities:

- **Menus** — enable you to control the functionality of the options that appear on the system navigator as well as the buttons that appear on the user's menu form. Different users access different menu forms based on different roles.
- **Display application** — enables you to control the functionality of buttons in the editor toolbar, on forms, and on items in the pull-down options menu.

Menus

The menu of utilities and applications available to an individual user are based on the user's role in the organization.

You can create and customize menus to restrict user access.

A menu form:

- Opens when you log on to the system
- Controls the user's access to different areas of the system
- Is associated with a corresponding menu record
- Opens based on the user's role in an organization

Menu record

Each option line in the menu record contains values that control the functionality of each link on the menu form. Based on the conditions specified, the menu record can call a RAD application to retrieve other menus, call stored queries, and launch external applications. The association between the form and the menu record is set up by entering the form name in the Format field of the menu record.

The numbers in the Option column correspond to the option on the referenced format. In general, there is a one-to-one relationship between menu and form. Each option calls a RAD application. Some RAD applications require parameters to be passed into the function while others do not.

The Command column permits you to call a menu option directly from the command line. For example, type `db` in the command line to go directly to the Database Manager.

The Condition column on the menu record controls the availability of the option. If the condition evaluates to false, the option displayed on the user's menu is unavailable.

Button properties

You can create menu buttons using the Button control in Forms Designer. When adding a button in Forms Designer, the following properties are important:

- **ButtonID** – A numeric value that matches an option number in the related menu record. (Required)
- **Images** – An image that represents the button's function. (Optional)
- **Balloon help** – A text string that appears when the cursor is held over the button to offer more information about the button's function. (Optional)

Access menu records

Applies to User Roles:

System Administrator

After you create a form, you must add a related menu record to specify the functionality of the buttons. Menu records are saved in the menu file. You can access menu records in the following ways:

- Click **Tailoring > Tailoring Tools > Menus**.
- In the Service Manager command line, type `menu`.

Example: Adding a URL link to a menu

Applies to User Roles:

System Administrator

Objective: Provide an example of how to add a link that navigates to a URL from the System Navigator tree or from a Service Manager menu form.

In this example, you need to find the menu file, modify the HOME menu record, and use Forms Designer to modify the `menu.gui.home` form.

To add a link that navigates to a URL from the System Navigator tree or from a menu form:

1. Click **Tailoring > Web Services > Database Manager**.
The Database Manager search window (format.prompt.db.g) opens .
2. Type menu in the File text box and click **Search**.
A record list of the files opens (format.qbe.g).
3. From the record list of menu files, double-click menu in the File Name column.
The Menu form opens (menu.g).
4. Type HOME in the Menu Name field and click **Search** from the More Actions menu.
The Menu form re-displays with a list of menu records for the HOME menu.
5. In the table, add a new menu entry with the following information:
 - o Option #: 48
 - o Description: Hewlett-Packard (The label to give the URL link.)
 - o Application: us.launch.external
 - o Parameter: name
 - o Parameter value: http://www.hp.com
 - o Thread: true
 - o Condition: true
6. From the More Actions menu, click **Save**.
Next you will use Forms Designer to add the menu item as a button on the form:f menu.gui.home.
7. Click **Tailoring > Tailoring Tools > Forms Designer**. Type fd in the Service Manager command line.
The Forms Designer search form opens.
8. Type menu . gui . home in the Form text box and click **Search**.
The menu.gui.home form opens in Forms Designer.
9. Click **Design**. The form opens in the edit window with the Design Tools and Properties panes open.
10. Drag-and-drop a button control from the Design Tools pane to the form.

11. Position the button control on the form. In this example, move the Exit Service Manager button, Service Manager Internal Mail button, and System Status button left to make room for the button you are adding.
12. Click the new button to display the sizing nodes on the button and then resize the button to match the other buttons.
13. In the Properties pane, update the following parameters:
 - Image File: Hewlett-Packard
 - Button ID: Type the Option # (48) you entered for this record in the HOME file.
 - Caption: Hewlett-Packard
 - Name: Hewlett-Packard
14. Click **OK**.
15. To verify the change, logout and then login and view the form you modified.
You should see the button you added on the menu.gui.home form. There should also be a new entry in the Menu navigator folder in the System Navigator.

Customize System Navigator menu icons

Applies to User Roles:

System Administrator

HP Service Manager provides out-of-box icons that you can use to customize the icons in the System Navigator. To do this, you only need to specify the name of the icon in the **Icon** field of the corresponding menu record. You can perform this task from either the web client or the windows client. The customized icons apply to both the accordion mode and the sidebar mode of the System Navigator.

Note:

- You can only specify an icon for a first-level menu item (menu group).
- The System Navigator menu icons are available only when the **enableSidebarMenu** parameter in the web.xml file is configured as TRUE. For more information about the **enableSidebarMenu** parameter, see [Web parameter: enableSidebarMenu](#).

For a complete list of the icons and their names, see [Service Manager menu icons](#).

For more information about how to access the menu record, see ["Access menu records" on page 343](#).

Adding and changing the image icon for menu and toolbar items

You can edit or replace the out-of-box icon for existing menu items and add icons for new menu and toolbar items that you create. The More Actions menu and toolbar icons are 16x16 .gif files. You can use any graphics design tool to edit out-of-box icons or create new icons of the correct size.

When you create a menu or toolbar item, Service Manager generates an icon name for it based on the **Default Label** you give the menu or toolbar item in the Display Application Option Definition form.

Service Manager displays the image with that name if it is located in the Service Manager Client images directory. If there is no image with that name, the form displays the default icon. The default icon is blank (no icon defined) in the out-of-box system.

In a typical installation the icons are located in the following locations.

- Windows client:

```
... \plugins\com.hp.ov.sm.client.common_x.xx\src\resources\icons\obj16
```

- Web client on Tomcat (for example):

```
... \<Tomcat>\webapps\webtier-x.xx\images\obj16
```

Add or change the image icon for options menu and toolbar items

Applies to User Roles:

System Administrator

You can edit or replace the out-of-box icon for existing menu items and add icons for the new menu and toolbar items that you create. The More Actions menu and toolbar icons are 16x16 .gif files. You can use any graphics design tool to edit out-of-box icons or create new icons of the correct size. These images are found in many forms.



To add or change the image icon for the options menu and toolbar items:



















Important: If you change an image, Service Manager will replace the original image with the new image on all forms. If you do not want to change the image on all forms, you must change the **Default Label** in the Display Application Option Definition for the form you want to change, and create a new image.





1. Open the Display Application Option Definition for the More Actions menu or toolbar item.
2. Add the item, if necessary.
3. Make a note of text in the **Default Label**. Using Database Manager, open a form that the option is displayed on.
4. Open the Administrator view and go to the **Detail Data tab** for a single record form or the **List Data tab** for a record list. The list of icons in the toolbar and options menu is at the bottom of the form. The `img="<bmpname>"` tag gives you the name of the image.
5. Make a note of the icon image name.
6. Using your standard operating system search functionality, search your Service Manager directory for a .gif file with that name. If the item with that name exists in the Service Manager images directory, the form will display it.
 - `<installation location>\plugins\com.hp.ov.sm.client.eclipse.web_x.xx\webtier\images`
 - `<installation location>\plugins\com.hp.ov.sm.client.common_x.xx\src\resources\icons`
 - `<installation location>\plugins\com.hp.ov.sm.client.eclipse.admin_x.xx\src\resources\icons`
 - `<installation location>\plugins\com.hp.ov.sm.client.eclipse.user_x.xx\src\resources\icons`
7. Create a new image if none exists, or replace this existing image with the one you prefer to use.

Common option icon names

When looking for the image to display, Service Manager replaces spaces in the image tag with `_` in the filename.

Detail/List Data entry	Windows Client		Web Client	
	Default Icon	File name	Default Icon	File Name
<pre><option value="1" img="tadd"> <caption>Add</caption> <description>Add</description></pre>		tadd.gif		tadd.png

Detail/List Data entry	Windows Client		Web Client	
	Default Icon	File name	Default Icon	File Name
</option>				
<option value="11" img="tprev"> <caption>Previous</caption> <description>Previous</description> </option>		tprev.gif		tprev.png
<option value="11" img="tnext"> <caption>Next</caption> <description>Next</description> </option>		tnext.gif		tnext.png
<option value="2" img="tok"> <caption>OK</caption> <description>OK</description> </option>		tok.gif		tok.png
<option value="205" img="tprint"> <caption>Print</caption> <description>Print</description> </option>		tprint.gif		tprint.png
<option value="211" img="texport/"> <caption>Export/Unload</caption> <description>Export/Unload</description> </option>		texport.gif		texport.png
<option value="212" img="tir quer"> <caption>IR Query</caption> <description>IR Query</description> </option>		tir_quer.gif		tir_quer.png
<option value="3" img="tcancel"> <caption>Cancel</caption> <description>Done Searching</description> </option>		tcancel.gif		tcancel.png
<option value="300" img="tcreate"> <caption>Create Operator</caption> <description>Create Operator</description> </option>		tcreate.gif		tcreate.png
<option value="4" img="tsave"> <caption>Save</caption>		tsave.gif		tsave.png

Detail/List Data entry	Windows Client		Web Client	
	Default Icon	File name	Default Icon	File Name
<description>Save</description> </option>				
<option value="5" img="tdelete"> <caption>Delete</caption> <description>Delete</description> </option>		tdelete.gif	✕	tdelete.png
<option value="5000" img="tmass ad"> <caption>Mass Add</caption> <description>Mass Add</description> </option>		tmass_ad.gif		tmass_ad.png
<option value="5001" img="tmass up"> <caption>Mass Update</caption> <description>Mass Update</description> </option>		tmass_up.gif		tmass_up.png
<option value="5002" img="tmass de"> <caption>Mass Delete</caption> <description>Mass Delete</description> </option>		tmass_de.gif		tmass_de.png
<option value="5003" img="tmass un"> <caption>Mass Unload</caption> <description>Mass Unload</description> </option>		tmass_un.gif		tmass_un.png
<option value="5005" img="tmass cr"> <caption>Mass Create Operators</caption> <description>Mass Create Operators</description> </option>		tmass_cr.gif		tmass_cr.png
<option value="5100" img="tcount"> <caption>Count</caption> <description>Count Records in the List</description> </option>		tcount.gif		tcount.png
<option value="5110" img="tprint l"> <caption>Print List</caption> <description>Print List</description> </option>		tprint_l.gif		tprint_l.png
<option value="5120" img="trefresh">		trefresh.gif		trefresh.png

Detail/List Data entry	Windows Client		Web Client	
	Default Icon	File name	Default Icon	File Name
<caption>Refresh</caption> <description>Refresh</description> </option>				
<option value="5130" img="tmodify"> <caption>Modify Columns</caption> <description>Modify Columns</description> </option>		tmodify.gif		tmodify.png
<option value="5140" img="texport"> <caption>Export to Excel</caption> <description>Export to Excel</description> </option>		texport.gif		texport.png
<option value="5160" img="tsave as"> <caption>Save As View</caption> <description>Save As View</description> </option>		tsave_as.gif		tsave_as.png
<option value="8" img="tfind"> <caption>Find</caption> <description>Find</description> </option>		tfind.gif		tfind.png
<option value="9" img="tfill"> <caption>Fill</caption> <description>Fill</description> </option>		tfill.gif		tfill.png

Display application

The display application allows System Administrators to customize certain features of their system without altering RAD code.

Data files within Display contain the individual records, or screens, in which options, events, and window controls are defined. Each screen is attached to a particular application and controls the features appearing on the forms associated with that application. The information from these screens is stored in data tables and is not embedded in the RAD code. A system administrator has ready access to all Display features and may edit them freely.

Note: Because custom features are independent of RAD programming, Display options are protected from conflict during the upgrade process.

Displayscreen definitions

The Display Application Screen Definition (displayscreen) record is the control point for applying options to HP Service Manager forms using the Display application.

To access the Display Application Screen Definition form, click **Tailoring > Tailoring Tools > Display Screens**.

The fields available to apply displayscreen options are described below.

Field Name (*Required)	Field ID	Description
*Screen ID	screen.id	The unique name, identifying the record whose display characteristics are being defined. The names of screen IDs generally reflect their function and closely resemble the names of the forms to which they are attached.
On option 0	on.enter.key	The action to take when the user presses Enter--Option 0 (zero). <ul style="list-style-type: none"> • Redraw screen: Refreshes the screen. The cursor remains in the same field, but if this is a multi-line field, the focus resets to the top of the field. • Do nothing: With this action, the display application will execute any statements and RAD applications defined in the display option, and then return to the original screen. • Return to appl: Exits the current screen to the calling application.
Title	title	The title of the form. The value in this field can be a variable or an expression.
Format	form	The form to open for this screen. The value in this field can be a variable or an expression.
I/O (if RIO)	io	Determines the user's privileges for modifying the displayed data. HP Service Manager uses this field only when the displayed panel is replacing an rio panel.
Time (if FDISP)	time	The date and time field determining the length of time a user may search the database. HP Service Manager uses this field only when the displayed panel is replacing an fdisp panel. The value in this field can be an expression.

Field Name (*Required)	Field ID	Description
Used only for Search?	search.only	This option is selected on screens that are used only for searches (such as db.search).
User Options	user.options	If selected (set to true), triggers functionality in the Display application. Do not change this setting in the production system.
Language	language	The language used on the form. The default is English (EN).
Main tab		
Initializations subtab	expressions	Expressions that set the initial values for variables and fields. HP Service Manager evaluates these expressions before the screen appears.
Javascript subtab	javascript	The JavaScript subtab is executed after the Initializations subtab. It is used to set and initialize variables, and run expressions on the screen being initialized.
Options tab		
Option #/Label/Action/Condition	gui.option	<p>Click Search () to select a Display Application Option definition.</p> <ul style="list-style-type: none"> Detail options: These numbers also depend on the display screen and whether the option will appear on the detail options or list options. For example, the display options greater than 200 with Screen ID apm.edit.problem appear on the detail options of an incident. List options: These numbers also depend on the display screen and whether the option will appear on the list options or detail options. For example, the display options greater than 200 with Screen ID apm.list.problem appear on the list options of the incident.
Events subtab		
Event/Action/Condition	event	Click Search () to select a Display Application Event definition.

Display application option definitions

The display application allows System Administrators to customize certain features of their system without altering RAD code. You can select an option that performs an important function in your system

within the Display application option definitions form. The following fields can be defined in the Display Application Option Definitions (`displayoption.g`) form.

Field Name (*Required)	Field ID	Description
*Screen ID	screen.id	The name of the <code>displayscreen</code> record that this option is associated with.
Modifies Record	modify.record	If selected (set to true), indicates that using this display option may make changes to the record. The display routine recognizes this and locks the record before executing the action defined in the Action field.
*Action	action	<p>The display action, which is a direct link to the action in the states record. Predefined values include the following:</p> <ul style="list-style-type: none"> • Do Nothing: With this action, the display application will execute any statements and RAD applications defined in the display option and then return to the original screen. • Redraw: Refreshes the screen. The cursor remains in the same field, but if this is a multi-line field, the focus resets to the top of the field. • Close: Exits the current screen to the calling application. • More: Selecting this action has the same effect as selecting Do Nothing. • NULL: If the Action field is blank, HP Service Manager assigns a value of NULL to the Action field. • UNKNOWN: If no option has a true condition, HP Service Manager assigns a value of UNKNOWN to the Action field. This only occurs if the condition for the option has changed to false by user input. • return: • ok: Save and exit. • cancel: Exit without save. • skip: Skip execution of a script. • views: Display a list of alternate viewing forms. • extend: • find: Find the related record detail.

Field Name (*Required)	Field ID	Description
		<ul style="list-style-type: none"> • fill: Fill in information from a related record. • validatefield: • useroptions: Execute user options defined in Format Control. • closeapplication:
*Unique ID	id	The unique ID of the option generated by the system and made up of the screen ID plus action. If multiple records for the same action are required, the GUI option is also appended.
*GUI option	gui.option	<p>The following guidelines should be adhered to when entering a value into the GUI option.</p> <ul style="list-style-type: none"> • Tool tray: Enter numbers less than 200. • Detail options: Enter numbers greater than 200. • List options: Enter numbers greater than 200. • Note: Option numbers between 4001 and 4099 and over 32,000 are reserved for system use. <p>To look up the ID numbers for the GUI options for buttons on detail forms, go directly to the form, using Forms Designer, and look at the property Button ID on the buttons inside the form. To find out the GUI options for tool tray or menu options, find the default label or text alternative that matches the item.</p>
Balloon Help (If Option greater than 200)	balloon	The text string that appears when the cursor is held over a button to offer more information about the button's function.
Text Option	txt.option	The number of this option in the Text mode. The Text Option was originally used for text-based clients. This field is no longer needed.
*Default Label	default	The default label for the option, which is required on GUI mode. You could use this as a way to identify display options, such as Create Template from Record.
Bank	txt.bank	Determines the set of function keys with which this option is grouped. This field is required only when 11 or more options appear on a form. Option 12 is reserved for the More button. This field is no longer needed.
Text	txt.alternative	If not NULL, determines the label used for the F-key in a Text

Field Name (*Required)	Field ID	Description
Alternative		form. This field is no longer needed.
*Condition	condition	This field has to either contain true or false, or an expression that when evaluated returns true or false. It defines the condition for the option. If the field is true or the expression evaluates to true, the option appears to the user. Note: It is possible to have two display option records with the same display screen and the same GUI option, but they have to have mutually exclusive conditions.
User Condition	user.condition	Same rules as the Condition field. This field has to either contain true or false, or an expression that when evaluated returns true or false. The User Condition expression overrides the Condition expression.
RAD tab		
PreRad Expressions subtab	post.expressions	RAD expressions evaluated when this option is selected, but before the RAD application is executed.
PreJavascript subtab	javascript.pre	JavaScript expressions evaluated when this option is selected, but before the RAD application is executed.
RAD subtab		
Rad Application	application	The RAD application to call when an operator selects this option. Note: The RAD application, us.launch.external, does not support launching a URL that contains a semi-colon.
Separate Thread?	new.thread	This field indicates whether or not the RAD application is executed in a new application thread.
Names	names	The parameters to pass to the RAD application. Optional, unless required by the RAD routine being executed.
Values	values	The value of each parameter to pass to the RAD application. Optional, unless required by the RAD routine being executed.
Post Rad Expressions subtab	post.rad.expressions	RAD expressions to be evaluated when this option is selected, but after the RAD application is executed.
Post Javascript subtab	javascript.post	JavaScript expressions evaluated when this option is selected, but after the RAD application is executed.
Comments tab	comments	Additional comments on this displayoption.

Restricting access to display options

As a system administrator, you may wish to restrict users' access to certain display options based on their system rights. This is accomplished by creating an expression in a displayoption record that determines who may use the option. This expression is constructed with capability words found in a user's operator record. You may use existing capability words or create new ones that more narrowly define the restrictions.

To restrict access to a display option, perform the following steps:

- Identify the option and the group to whom you want to deny access.
- Define a capability word unique to the option.
- Add your capability word to the operator records of all users to whom you wish to grant access to the option.
- Create a condition expression in the displayoption record using your new capability word.

Selecting display options

Select an option that performs an important function in your system.

- Is this option part of a broader functionality?
- Should all users with rights to this functionality have access to this option?

For example, should all users with access to Incident Records in Incident Management be able to close a record?

By using a **condition** expression in a **displayoption** record, a system administrator can limit a user's right to close record without restricting rights to the other features of Incident Management or incident record options.

Note: Users with SysAdmin capability will be able to view, update and save existing incident records, but will not be allowed to close records.

Calling an application

Display panels allow the system administrator to set up a series of actions that otherwise would have to be hard-coded into each individual application. Among the added functionality is the ability to call separate applications from within the current application.

An example of this is calling the `build.list` application from a combo box on the `problem.initial` form.

The call to `build.list` recreates a list of viable subcategory, location, or other options, based on the information already appearing on the `problem.initial` form. When calling an application, several options are available, including threaded windows and scripting.

Custom RAD

Determine whether your display screen record should apply to a single form or be available to all the forms within your application. If applying to a single form, consider using Format Control. If you want to apply the display options to a single form, enter the name of that form in the **Format** field in the **displayscreen** record. If you want the same display options to be available on multiple forms, you must bind the Format field to the appropriate local variable. This is the variable you have defined in the **Array of local variables** field of the display panel in your RAD application.

When you are creating your displayoptions, the values you enter in the Action field of the **displayoption** record must be defined on a decision panel in your RAD application as **\$L.action="<action type>"**

Creating displayscreen records

Your system contains a number of **displayscreen** records that have been predefined in the RAD code. Some of these records bind their options to a specific form, while others bind the options to local variables for use with multiple forms. If you edit an option in a displayscreen record used by more than one form, every form that uses that Screen ID is affected. If you need option combinations that do not exist in the displayscreen records provided with the standard system and do not wish to edit any of the existing Screen IDs, you must create your own unique displayscreen record. You might choose to do this when:

- Writing custom RAD applications
- Creating script forms

Displayscreen records are created and hard-coded into RAD applications, which search for the appropriate **Screen ID** when called.

- RAD programmers who write custom applications for their enterprise can create new displayscreen records.
- Non-RAD licensee's can add display screens to support specific scripts because the display screen name is specified within the script record and is not hard coded in RAD.

Displayoption database dictionary keys

These keys control how the data is placed and retrieved from the file. Use values based on these keys when issuing queries on the **displayoption** file.

Field	Key	Description
unique.id	unique	Ensures record numbers are unique
screen.id language gui.option gui.sig	unique	This key ensures that no two GUI options can have the same screen ID, language, or condition.
screen.id language txt.bank txt.option txt.sig	unique	This key ensures that no two text options can have the same screen ID, language, text bank, or condition.
action screen.id language gui.option	nulls&duplicates	Useful when querying by action.

Displayevent database dictionary keys

Field	Key	Description
unique.id	unique	Ensures record numbers are unique.
screen.id language event event.sig	unique	This key ensures that no two Events records can have the same screen ID, language, event.name, object.name, or condition.

Field	Key	Description
actions screen.id language event	nulls&duplicates	Useful when querying by action.

Access display records

Applies to User Roles:

System Administrator

The Display application reads data from the following data files:

File name	Description
displaymaster	The displaymaster file contains the general controls for Display . There is one record per supported language. The default is ENG for English.
displayscreen	<p>A displayscreen record defines the attributes of a screen and provides the user with access to the individual records for options and events. Use the vertical scroll bar to view the entire form.</p> <p>All Service Manager applications using rio or fdisp panels have been converted with the display.cv utility before being shipped to you. The displayscreen records have been created and hard-coded into RAD applications, which search for the applicable Screen ID when called.</p> <p>RAD programmers who write custom applications for their enterprise will need to create new displayscreen records. Non-RAD licensee's can add display screens to support specific scripts because the display screen name is specified within the script record and is not hard coded in RAD.</p> <p>The existing displayscreen records are powerful tools for controlling the display attributes of certain forms within your system. Knowing which Screen ID is attached to a form is vital if you are to take full advantage of the features offered by the Display application.</p>
displayoption	<p>Set the various display options in the displayoption file. Display options can appear in the More Actions menu, as buttons in the toolbar, and as F-keys. Option numbers less than (<) 200 in GUI mode appear as buttons. Option numbers greater than(>) 200 appear in the More Actions menu.</p> <p>Note: Option numbers between 4001 and 4099 and over 32,000 are reserved for system use.</p> <p>RAD applications can be called from Options Definition records. Access the Options Definition record from any displayscreen record:</p>

File name	Description
	<ul style="list-style-type: none">• Click on the Ellipses button for a record list of all options.• Click on a numbered option button for individual displayoption records.
displayevent	The displayevent file defines the events a screen will handle. Access the form by clicking an Event button in a displayscreen record.

To access display records:

1. Click **Tailoring > Database Manager**.
2. Type `display` in the Form field of the Database Manager form.
3. Click **Search**. A record list of data file records opens.
4. To configure the Display application, open the **displaymaster** file first. Or select another file from the record list, and double-click the name to open it.

Create a displayscreen record

Applies to User Roles:

System Administrator

The displayscreen record is the control point for applying options to Service Manager forms using the **Display** application. You must create the **displayscreen** record first, then create the individual options in the **displayoption** file.

You can create a display screen record from scratch, or base your new record on a previously existing record.

To create a new displayscreen record:

1. Access the **displayscreen** file.
2. Type a unique name in the **Screen ID** field of the blank **displayscreen** record form. This name should identify the file involved and describe the process to which the options are attached. For example, a displayscreen record created for a script that opens incident records might be called `pm.script.open`.

3. In the **Format** field, type the name of the form on which the new options should appear.
Note: Type the appropriate format variable if you want the display options to appear on more than one form.
4. Complete any other applicable fields.
5. Click **Add**. The system creates the displayscreen record and displays the message “Record Added.”

To create a displayscreen record based on an existing displayscreen record:

1. Access the **displayscreen** file.
2. Open the **displayscreen** you want to use as a base. For the example, select **adlusermod**
3. Change the **Screen ID** to a unique name. For the example, change the name to **adlusermodxx** and then click Add. The Include displayoption/displayevent prompt opens.
4. Specify whether to add any associated displayoption or displayevent records when the new displayscreen record is added by checking the applicable box.
 - If you check **Include displayoption records**, adlusermodxx records will be created from the adlusermod records in the displayoption file.
 - If you check **Include displayevent records**, adlusermodxx records will be created from the adlusermod records in the displayevent file.
5. Click **Next**. The system creates the displayscreen record and any optional files you chose and displays the message “Record Added.”

Create a displayoption record

Applies to User Roles:

System Administrator

Data files within Display contain the individual records, or screens, in which options, events, and window controls are defined. Each screen is attached to a particular application and controls the features appearing on the forms associated with that application. For example, you can create a Validity lookup option in the More Actions menu by creating a displayoption record.

To create a displayoption record:

1. In the Service Manager command line, type `do` and then click the Execute Command icon.
2. In the **Screen ID** field of the blank displayoption record form, type the name of the screen ID you created for the displayscreen record.
3. Enter the applicable action for your option in the **Action** field.
4. Complete the displayoption record.
Important Notes:
 - If you create a `displayoption` by copying an existing record, you must change the screen ID and leave the Unique ID field blank; the system will assign a unique ID to your option.
 - If you are defining a Fill option, be sure that the value in the GUI option field matches the Button ID value of the field in the form.
5. Click **Add** to add the record to the database.
6. Open the displayscreen record for your screen ID and select the Options tab to make sure your new option appears there.

Define display conditions

Applies to User Roles:

System Administrator

Access to an option is controlled by the Condition field in a displayoption record. This field may evaluate to **true** or contain an expression defining a condition for display.

In the following example, you restrict access to the **Close** button in the toolbar of Incident Management incident records to only users with **SecClose** capabilities.

To define display conditions:

1. Select **Tailoring > Tailoring Tools > Display Options**.
2. Enter the screen ID of the update form, `apm.edit.problem`, and press Enter.
3. Locate the **Close** option from the record list. The **User Condition** field is empty by default. The User Condition is processed after the Conditions field and operates at the user-specific level.
4. Add the following condition statement to the User Condition field:

```
index("SecClose", $!o.ucapex)>0
```

Where *SecClose* is the name of the capability word you created.

5. Click **Save** to record the modification to the record.
6. Click **OK** to back out and return to the home menu.
7. Test the new display option, as follows:
 - a. Open the **Incident Management Queue** and view an incident record. The **Close** button should appear.
 - b. Open a new client as an operator who does *not* have the capability word defined in his or her operator record.
 - c. Go to the **Incident Management Queue** and view the same record as you did above. The Close button should not appear for the operator whose record did not contain the capability word.

Enable the Merge Conflicted Updates function for customized user operations

Applies to User Roles:

System Administrator

The Merge Conflicted Updates feature is out-of-box for HP Service Manager 9.41 default user operations. However, to apply the Merge Conflicted Updates function to a record with customized operations, you need to tailor your customized operations first.

The following example describes how to enable the Merge Conflicted Updates function for customized Save operation on Service Desk. In this example, you need to:

1. Enable manual merge operation for Service Desk.
2. Enable auto merge operation for a specific operation on Service Desk

Note: Manual merge operation can be enabled together with auto merge operation for a specific operation.

Task 1: Add a display option for the Merge Conflicted Updates function

1. Click **Tailoring > Tailoring Tools > Display Options**.
2. As listed in the table below, fill in the required fields to add the display option:

Field	Value
Screen ID	The display screen you are using to display Service Desk records. For example, cc.edit.incident.
Unique ID	cc.edit.incident_merge
Action	merge
GUI option	Unused number under the display screen. For example, 5.
Text option	Unused number under the display screen. For example, 5.
Bank	1
Balloon Help	Merge
Default Label	Merge
Text Alternative	Merge
Condition	updatestatus(\$L.file)=\$G.RC.MODIFIED and gui() and \$L.mode~#"add"

3. Click **Add**.

Task 2: Add process call in the State record of a Service Desk Object.

1. Click **Tailoring > Document Engine > States**.
2. Search for the state that you use to view a Service Desk record. For example, sm.view.

3. As listed in the table below, add the values to the last line of the Non-base methods table:

Column	Value
Display Action	merge
Process Name	record.update.conflicts
Condition	true

4. Click **Save**.

By completing the first two tasks, you have enabled the manual merge function for Service Desk. If the record being updated has been modified since being read, the **Merge** button appears on the option toolbar.

Task 3: Enable auto merge operation for the Save operation on Service Desk.

1. Click **Tailoring > Document Engine > Processes**.
2. Search for the Process that you use to save a Service Desk record. For example, sm.save.
3. As listed in the table below, add the values to the **Next Process** tab:

Column	Value
Next Process	record.update.conflicts.auto
Condition	updatestatus(\$L.file)=\$G.RC.MODIFIED and not (\$G.bg)

4. Click **Save**.

By completing the third task, you enable the auto merge function for a specific operation. If there are conflicted updates of the same field, the auto merge process does not merge this record. Instead, you must perform a manual merge operation.

Advanced functions

The system permits control of user interaction with the following utilities:

- Service Manager macros — invoke to do things like sending an email to a specific address when a specified event occurs.
- Publish and subscribe — display static or dynamic labels, marquees, and charts.
- Scripting — collect data by prompting users to enter specific information into their forms.

Service Manager macros

Service Manager macros are discrete units of work a System Administrator invokes to do things like sending email to a specific address. These macros are more similar to Microsoft Access macros than to Microsoft Word macros, which simply record and play back keystrokes.

Macros are distinct actions, driven by predefined conditions, that execute when a record is saved in the database. Macro actions are associated with files and reflect certain states in the records of those files. If a macro's condition evaluates to true when a record is saved, the macro's action executes. A typical condition is `priority.code in $L.new= "1"`, causing that macro's action to execute when a saved incident record has a priority code of 1.

As a Service Manager System Administrator, you can create macros to run processes automatically when specified events occur. For example, you can create a macro to send an email to a manager when an incident record hits a deadline alert.

Macro conditions

Macro conditions are expressions written in Service Manager Rapid Application Development (RAD) syntax that evaluate at run time. If the expression evaluates to true, the macro executes. If the expression does not evaluate to true, the macro cannot proceed.

A macro's condition can be very simple. For example, `true or tod() <= '17:00:00'`. Macro conditions often include a check against the record being saved. For example, a macro condition can be expressed for an action or group of actions when a specific set of incident records is saved.

The record currently being saved is identified to macro conditions as `$L.new`. This variable can be used as `$file` in Format Control expressions. Macro expressions also have `$L.old` available. This represents the state of the record before it was altered. All the following expressions are valid:

- `priority.code in $L.new="1"`
Executes whenever a priority 1 record is saved.
- `assignment in $L.new~=assignment in $L.old`
Executes whenever the assignment group changes.
- `tod() <= '17:00:00'`
Executes whenever a record is saved before 5:00 PM.

The variable `$L.message` can create evaluating expressions that gather certain information about incident records. This data is then sent as a message to specific users or groups defined in the Macro Parameters form. The `$L.message` is expressed as an array, using the following syntax:

```
$L.message={"Incident#" +number in $L.new, brief.description in $L.new}
```

The result is an array of the Incident ID and Incident Title in the record being saved (`$L.new`). The message might look like this:

```
Incident # IM1012  
Phone is going dead intermittently.
```

Accessing macro records

You can access macros from either of the following locations:

- Click **Tailoring > Tailoring Tools > Macros**.
- Incident Management Security Files.

Warning: Do not edit macros through the Database Manager. Certain processing does not occur and your edits may not be saved.

You can select individual macro records from a list or search for groups of related macros. For example, those associated with `probsummary`.

Note: You need System Administrator privileges to run the macro editor.

Access a macro record

1. Click **Tailoring > Tailoring Tools > Macros**.

The list of available macros opens. This is the access point for all your macro activities. You must go through this form to add, edit, or delete macro records.

The macro list form displays the results of macro queries. Option buttons in this form provide controls for viewing and processing macros.

2. Use the following procedure to open an individual record:

- a. Highlight an item in the list.

- b. Click **Edit**.

3. Use the following procedure to display a list of related records:

- a. Click **Search**.

The macro search form opens.

- b. Add optional search criteria in any field. For example:

- ID — the number identifying the macro. The system assigns this number.
- Filename — the Service Manager file associated with the macros.
- Name — the unique name of the macro.
- Type — the macro type for the macro(s) that you are searching.

- c. Click **OK**.

The system opens the macro list form, showing all the macros matching your search criteria.

- d. Select a macro record to edit.

You can edit existing macros and create new ones. You can also select macro types and set conditions for their execution.

The values selected in this form determine which fields show in the parameter form.

4. If you make any changes, click **OK**.

Create a macro

When creating a macro, you must name and define the conditions of the macro before setting the parameters for its execution.

1. Click **Tailoring > Tailoring Tools > Macros**.
2. Click **Add**.
3. In the **Macro Name** field, type a name for the new macro.
4. In the **Applies When** field, select an event option from the list indicating when you want the macro to execute.

For example, select **Incidents are Saved**.

5. In the **Macro Type** field, select an action you want the macro to execute. Options include mailing, starting and stopping clocks, executing a RAD function, or evaluating an expression.
6. Type a **Macro Condition** that triggers the macro to execute. When this condition evaluates to true, Service Manager executes what is defined in the Macro Type field.

For example, you can notify a specific person when a new incident record is set to priority 1. The condition would look like this: `priority in $L.new="1"`.

7. Click **Set Parameters** to establish the parameters for this macro.

Note: The available fields in this form vary, depending on the value in the Macro Type field in the edit form for your new macro.

8. Provide additional information where needed. For example, Construct Message By.
9. Click **Save**.

The macro edit form opens.

10. Click **OK**.

The macro record saves and you are returned to the macro list form. Updating a macro record uses the same edit forms as the creating process.

11. Click **Search** from the macro list form to refresh the list of macros.
12. Click **OK** in the search dialog box.

Definitions for macro forms

Macros are distinct actions, driven by predefined conditions that execute when a record is saved in the database. Definitions for macro forms are as follows:

- Macro list form
 - Column headers
 - Option buttons
- Macro editor
 - Fields

For more information on links to the definition tables, see the related topics.

Macro list form definition

Column Label	Database field name	Description
Id	id	A unique number assigned to the macro to identify it.
Filename	filename	The Service Manager file that the macro is attached to. For example, <i>problem</i> or <i>device</i> .
Name	name	The name you give the macro.
Type	type	The type of action the macro takes when activated.

Button label	Description
Add	Opens a blank macro editor form for adding a new macro.
Edit	Accesses the macro editor to change the selected macro record.
Delete	Deletes the selected macro record. Warning: When you delete a record, no warning is given and the record is simply deleted.
Search	Accesses a query form.
Clear Filter	Removes the current filter used for searching the macros and returns the list to its previous state.

Button label	Description
Back	Returns to the previous process.

Macro editor definition

Field label	Database field name	Description
Macro name	name	System administrator creates unique name for this macro.
Applies When	filename	Predefined event option for execution of this macro. Select an event from the drop-down list. For example, when a change request is saved.
Macro Type	type	Predefined macro type for this macro.
Macro Condition	condition	Define a condition for when this macro should execute. For example, when a priority 1 incident record is opened, email the members of an assignment group.

Macros provided with Service Manager

Macro Action	Description
Call A RAD Routine *	Executes a user-specified RAD routine and passes its parameters every time it executes. Warning: Macros do not work properly when calling a RAD application involving user interaction. For example, fill.recurse or validate.fields. Continue to use Format Control to call these types of applications.
Evaluate Expressions	Executes a number of user-defined expressions whenever it fires.
Get a Sequential Number	Fetches the next available sequential number of a specific class and stores it in a field in a file.
Mail 1 Person	Sends e-mail to one person only. This person can be defined as an operator, a contact, or a simple "raw" email address.
Mail a Change Request to 1 Person	Sends e-mail of a change request to one person only. This person can be defined as an operator, a contact, or a simple "raw" email address.
Mail a Change Request to a Message Group	Sends e-mail of a change request to all the members of a Change Management Message Group.
Mail a Change Request to Many People	Sends an e-mail of a change request to an arbitrarily defined list of people. These people can be defined as operators, contacts, or "raw" phone numbers.

Macro Action	Description
Mail a Change Task to 1 Person	Sends e-mail of a change task to one person only. This person can be defined as an operator, a contact, or a simple “raw” email address.
Mail a Change Task to Many People	Sends an e-mail of a change task to an arbitrarily defined list of people. These people can be defined as operators, contacts, or “raw” phone numbers.
Mail 1 Assignment Group	Sends e-mail to an entire assignment group.
Mail Many People	Sends e-mail to an arbitrarily defined list of people. These people can be defined as operators, contacts, or “raw” email addresses.
Mail an Incident Ticket to 1 Person	Mails an incident record to one person. This person can be defined as an operator, a contact, or a simple “raw” email address.
Mail an Incident Ticket to an Assignment Group	Mails an incident record to all the members of an assignment group.
Mail an Incident Ticket to Many People	Mails an incident record to an arbitrarily defined list of people. These people can be defined as operators, contacts, or “raw” email addresses.
Mail an Incident Ticket to a Change Management Message Group	Mails an incident record to all the members of a Change Management Message Group.
Mail a Task to a Message Group	Mails a change task to all the members of a Change Management Message Group.
SC Mail 1 Person	Sends Service Manager mail to one person only. This person can be defined as an operator, a contact, or a simple “raw” phone number
SC Mail an Incident Ticket to Many People	Sends an incident record to an arbitrarily defined list of people with Service Manager mail. These people can be defined as operators, contacts, or “raw” email addresses.
SC Mail an Incident Ticket to a Change Management Message Group	Sends an incident record to all the members of a Change Management Message Group with Service Manager mail.
SC Mail Many People	Sends Service Manager mail to an arbitrarily defined list of people. These people can be defined as operators, contacts, or “raw” phone numbers.
Start a Clock	Starts a specified clock.
Stop a Clock	Stops a specified clock

Publish and subscribe

You can present virtually any data generated by Service Manager to users. End users, from first level support personnel to administrators, need current, updated information about changing conditions across the system. Publish and Subscribe allows a system administrator to select pertinent data and present it to user groups in a visual format.

Publish and Subscribe saves system resources by eliminating the need for individual users to query the system every time they want an update. Desired system data is gathered by automatic background processes at prescribed intervals and published to subscriber groups.

Display objects subscribing to Service Manager data are automatically updated with current information in intervals defined by the System Administrator. Rapidly changing features and slowly developing trends can be expressed with the same process.

Forms created for specific user groups (for example, Change Management personnel or Incident Management administrators) provide the end user with dynamic system data that requires no user interaction to extract data. The System Administrator chooses the data to be published and defines display attributes, such as chart values, marquee messages, and color.

Publish and subscribe workflow

- Information is published in Service Manager when system data is associated with selected variables and stored in SYSPUB (Service Manager's publishing house). The System Administrator designates what information stored here is available to subscribers.
- Forms designed by the System Administrator for user groups that subscribe to the data by using published variables to populate display objects, such as charts and marquees.
- Charts display data in bar graph formats. For example, record activity for different categories, assignment groups, time periods, and priorities. Charts can be added to a form and updated automatically at scheduled intervals; colors can be set to reflect priorities; and button bars can be added to provide drill-down capabilities for viewing underlying data, such as individual records within categories.
- Marquees display messages in horizontal, scrolling banners on a user's screen. This is an effective method of keeping users informed of the status of the system. Marquees are added to a form in

Forms Designer and set by variables to subscribe to a particular message. They may be updated automatically at scheduled intervals.

Static messages

Simple static messages and marquees can be published to any form in Service Manager. For example, you may want to attach a static message to a menu advising a certain user group of the average time for resolution of an open record. The text of a static message does not change until it is republished in Service Manager. This is the simplest method of publishing and subscribing.

Publish the static message

1. Click **System Administration > Base System Configuration > Publishing Utilities > Publish Messages**.
2. Type the name of the variable from the Input field of the Forms properties Box (\$SYSPUB.xxx) and the text of your message in the field provided. In this case, type: The average time for the resolution of an incident record is 8 hours.
3. Click **OK** to publish your message to the Service Manager publishing house.
The following message appears in the status bar:

Your message has been published

Important: You cannot delete SYSPUB messages. You must edit the label property on the form you modified in order to remove the message.

Modify the form for static messages

This process publishes a non-updating, text message to a user's form. The message remains unchanged until you edit it manually.

1. Click **Tailoring > Forms Designer**.
2. Type the name of the form (menu.gui.pm) in the Form field of the Forms Designer dialogue box, and then click **Search**.

The Incident Management menu opens.

3. Click **Design** to open the Properties Box and the Tool Palette.

4. Use the Label tool to add a new label to the form. Ensure that the length of the label field is long enough to accommodate your message.
5. In the Properties Box, erase Caption. Ensure this field is blank.
6. Set Input to \$SYSPUB.xxx, where xxx is your initials. The Label field is an empty box outlined with a dotted line in the Design mode.
7. Click **OK** to save your changes. The label field is invisible in this mode.

Modify the form for a simple marquee

This process publishes a non-updating marquee to a user's form. The marquee message remains unchanged until you edit it manually.

1. Click **Tailoring > Forms Designer**.
2. Type the name of the form (menu.gui.pm) in the Form field of the Forms Designer dialogue box.
3. Click **Design** to open the Properties Box and the Tool Palette.
4. Add a marquee to the form with the marquee tool.
5. In the Properties Box, erase Caption. Ensure that this field is blank.
6. Set Input to \$MARQUEE.xxx, where xxx is your initials. The Label field appears as an empty box outlined with a dotted line in the Design mode.
7. Click **OK** to save your changes.

Publish the marquee

1. Click **System Administration > Base System Configuration > Publishing Utilities > Modify Marquees**.
2. Type the variable from the Input field of the Forms properties Box (\$MARQUEE.xxx) in the Name field.
3. Select a display color from the drop-down list in the Color field.
4. Type in the text of your message in the Text field. For example: The average time for the resolution of an incident record is 8 hours.

5. Click **Add** to add your marquee to the Marquee Table. The marquee is published by the marquee background agent.
Service Manager automatically adds the date and time of day to the Update Time field.
The following message is displayed in the status bar: Record added to the marquee file.
6. Return to the system administrator's home menu.
7. Click **Incident Management** to check the marquee you created.

Important: You may have to wait a minute until the background agent publishes the data.

Stored queries for management view publishing

Stored queries define the search parameters for data acquisition. The agents responsible for updating messages must query the database and report back with the current information. To publish a message that monitors alert status or counts records of a certain priority, queries must be created and stored.

The easiest method to generate a stored query is to select the information you want using normal search methods, and then open the Advanced Search window and store your query directly. By following these steps you can ensure that the query is optimized for best performance.

Select the queries you want to use for your published message from the stored query list.

Agent records for publishing

Agents are special background processes used to perform certain tasks for the user. In the case of Publish and Subscribe, they count information in your database, such as the number of open incident records in a specified alert status, or the number of records assigned to a certain group. Agent Records are the resources used by these background processes to reference stored queries, define messages and colors, and select display options. These records are published to the Service Manager publishing house and used by agents to update a subscriber's charts and marquees.

Create an agent record for publishing

1. Click **System Administration > Base System Configuration > Publishing Utilities > Define Agents**.
2. Click **Search** to open the agent records list in your database.
3. Select **pm.category** from the record list. This agent publishes records by category.

4. Select **pm.priority** to open the agent record for publishing priority status reports.
5. Refer to the field help if you have any questions about the main tab fields.
6. Click **Save** to save your changes.

Scheduling background processes for publishing

A schedule record is required for every agent that runs in Service Manager. In order for a subscriber to receive dynamic data, an agent background processor must be running to process agent records. Publish and Subscribe requires two background agents, marquee and agent, to be running in order to display data in marquees or charts.

These background agents are set to read records at specified intervals.

There are two methods for running these agents:

- Register them at startup.
- Start and stop them during a session from the `system.status.list` form.

Establish the publishing interval in the schedule file

1. Click **System Administration > Base System Configuration > Publishing Utilities > Modify Schedule**.
2. From the record list, select a record for the type of display object you want to create. For example, select Agent-Priority, which retrieves priority status data.

Tip: The entry in the Query field in the Description tab must relate to the agent record. For example, the Agent-Priority record's query entry is `pm.pri.1`.

3. Reset the counter in the Repeat Interval structure to the desired update interval.
4. Click **Save**.

Load agents into the startup record for my system

1. Click **System Administration > Ongoing Maintenance > System > Startup Information**.
2. Type `startup` in the Type field, and then click **Search**.

3. Using the scroll bar in the Processor Information structure and looking at the **Name** field for each process, make sure the **agent** and **marquee** names are listed here.

Note: To run at startup, agent and marquee must be listed.

4. If necessary, you can edit the **Wakeup Interval** in which the agent background process reads the records.

Note: The startup registry functions separately from the registry for individual agents. To change the settings for an agent, be sure to change both records.

Important: Background processes must be started from the server.

Start background processes

Agent background processes can be started from the System Status form (system.status.list.g).

1. Click the **System Status** button in the system administrator's home menu.

The System Status form opens, listing all current background processing agents.

2. Click **Start Scheduler** to open a list of background processes to start.

A list of all the available processes opens.

3. Double-click on the agent you want to start.

Note: Click **startup** to start all agent background processes.

Stop background processes

Agent background processes can be stopped from the System Status form (system.status.list.g).

1. Click the **System Status** button in the System Administrator's home menu.

The System Status form opens, listing all current background processing agents.

2. Type S in the **Command** field beside the process.

3. Click **Execute Commands**. The Background Scheduler Status form opens.

4. Click **Stop schd**. The Schedule Stop window opens.

5. In the **Schedule Stop Time** field, type the date and time you want the background process to stop.

6. Select one of the following options:
 - Click **Schedule** to stop the process.
 - Click **Cancel** to return to the system status form without stopping the process.
 - Click **Menu** to return to the System Administrator's home menu without stopping the process.

Define which system processes manage message traffic

Applies to User Roles:

System Administrator

To define which system processes manage message traffic:

1. Click **System Administration > Base System Configuration > Miscellaneous > System Information Record**.
2. Click the **Message Processors** tab.
3. In the **Processor Name** field, type the names of the background or scheduler processes that are to manage message traffic. The processor name must match the class name specified for the message class processor.
4. Click **Save**.
5. Start the message processors.

HP Service Manager rotates messages among all the processes you define. If you do not define any message processors, then Service Manager uses the problem processor to manage all message traffic.

Marquee processor

If you have selected the **Add marquee to table** function in the agent record, your marquees are not published immediately, but are added to the marquee table. To publish them from the table, you must schedule a publishing interval with the marquee processor.

Establish the publishing interval in the marquee processor

1. Click **System Administration > Base System Configuration > Publishing Utilities > Modify Schedule**.

2. Select **Marquee processor** from the record list.
3. Schedule the time interval for publishing your marquee in the Repeat Interval structure.
4. Click **Save** to save the record. The following message is displayed in the status bar: Record updated in the schedule file.

Schedule file

Once you have created your Agent Record, you must schedule it to be run by the background agent. The Schedule File enables you to establish the interval in which your background processing agents refresh your charts and marquees.

Subscribing

Subscribing to Management View data published in Service Manager requires the use of display objects whose fields are identified by selected variables. These variables, stored in Service Manager's publishing house (SYSPUB), return the data published by the Agent Record.

View published data

There are several ways in which to view published data:

- **Startup screens for user groups:** Startup screens are created with each user's system rights in mind. Custom menus with automatically-updating display objects provide end users with applicable system tools.
- **Display objects added to another form:** Existing forms and menus can be modified to display Basic or Management View data.
- **Button access to a custom form:** Add a button in any Service Manager form to access a custom form containing automatically-updating display objects.

Publishing in the RAD environment

The following is a list of Rapid Application Development (RAD) routines that you can call from Format Control to display Publish and Subscribe messages:

- `publish`
- `marquee.publish`
- `marquee.send`

Scripting

Rapid Application Development (RAD) code controls the screen flow within Service Manager applications. Scripting enables you to interrupt the normal screen flow to display a series of forms or execute decision-tree processing without modifying the original RAD code. Scripting does not affect the RAD screen flow.

When scripting is useful

Scripting is useful for any process that requires an operator to supply prerequisite information. For example, during the incident determination cycle you can create a script flow for operator-entered data. Based on how a caller replies to questions regarding the incident, your script determines which screen the operator sees next. While the script is executing, the operator-entered data is accumulated in a file variable which is returned to the calling application when the script is complete.

When a script displays a form

During execution when a script displays a form, it has the look and feel of a customized RAD application and can be used in place of most RAD routines designed to gather data from a caller. Each script can display a standard form and execute its Format Control record. The instructions in the Format Control record where there is a condition on "display" that evaluates to true will be executed before the form opens, and the instructions where there is a condition on "add" that evaluates to true will be executed after the user clicks OK. If a Format Control definition fails (an error condition is detected), the user returns to the last script form.

When multiple complex decisions must be made

Scripting is also beneficial when multiple complex decisions must be made in order to reach a conclusion. For example, Change Management approval requirement conditions are normally based on the data contents of one field in the change record. Such a condition might be expressed as

header,risk.assessment in \$cm3r>4. However, there may be circumstances where the approval requirement condition is based on the values of several different fields. For instance, there may be three fields that affect the approval requirements: division, area, and department. Hard coding all the possible combinations of these field values into condition statements in RAD involves a great amount of work and would be nearly impossible to maintain. You can define these conditions in scripting records which do not display forms but that allow you to call a RAD subroutine or execute standard Service Manager processing statements (similar to Format Control calculation statements). These options allow for the manipulation of record data. At run time, the script is a decision-tree which results in significant processing reductions over the original method of RAD coding.

Scripts

A script is a series of instructions run from an *option*. These may include a progression of screens, requesting information or user-specific responses, or instructions to perform certain computations with existing record data. Since the script call is set up and performed at the **displayoptions** application level, it does not need to be hard-coded into the **RAD** of the application. This allows for easier maintenance and modification.

Script display calls often employ the **script.execute** application, specifying the progression of forms to appear for the user. An example of this form of display call is seen in **Service Manager Request Management**, where the **New** button option forces the user through a script at open time. The script calls a preliminary information screen which requests certain data that is later populated into a new Request Management record. The screen displayed serves as an organized and user-specific entrance point into the application.

Script forms

You can create unique **displayoptions** for your script forms which are unrelated to those appearing in the application that executes the script. Changes to the **RAD** code are unnecessary when creating displayscreen records for this purpose; the appropriate variables have been hard coded into the **script.execute** application and need only be applied correctly.

Format variable

The format variable for **script.execute** is **\$L.script.format** and is hard coded into the RAD layer of the application. Enter this variable into the Format field of any displayscreen record to bind its options to multiple script forms. This local variable is bound to the display form you entered in the Format field on the script definition record.

To apply unique options to a single form in a script, name the form in the **Format** field of the displayscreen record rather than use the local variable.

Display actions

Actions for displayoptions in scripts are defined on a decision panel in the RAD code of the script.execute application. Unless you possess a RAD license and are capable of programming in RAD, you are limited to the following, hard coded actions:

- do nothing
- back
- close
- more
- redraw

Script flow

Script flow defines the order in which script panels execute. Scripts can move in a straight line from start to finish, or branch into several possible processing flows.

Condition statements that must evaluate to true before certain scripts can execute, control more complex script flows. If none of the specified conditions evaluates to true, the script flow is considered complete, and control passes back to the calling application.

Scripting processing flow

Each script can execute a Service Manager form, display options, a RAD subroutine, and condition statements. Note that each script can use all of these options. The sequence of this processing is:

1. Execute any display Format Control processing associated with the script's form.
2. Display the script form.
3. Execute any add Format Control processing associated with the script's form.

4. Execute statements defined in the script.
5. Execute the RAD application defined in the script.

Diagramming the script flow

Develop a map of the entire flow showing the name of the form (if any) displayed by each panel and the conditions controlling the flow from panel to panel. Follow the map throughout the scripting process to avoid simple errors that could prevent your script from executing properly.

The first panel of this script provides an incident record number for your incident. The second panel displays a form requesting category and asset information. Depending upon the category selected, the third panel displays a category-specific form before exiting back into the common flow. Successive forms gather the remainder of the information needed to open the record: assignments, details, service agreements, and contacts. When the script exits, the new record containing all the data you accumulate opens and is added to the database.

Notice that the two conditional script panels for the hw.desktop and hw.mainframe categories display the same form. You can write a single condition statement to include all the categories using that form.

Using fill boxes in script forms

For a Fill box to function properly on a script form, you must create the following additional records:

- **Link record:** Select Link from the Options menu in Forms Designer and establish a link between the field in your script form and the field in the source file.
- **Format Control record:** Select Format Control from the Options menu in Forms Designer and add a Format Control record granting Fill privileges.
- **Displayscreen record:** The displayscreen controls the options that are displayed with a form. Create a displayscreen record in which you define displayoptions for your script forms, including a Fill button. Enter the Screen ID of your new record in the Display Screen field of each script definition that displays a form in which a Fill box appears. You can create unique displayscreen records for each form in the script flow or associate a single record with all the forms.

Note: You can use a combination of displayscreen records and Format Control to activate Fill buttons in your script.

Displayscreen record options

If you create a displayscreen record and want to bind its options to more than one script form, you must enter the local variable `$L.script.format` in the Format field of the displayscreen record. This variable is hard coded into the `script.execute` application.

Defining displayoptions for script forms

When defining displayoptions for your script forms, refer to the following list of possible actions that are hard coded into the `script.execute` application. Your choice of options in the Action field is limited to the values in this list:

- ok
- back
- cancel
- skip
- views
- extend
- find
- fill
- validatefield
- redraw
- re draw
- closeapplication

Note: Use the ok action when defining a Continue option.

Executing scripts

After you have created your forms and the script definitions to display them, you must decide how you want to execute the script. You have several choices:

- Incident Management (using profile records)
- Display option (from within an application)
- Format Control
- Stored query

Access the scripts file

You can access the scripts file using one of the following methods:

- Menu button
- Database Manager

Executing a script from Incident Management

Profile records within Incident Management allow you to specify the script to execute when a user opens a new incident record. You can design a script that displays forms containing only those fields necessary for recording the particulars of an incident. For example, if the technician taking the call selects hw.network as the incident category, a condition statement in the script exits to a panel that displays a network hardware related form. When the script finishes, the record carrying all the necessary information is created and the technician is ready to work on another issue.

Executing a script from displayoption

Use a displayoption to create a button, the Options menu and the Context menu selections on a Service Manager form from which to execute your script. For information on the Display application, see the related topics.

Executing a script from Format Control

Scripting can be called from the routines process of Format Control. It is important to place the script call as the last item on the routine call list when executing a script from Format Control.

Executing a script from a stored query

A script executed from a stored query searches for records by allowing the user to select precise search parameters. The Service Manager standard system contains an inactive script that can execute from a

stored query to display a list of key words relating to previous incidents. The stored query then uses the key word selected to search for similar incidents.

This stored query runs from an open incident record using either Format Control or a displayoption (Screen ID: `apm.edit.problem`). In this example, a displayoption has been created that adds an option called Probable Cause to the More Actions menu of an open incident record. This option runs the stored query called `probcause.user`.

The stored query, `probcause.user`, executes the script called `probcause.user.1`.

The script `probcause.user.1` has a single panel that displays the `probcause.user.1` form. The condition for exit (`null($key.words)`) requires the user to choose a key word before allowing the script to exit.

Important: Scripts executed from stored queries typically set values into the same fields used by the query to retrieve records. In this example, the common field is `key.words`.

The script is executed before the form from the query (`probable.cause.user`) opens. The form opened by the script is called `probcause.user.1` and supplies the query form with a key word for searching.

When the user selects a key word and clicks **OK**, the script exits. The stored query then uses the key word to search the `probcause` file for matching entries. If more than one match is found, the system displays a record list of possible causes for your incident.

Click a selection to view the details with the `probable.cause.user` form. When you have finished viewing all the choices, click **Cancel** to return to the incident record.

Script reports

Script reports display the entire flow with different views. You can view the entire flow or just a part of it. You can display and print the details of each panel for comparison or troubleshooting. There are three script reports you can print:

- **Script Flow** — generates a summary report of all possible script flow paths based on the assumption that the currently displayed script panel is the starting point of the flow. This report enables you to isolate each separate branch within the script.
- **Script Detail** — lists all the fields in the script definition record and the values you have entered for the currently displayed script panel.
- **Script Tree** — generates a summary report showing the logical flow of the script panels in an outline form. The currently displayed script panel is used as the starting point. This report enables you to see the relationships of panels to one another at a glance.

Each report has the same header and prints in the default printer font.

Access a script record from a menu

1. Click **Tailoring > Scripts**.
2. Click **Search**.

A record list of scripts opens.

3. Select the target record.

Access a script record from the Database Manager

Accessing scripting from the Database Manager gives you additional options for manipulating the data. For example, you can perform a mass update to change the cluster name for each panel in your script flow or you can unload selected records to an external file.

1. Click **Tailoring > Database Manager**.
2. Type **scripts** in the Form field, and then click **Search**.
3. Click **Search** again.

A record list of existing script records opens.

4. Select the target record.
5. Make any necessary changes, and then click **Save** to save your changes.

Create a script

Rapid Application Development (RAD) code controls the screen flow within Service Manager applications. Scripting enables you to interrupt the normal screen flow to display a series of forms or execute decision-tree processing without modifying the original RAD code. Scripting does not affect the RAD screen flow. To learn more about scripting, see [Scripting](#).

To create a script:

1. Design and diagram the script flow showing the names of all script panels and forms. For information, see [Diagramming the script flow](#), [Script reports](#), and [Define the scripts](#).
2. If your script requires special forms, create them using the Forms Designer utility. Be sure to follow the naming conventions defined in your map. Create any support links and Format Control that are necessary for your forms. See the [Tailoring Best Practices Guide](#) for aids in creating forms.
3. Create the script definition records. For information, see [Define the scripts](#), [Using fill boxes in script forms](#), and [Script forms](#).
4. Add your script to the Service Manager application, Format Control record, or stored query record that will execute it. See [Executing scripts](#).

Note: If you include expressions in your script, take care to use good naming conventions to ensure unique names for any local variables you define. This helps you to avoid potential naming conflicts and achieve desired tailoring results.

Define the scripts

After diagramming the flow, the next step is to create each panel in the script flow. Make sure to follow the naming conventions for scripts and forms you have established in your script map.

1. Access a blank script definition record. For more information, see "[Executing scripts](#)" on page 385.
2. Complete the definition for the first script in your flow.
For example, enter the following values:

Field	Value	Parameter Values/Conditions
Name	pm.number	
Start	true	
Application	getnumb.fc	
Name	record prompt text name string1	\$script number string Incident Management PM
Next Script	pm.open.1	true

Note: The value for Start is true, indicating that this is the first panel in the script flow.

This panel associates a properly incremented incident record number with the data you are going to accumulate from the caller. This panel calls a RAD application only and does not display a form.

3. Click **New** to add your panel to the scripts database.
4. Click **Back** to return to a blank script definition form.
5. Complete the definition for the second script in your flow.
In this example, we would enter the following values:

Field	Value	Parameter Values/Conditions
Name	pm.number	
Format	pm.select.category	
Skip Display	false	
Bypass Cond	false	
Next Script	pm.hw1 pm.hw2	category in \$script="hw.desktop" or category in \$script="hw.mainframe" category in \$script="hw.network"

6. Click **New** to add your panel to the scripts database.
7. Create the remainder of the script definitions in your flow.

In this example, we must create specific definitions for each category (pm.hw1 and pm.hw2) as well as the remainder of the scripts in the common flow (pm.open.2 through pm.open.5). Make sure that each panel except the last panel in the flow has the name of the next panel to execute in the Next Script field.

In this example, panel pm.open.5 displays a form and calls the RAD application apm.edit.problem. This application displays any information you have gathered in an incident record and enables you to abandon the incident or add it to the database.

Define an initial script in an Incident Management profile record

1. Click **System Administration > Ongoing Maintenance > Profiles > Incident Management Profiles**.
2. Click **Search**. A list of Incident Management profile records opens.
3. Select a profile record.
4. Select the **Forms** tab.
5. Select **Initial Script** and enter the name of the first panel of your script in the adjacent field.
6. Click **Save** to save your changes.

Execute a script from a displayoption

1. Determine the screen ID of the form for which you want to define the option.
2. Open a blank displayoption record using one of these methods:
 - Type `do` in the command line and press **Enter**.
 - Type `displayoption` in the Form field of the Database Manager dialog box.
 - Click **Tailoring > Tailoring Tools > Display Options**.

3. In the **Screen ID** field, type the screen ID for the form.

4. Click **Search**.

The displayoption records for that form are listed.

5. Scan the list of options to determine an available number for your new option.

Remember: Option numbers < 200 appear as buttons in the toolbar, and option numbers > 200 appear as Options menu items.

6. Create the record from scratch or modify an existing record.

7. Type `script.execute` in the RAD Application field and enter the following values:

Name	Value
file	<code>\$L.filed</code>
name	Name of the first script panel in the flow

8. Click **Add**.

Execute a script from Format Control

1. Click **Tailoring > Format Control**.
2. In the **Name** field, type the name of the Format Control record you want to edit. For example, type `apm.quick`.
3. Click **Search**.
4. Click **Subroutines**, or click **Subroutines** from the Options menu.
5. In the **Application Name** field, type `script.execute` and enter the following values:

Name	Value
file	<code>\$file</code>
name	Name of the first script panel in the flow

6. In the **Before** field, type `true` or type an expression for a condition that evaluates to true to execute the script before any other processing takes place.
7. In the **Dis** field (Display field), type `true` or type an expression for a condition that evaluates to true to execute the script before `apm.quick` opens.
8. Click **Save** to save your changes.
9. Click **OK**.

Delete a script

You can delete script panels and the related forms manually or allow the system to delete all the elements of a script flow automatically.

1. Select the first script in the flow. To access scripts, see ["Executing scripts" on page 385](#).
Note: If you select any other script panel in the flow, you may be given a partial listing only. Partial lists stop at any script panel in the flow whose display is controlled by a conditional statement. This feature enables you to isolate a single conditional branch of a script flow for deletion.
2. Click **Delete**.
A confirmation prompt opens asking if you want to delete the related forms as well.
3. Select the **Delete associated forms** checkbox if you want to automatically delete all related forms.
4. Click **Delete**.
All the script panels in the flow and all the associated forms are listed. Scripts or forms used elsewhere in the system are not listed.
5. Click **Delete** to delete all the items listed in the form.
If the deletion procedure is successful, the following message displays in the status bar: *The Script/Format delete process is complete. Check for error messages.*
6. Click **Cancel** to exit the delete routine and return to the last form.

Print a report on a script

1. Access a script definition record. For more information, see ["Executing scripts" on page 385](#).
You can report on any script panel in the flow. If you want to view the entire flow, run the report from the first panel.
2. Click **File >Print** or the Print icon.

A prompt asks you to select a report type.
3. Select one of the radio buttons.
4. Click **Print**.

Service Manager Web tier

The Service Manager Web tier is an optional component that allows users to connect to the Service Manager server through a Web browser. The Web tier requires a Web application server and a free communications port to install. For more installation information, see the "[Interactive Installation Guide](#)" (on page 1).

When users connect to the Web tier they see the Web client, which offers several different interface views:

Web client view	Description	URL to Web client view
Power user view	This Web client view most closely matches the Windows client and is intended for Service Desk technicians, managers, and administrators. This view can provide access to all licensed modules, such as Service Desk, Incident Management, Problem Management, Configuration Management, Change Management, Request Management, etc.	<code>http://server_name:port_number/app_name/index.do</code>
Accessible view	This Web client view is a modified version of the power user view with record list mode turned off, high contrast colors, and larger fonts. This view is intended for Service Desk technicians, managers, and administrators who need accessibility features.	<code>http://server_name:port_number/app_name/accessible.do</code>
Self-Service view	This Web client view provides a simplified Service Desk interface for users to create, view, and update service requests. This view is intended for end-users as an entry point to Service Desk.	<code>http://server_name:port_number/app_name/ess.do</code>
Accessible self-service view	This Web client view is a modified version of the self-service user view with high contrast colors and larger fonts. This view is intended for end-users who need accessibility features.	<code>http://server_name:port_number/app_name/accessible_ess.do</code>

Administrators can configure the behavior of the Web tier and set Web client preferences from the `web.xml` file. In addition, administrators may customize the Web client interface by creating custom style sheets and images.

Note: Do not use the buttons (such as Refresh, Back, and Forward) on your browser toolbar or their keyboard shortcuts (such as **Ctrl+R**, **Ctrl+Left**, and **Ctrl+Right**) to perform Service Manager actions. Instead, use the buttons on the Service Manager interface, as described in [List and detail toolbars](#).

Editing Web client keyboard shortcuts

Key bindings for common operations for the Service Manager Web client are determined by standard decimal ASCII character codes. ASCII character codes are commonly available on the World Wide Web. You can change the default settings for the bindings by editing the **key_val** parameter in the shortcut.xml file on the Service Manager Web tier server.

Note: Certain third-party browser toolbars trap ALT+<key> for their own use. Be sure to select keys that will not conflict with those toolbar key mappings.

Example:

To change the messages key from Alt+g to Alt+x, open shortcut.xml and make the following change.

Old

```
<shortcut ctrl="false" alt="true" key_val="71" key_ref="keyMessages" desc_ref="Misc.ToggleMsg"/>
```

New

```
<shortcut ctrl="false" alt="true" key_val="88" key_ref="keyMessages" desc_ref="Misc.ToggleMsg"/>
```

Using the compact layout

The sizes of some user interface (UI) elements in Service Manager 9.41 are bigger than those of version 9.33 or earlier. There is a possibility that the new size may cause certain display issues in your tailored forms. In this situation, you can use the compact layout of the web client. For more information, see [Web parameter: compactLayout](#).

Setting web client preferences

You can define global web client preferences from the web configuration file (web.xml). You can find the web.xml file in the sm/WEB-INF folder of your application server installation. This file contains global preferences that you can change by editing the file with any text or XML editor. Any changes that you make to this file must conform to XML syntax requirements. You must have administrative access to the server operating system in order to set global web client preferences.

The settings that you define in the `web.xml` file determine the default client preferences for all web clients. However, users can override some of these preferences directly from the URL. For more information about which preferences users can set from the URL, see the comments in the web configuration file.

Set Web client preferences from the server

You can define global Web client preferences from the Web configuration file (`web.xml`). This file contains global preferences that you can change by editing the file with any text or XML editor. Any changes you make to this file must conform to XML syntax requirements.

The settings you define in the `web.xml` file determine the default client preferences for all Web clients. Users can override some preferences however, directly from the URL. See the comments in the web configuration file for information about which preferences users can set from the URL.

You must have administrative access to the server operating system to use this procedure.

1. Open the `web.xml` file in a text editor.
This file is located in the `sm/WEB-INF` folder of your application server installation.
2. Add or edit the preferences in the file.
3. Save the file and restart the Web application server.
All Web clients that log in to the HP Service Manager server use the new client preferences you defined.

Optimize web client cache control

Applies to User Roles:

System Administrator

You can define the web client cache control from the following control file on the Web tier server: `<webtier>/WEB-INF/classes/application-context.xml`. The web tier uses the incoming request URL to determine whether to apply one of the following cache filtering rules:

- `cacheFilter`
- `neverCacheFilter`

When the `neverCacheFilter` rule applies, the response page or resource will never be cached in the user's browser. When the `cacheFilter` rule applies, the user's browser caches the response page or resource for a period of time specified in the `max-age` value (in seconds).

The `cacheControlProxy` bean defines which web resources the `cacheFilter` and `neverCacheFilter` rules apply to. For example, the Web tier uses the `cacheFilter` rule for any web resources whose path matches `/images/**`, and sets the `Expires` or `'Cache-Control'` HTTP header in the response. The Web tier uses the `neverCacheFilter` rule on any web resources whose path matches `/**/*.jsp`, and sets the corresponding HTTP header in the response to disable browser caching for the matched resources.

Note: HP recommends that you do not change the default settings in the `cacheControlProxy` bean.

```
<bean id="cacheControlProxy" class="org.acegisecurity.util.FilterChainProxy">
  <property name="filterInvocationDefinitionSource">
    <value>
      PATTERN_TYPE_APACHE_ANT
      /images/**=cacheFilter
      /js/**=cacheFilter
      /css/**=cacheFilter
      /cwc/images/**=cacheFilter
      /cwc/js/**=cacheFilter
      /cwc/css/**=cacheFilter
      /cwc/tree/**=cacheFilter
      /gwt/**=cacheFilter
      /gxt/**=cacheFilter
      /*.do*=neverCacheFilter
      /**/*.jsp=neverCacheFilter
      /*.jsp=neverCacheFilter
    </value>
  </property>
</bean>
```

As of version 9.32, the directory structure of the Web tier has been modified to reflect the current build number of the Web tier. These changes were implemented because the SM Web tier is typically updated every few months. When these updates occur, it is sometimes necessary for users to clear their browser cache. For enterprise clients that have tens of thousands of users, getting all of those users to do this requirement is onerous. With this revised implementation, users will no longer need to clear their browser cache.

HP recommends setting the `max-age` value to a value not less than 15552000 seconds (6 months) to better utilize browser cache.

To change the `max-age` value:

1. Open the file <webtier>/WEB-INF/classes/application-context.xml in a text editor.
2. In the cacheFilter bean section, set the max-age property to an appropriate value (the default is 1555200 seconds):

```
<bean id="cacheFilter" class="com.hp.ov.cwc.web.CacheControlFilter">  
  <property name="headers">  
    <map>  
      <entry key="Expires"><value>Sat, 6 May 2017 12:00:00  
GMT</value></entry>  
      <entry key="Cache-Control"><value>public, max-  
age=1555200</value></entry>  
    </map>  
  </property>  
</bean>
```

The cacheFilter bean defines when cached web resources will expire (from the Expires key) or how long the user's browser will cache web resources (from the max-age value, if any). The max-age value supersedes the Expires value. For example, if you set the max-age to 1555200 seconds (default), the user's browser caches web resources for up to 180 days.

3. Restart the web application server for your change to take effect.

Branding the web client

Applies to User Roles:

System Administrator

The out-of-box HP Service Manager web client includes a preset HP theme, which can be modified to express your organization's own branding.

As of Service Manager 9.34, you can change your Service Manager logo icons, colors, and font style from the integrated user interface. You can also perform additional branding implementation options to change the splash screen, or to replace the images, including all the icons.

The following steps show how to implement web client branding changes.

Note: To perform branding operations, the operator should have **SysAdmin** capability. To enable the branding rights for an operator, see ["Update the operator records to enable the branding rights" on the next page](#).

1. ["Specify the location for the branding files" below](#)
2. ["Branding implementation options" on page 401](#)

If you are running on a Service Manager applications version earlier than 9.34, you need to perform an additional step to set up the branding menu. See ["Enable the branding menu in old applications" on the next page.](#)

3. ["Additional branding implementation options" on page 404](#)

Note: Currently, the branding feature does not support JAWS or color settings for the accessible.do web client.

Update the operator records to enable the branding rights

In order to prevent unintended changes or problems in the web client interface, it is recommended that the web client branding be performed by an administrator.

To enable the branding rights for an operator, follow these steps:

1. Click **System Administration > Ongoing Maintenance > Operators**.
2. Type or select optional search criteria.
3. Click **Search**.
4. Select the operator that you want to update from the record list.
5. Add the **SysAdmin** capability word to the **Execute Capabilities** table on the **Startup** tab.
6. Click **Save**.
7. Click **OK**.

This particular operator now has the right to do the branding operations.

Specify the location for the branding files

The branding feature is enabled only when you specify a location to store your branding files. If you do not do so, when you click **Tailoring > Branding**, an error message is displayed, which indicates that the

branding feature is not enabled.

To specify the location for the branding files, follow these steps:

1. Stop the web application server.
2. Specify the folder location in [Web parameter: customize-folder](#).

For example:

```
<context-param>  
  <param-name>customize-folder</param-name>  
  <param-value>C:/customize</param-value>  
</context-param>
```

Tip: To apply the branding settings from one web tier to another, copy the "customize" folder you created from the source web tier to the target web tier.

3. On the web tier host, create a folder (for example, C:\customize) . This folder is the root folder to store your branding image files, including your custom image files to replace the original ones and your branding settings in the branding menu. In this folder, Service Manager saves your color and font settings in a branding.xml file and custom images in a subfolder named **branding-images**.

Note: You must have write access to this folder.

4. Restart the web application server.

Enable the branding menu in old applications

Note: In Service Manager 9.34 or later, you only need to set the *customize-folder* parameter to enable the branding feature. However, if the version number of your Service Manager applications is earlier than 9.34, you need to perform this task to set up the branding menu.

To enable the branding menu in Service Manager applications, follow these steps:

1. Click **Tailoring > Tailoring Tools > Menus**.
2. Type **TAILORING** in the **Menu Name** field.
3. Click **Search**.

4. Select the **TAILORING** menu record.
5. Append a row to the table with the values described below, and leave the rest of the columns empty.

Option#	Description	Application	Parameter Name	Parameter Value	Condition
Enter a sequential number	Branding	us.launch.internal	url	branding.jsp	index("SysAdmin", \$lo.ucapex)>0 and sysinfo.get("environment") ~="scguiwswt"

Note: When you add the branding menu item to the applications, you assign a new id for this item. The new id may cause id conflicts when you upgrade the applications later. HP recommends that you delete this branding menu item before you upgrade the applications and add it back manually if you still need it after you upgrade the applications.

6. Save the menu record.
7. Log off Service Manager and then log back in.
8. Click **Tailoring > Branding**. The branding menu opens.

Branding implementation options

The branding menu provides the following implementation options.

Note: The new settings are applied when you next log in to Service Manager or when the system restarts.

Tip: If you want to restore the default settings, click **Restore Defaults**, and then click **Save**.

- Change the logo icons

The following table describes the format requirements for the logo files.

Logo name	Description	Format
Favorites Icon	This icon appears in the browser’s address bar, the browser tab title, and the web client URL bookmark link. <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> Note: The MIME type of the logo file should be x-icon and the browser must be able to display the .ico files. </div>	.ico
Main Page Header logo	This icon appears on the horizontal masthead at the top left corner of the web client screen.	.jpg, .gif, .png
Login and About Page logo	This icon appears on the Login page and the About information page.	.jpg, .gif, .png

To change the logo icons of the Web tier, follow these steps:

- a. Log in to the web client.
- b. Click **Tailoring > Branding**.
- c. Click **Browse** to find the logo file and select it.
- d. Click **Save**.

Service Manager saves your images in the <customize-folder>\branding-images subfolder.

- Change the colors

To change a color of the Web tier, follow these steps:

- a. Log in to the web client.
- b. Click **Tailoring > Branding**.
- c. Type a desired hex color code in the color value field. A sample color displays after the value field, which automatically changes according to the hex color code you type.
- d. Click **Save**.

Service Manager saves your changes in the <customize-folder>\branding.xml file.

You can change the following colors.

Section	Title	Description
Login Page	Body Background/Foreground	The background/foreground color of the login page
	Button Background/Foreground	The background/foreground color of the login button on the login page
	Button Hover Background/Foreground	The background/foreground color of the login button on the login page when you hover your mouse over the button
	Footer Background/Foreground	The background/foreground color of the footer on the login page
Main Page	Header Background/Foreground	The background/foreground color of the header on the main page
	Header Command Background/Foreground	The background/foreground color of the command line box on the header of the main page
	Header Inactive Switch Button	The color of the inactive switch button on the header of the main page
Navigation	Background/Foreground	The background/foreground color of the navigation accordion on the left side of the applications menu (only apply to the expanded mode)
	Menu Item Background/Foreground	The background/foreground color of the navigation menu items
	Menu Group Background/Foreground	The background/foreground color of the navigation menu groups
	Menu Group Icon	The color of the menu group icon
Tab	Active Tab Background/Foreground	The background/foreground color of the active tab

- Change the font

To change the font of the web client, follow these steps:

- a. Log in to the web client.
- b. Click **Tailoring > Branding**.
- c. Type a desired font name in the font value field.
- d. Click **Save** to save the color changes in the **<customize-folder>\branding.xml** file

- The new font takes effect only for users whose browser has the font installed. For users who do not have the font installed, the default font of the user's browser is used instead.
- In the login form container, the new font only takes effect on the login button.
- Your customized font does not apply to printed pages.

Additional branding implementation options

Service Manager provides additional branding implementation options to change the name on the splash screen to reflect your organization's name, and to replace the images, including all the icons.

- Change the applications name

To change the splash screen, follow these steps:

- a. Stop your Web application server.
- b. Navigate to the following directory:

```
<SM web tier.war file>\WEB-INF\classes\com\hp\ov\cwc\web
```

- c. Open the **app_labels_<language version>.properties** file in a text editor, for example, open the **app_labels_en.properties** file.
- d. Modify the **App.Name** and **App.Title** field values according to your organization's needs. The default value of these two fields is **HP Service Manager**.
- e. Save the properties file.
- f. Restart your Web application server.
- g. Launch the web client in your browser.

Your organization's name now displays on the login page and in the browser tab title.


- Tailor the Web client images

You can change the appearance of images in the Web client by placing a new image of the same name in the **<customize-folder>/images** folder. Any images you place here override the existing out-of-box images. Using this images folder enables you to preserve the existing HP-provided images, and ensures that any future upgrade process does not overwrite your tailoring changes. You can

delete your specified images folder at any time to restore the original HP images.

Each of your custom images must be in the .png, .gif, or .jpg format, and have the same name (but not necessarily the same format) as the out-of-box one. Service Manager looks for a custom image by name and then by format in the following order: .png, .gif, and then .jpg. For example, if the out-of-box image is back.gif, and your customization images folder contains back.png and back.gif, Service Manager uses back.png.

To change the images, follow these steps:

- a. Stop your Web application server.
- b. Create an **images** subfolder in the customize-folder you specified. For example:
C:/customize/images.
- c. Browse to the **images** folder of your Web tier to find the context path of the image you want to replace. For example, you want to replace the refresh icon  on the toolbar, the path of this icon is **<SM web tier.war file>\images\toolbar\trefresh.png**.
- d. Add the new image you under the same Web tier context path in your **images** folder. For example, you add a new image as **<customize-folder>\images\toolbar\trefresh.png** to replace the icon image mentioned in the previous step.
- e. Restart your Web application server.
- f. Clear your browser's cache.

To clear the browser's cache, for example, for Internet Explorer 10, follow these steps:

- i. Go to **Tools > Internet Options** from the menu bar at the top of your browser's screen.
- ii. On the **General** tab, click **Delete** in the **Browsing history** field .
- iii. Check all the items.
- iv. Click **Delete** to delete the browsing history.

Now you can log in to the web client to verify the new images.

If you prefer to retain the previous icons in the toolbar and forms of Service Manager 9.33, you can download the Service Manager legacy icons from HP Live Network:

<https://hpln.hp.com/node/6/otherfiles/?dir=20276>.

Service Manager provides the following packages. Copy the unzipped files to the subfolders in the <customize-folder> folder that you created earlier. The table includes example locations for the subfolders.

File name	Description	Icon location
images	This package contains the Service Manager 9.33 toolbar and form images.	C:/customize/images
toolbar	This package contains the Service Manager 9.33 toolbar icons.	C:/customize/images/toolbar

Web client forms

The Service Manager Web tier offers automatic support for existing applications and their forms. The Web tier generates dynamic HTML that approximates the exact layout of forms as you define them with Forms Designer. The default application forms are portable from the Windows client to the Web client with no required modification.

If you upgrade an existing system, you will find that Service Manager clients automatically support and display your customized forms. However, there may be cases when further form modification is necessary to correct cosmetic issues that appear when you view the form with the Web client. The following table describes common form revisions that might be required.

Area	Correction
Overlapped objects	Ensure that form objects do not overlap each other. The Windows client makes slight adjustments to correct for overlapped objects, but these design issues are exposed on a web browser.
Dynamic resizing	The Windows client dynamically resizes objects such as text areas and notebooks when you resize the window. The Web client does not resize most objects and does not support elastic properties as you resize the browser window. Therefore, ensure that you assign a default initial size.
Graphics and images	Size any graphics and images to fit conservatively in the form. The Windows client supports scaled images used as buttons. However, the Web client displays these images in their native size. The button grows to accommodate the size of the image.
Dynamic View Dependency (DVD) conditions	Hidden data with DVD conditions may be exposed if the form permits the user to make changes.

Configuring color indicator

To add color indicator settings, follow these steps:

Note:

- The color indicator requires the 9.40 (or higher) version of the Service Manager applications.
- The color indicator does not support the Accessible view (accessible.do) and the Self-Service Accessible view (accessible_ess.do).
- The color indicator only assists users who have normal vision. Therefore, the content in color indicator settings does not support JAWS.

1. Go to **Tailoring > Color Indicator Setting**.
2. Specify the following fields and then click **Add**.

File: Select the table that contains the field on which you want to apply the color indicator.

Field Caption: Select the field on which you want to apply the color indicator. This field is optional at this step; you can specify this field later.

3. Set the following fields and options:

Field name	Description
File	Select the table that contains the field on which you want to apply the color indicator.
Field Caption	Select the field on which you want to apply the color indicator.
Apply to list	Select this option if you want to apply the color indicator setting on this field in the record list.
Apply to Combo & Comfill	Select this option if you want to apply the color indicator setting on this field in the record detail. Note: In the detail form, the color indicator only applies the background color

Field name	Description
	setting to the Combo and Comfill widgets.
Apply to chart	<p>Select this option if you want to apply the color indicator settings on this field in charts of the Reporting module.</p> <p>Note:</p> <ul style="list-style-type: none"> ○ The color indicator only applies the background color settings to charts in the Reporting module. ○ The color indicator settings are only applied to the charts in the Reporting module. For example, color indicator settings do not affect the chart that you create by using More > Chart by field count.

4. Click the **Add** button in the form. The Color Indicator Setting Wizard opens.
5. Set the following fields and then click **OK**.

Field name	Description
Value	<p>Specify a possible value of the field.</p> <p>The following color settings apply when the field contains the specified value.</p>
Foreground Color	<p>Specify the foreground color.</p> <p>You can directly type in the hex code of the color, or you can first click in the text field and then select a color from the color palette.</p>
Background Color	<p>Specify the background color.</p> <p>You can directly type in the hex code of the color, or you can first click in the text field and then select a color from the color palette.</p>

6. Review the **Preview** column for the color indicator setting of each value and perform one of the following actions as necessary:
 - Click **Edit** to edit the color indicator setting for the value.
 - Click **Delete** to delete the color indicator setting for the value.
7. Repeat Step 4 to Step 6 if you need to add color indicator settings for more values.

8. Click **OK**.

The new setting takes effect in the next login.

Accessible Web client forms

The accessible mode of the Web client allows users that require different levels of accessibility to apply personal preferences to improve their user experience. The accessible Web client also enables accessibility tools, such as screen readers, to work with Service Manager. The accessible Web client omits the graphical workflow feature, and thread navigation links, which are the tabs that identify open forms in the Windows client.

If you are designing forms for accessible use, these are the most important design requirements:

- High-contrast color graphics
- Larger default fonts
- Larger form spacing (more white space)
- Simplified navigation (fewer buttons, objects and icons)
- Browser settings must be able to control
 - Resizing fonts
 - Foreground and background color selection

A visually impaired user might want a well-designed form that reads left to right with labels announcing the name of the subsequent form object and tables with row labels that read horizontally (not by column). This user might also want to specify a black background with white text in a 14-point bold font, instead of the default color and font combinations.

Putting the HTML Editor on accessible forms

The HTML Editor must be the last widget in its form for reverse tabbing to work properly. Widgets placed under it cannot be tabbed.

The Accessible Web client ignores the tab stop information set in the Form definition. Instead, the tabbing always moves from left to right and top to bottom so that the tabbing order matches the order in which the widgets are described by a screen reader.

For complex editing requiring 508 compliance, edit the text in a 508 compliant HTML Editor and then paste it into the HTML Editor.

Example: Running custom JavaScript from the Web client

This example runs a JavaScript during the Web client onload.

- The script opens an alert window before loading the Web client
 - The script returns to the Web client when the user clicks OK.
1. Log on to the Web tier system.
 2. Stop the Web tier Web application server.
 3. Browse to the **ext** folder of your Service Manager Web tier's context root. For example, **sc/ext**.
 4. Open the Java Server Pages (JSP) file named **appheader.jsp** in a text editor.
 5. Search for the following text:

```
<script type="text/javascript">
```
 6. Add the following entries to the **appheader.jsp** file just after the script element:

```
document.onload = init();  
function init()  
{  
    alert("Testing document.onload from appheader extension./n Press Ok to  
continue with navmenu application.");  
}
```
 7. Save the **appheader.jsp** file.
 8. Clear your Web application server's cache.
 9. Restart your Web application server.
The Web client now runs your custom JavaScript when the Web client starts.

Example: Sending Web tier URLs through e-mail notifications

The following example creates a notification record with the following properties.

- Creates an e-mail message whenever someone updates a Service Desk interaction record.
- The e-mail message contains a direct URL to the updated interaction record.
- The e-mail message contains a URL to the self-service interface if the record was opened from self-service or a URL to the Web client interface if it was opened from any other interface.

Note: This example assumes you have installed the Service Manager Web tier and an e-mail service.

Update the Web configuration file:

1. Log on to the Service Manager Web tier server.
2. Stop the Web tier Web application server.
3. Open the Service Manager Web tier's Web configuration file (web.xml) in a text editor.

Note: The web.xml file is WEB-INF folder of your Web tier's context root.

4. Search for the following entry:

```
<init-param>  
  <param-name>serverHost</param-name>  
  <param-value>localhost</param-value>  
</init-param>
```

5. Change the localhost param-value to the fully-qualified domain name of your Service Manager server, for example, myserver.mydomain.com.

Replace the values *myserver* and *mydomain.com* with the server host name and domain name of the Service Manager server.

Important: You must specify a fully qualified domain name for the **serverHost** parameter or the server will generate invalid URLs.

6. Save the Web configuration file.

Update the system information record:

1. Restart your Web application server.
2. Log on to the Service Manager server as a system administrator.

3. Click **System Administration > Base System Configuration > Miscellaneous > System Information Record**.

4. Click the **Active Integrations** tab.

5. In the **WebServer URL** field, type the fully qualified URL to your Web tier. For example:

`http://myserver.mydomain.com:myport/SM/index.do`

Replace the values *myserver* and *mydomain.com* with the server host name and domain name of the server running the Service Manager Web tier. Replace *myport* with the communications port your Service Manager Web tier Web server listens for HTTP requests.

The server stores the value of this field in the **\$L.web.url** global variable.

6. In the **ESS URL** field, type the fully qualified URL to your Web tier. For example:

`http://myserver.mydomain.com:myport/SM/ess.do`

Replace the values *myserver* and *mydomain.com* with the server host name and domain name of the server running the Service Manager Web tier. Replace *myport* with the communications port your Service Manager Web tier Web server listens for HTTP requests.

The server stores the value of this field in the **\$L.ess.url** global variable.

7. Save the system information record.

Update the SD.incident.mail form:

1. Log out and restart the Service Manager server.
2. Log back into Service Manager as a system administrator.
3. Click **Tailoring > Forms Designer**.
4. Search for the form SD.incident.mail.
5. Click **Design**.
6. Add a new text area field with the following properties.

Property	Value
Input	\$L.web.url
Width	400

This form will display the URL to the Web client interface.

7. Make any other changes to the form.
You will use this form as the template for your e-mail messages.
8. Click **OK** twice to exit Forms Designer and save the form.
9. Search for and select the **SD.incident.mail** form .
10. Click **More** or the More Actions icon, and then click **Copy/Rename**.
11. In the **New Name** field, type `SD.incident.mail.ess`.
12. Select the **Copy** option.
13. Click **OK**.
14. Click **Design**.
15. Select the text area field you created to display the Web tier URL and update it to have the following properties.

Property	Value
Input	<code>\$L.ess.url</code>
Width	400

This form will display the URL to the self-service interface.

16. Click **OK** twice to exit Forms Designer and save the form.

Update the SM Update notification record:

1. Click **Tailoring > Notifications > Notifications**.
2. Search for the **SM Update** notification record.
3. Add the following new rows to the notification record.

URL sent	Msg class	Msg No.	Arguments	Condition	Format	Notify Method	Recipient(s)
Self-service	sm	10	incident.id in \$.file	ess.entry in \$.file=true	SD.incident.mail.es s	email	contact.name in \$.file
Web client	sm	10	incident.id in \$.file	nullsub (ess.entry in \$.file,false)=false	SD.incident.mail	email	contact.name in \$.file

4. Click **OK** to save the notification record.

Create and update a self-service interaction record:

1. Open a Web browser and log on to the self-service interface.
2. Create a new self-service interaction record and note the interaction record ID.
3. From a windows client, click **Service Desk > Search Interaction Records**.
4. Use the interaction record ID of the self-service interaction record to retrieve the interaction record details.
5. Add a category, sub category, and update to the self-service interaction record.
6. Exit out of the self-service interaction record and return to the Service Desk search form.
7. Search for SD1001.
8. Open the **Activities** section.
9. In the **Update** field, type an update.
10. Click **Save**.

Review the output event

1. Click **Tailoring > Event Services > Output Events**.
2. Click **Search**.
3. Look for an output event with an **Event Code** of email and the ID number of the self-service interaction record that you created in the **External Information String** field.
4. If you are working in a Windows client, right-click in the **External Information String** field and click **Magnify** to display the field text in a new window.

5. Search the form for a URL.

The URL should contain the path to your self-service interface, for example:

```
http://myserver.mydomain.com:myport/SM/ess.do?  
ctx=docEngine&file=incidents&query=incident.id=%22SD10005%22  
&queryHash=eb12d3d8&action=&title=Interaction%20;SD10005
```

6. Look for an output event with an **Event Code** of email and SD1001 in the **External Information String** field.
7. If you are working in a Windows client, right-click in the **External Information String** field and click **Magnify** to display the field text in a new window..

8. Search the form for a URL.

The URL should contain the path to your Web client interface, for example:

```
http://myserver.mydomain.com:myport/SM/index.do?  
ctx=docEngine&file=incidents&query=incident.id=%22SD1001%22  
&queryHash=89472347&action=&title=Interaction%20;SD1001
```

Example: Notifying specified operators upon an incident update

This example specifies a group of operators that will receive a notification when an incident is updated.

Note: The notification will not be sent to a specified operator if the incident is created or updated when the operator is offline.

Add a Script Library record

1. Execute the `sl` command in the command line.
2. In the **Name** field, enter `IncidentMsg1`.
3. Select a package in the **Package** field.
4. Enter the following script in the text box below:

```
system.functions.msg("Incident "+vars.$file.number+" updated successfully by:  
"+vars.$lo_operator.name , 1, ["falcon","Incident.Manager"], "popup", 1);
```

You need to specify each of the operators in the script. The example script above specifies that falcon and Incident Manager will receive the notification of an incident update.

Note: The script above is an example for Service Manager 9.32. If you use a previous version of Service Manager, change `vars.$file.number` to `vars.$L_file.number`.

5. Click **Add**.
6. Click **Compile**.

Update the `im.save` process

1. Click **Tailoring > Document Engine > Processes**
2. Search the `im.save` process.
3. Select the **RAD** tab of the `im.save` process.
4. Scroll down until you find the first empty RAD Application.
5. Add the following script in the **Post RAD Expression** field of the empty RAD Application:

```
if $L.continue then ($L.void=jscall("IncidentMsg1"))
```

6. Click **Save**.

Generating Web tier URL queries

If you have installed a Web tier, you can have Service Manager server generate URL queries to your Web tier that display specific records in your applications. You can use these generated URL queries to programmatically send links to your users when records are opened, updated, or closed. Service Manager offers the following out-of-box methods for generating valid Web tier URL queries.

- Create custom JavaScript that uses the **makeSCWebURL** method. You can invoke your custom JavaScript from a macro or RAD expression.
- Create custom notification records that use the Web tier URLs defined in the system information record.

When a user clicks on a generated URL query (link), Service Manager requires the user to log in before displaying the requested data, although the user can bypass the login screen if you have enable trusted sign-on on the Web tier.

By default, the Service Manager server secures Web tier URL queries by including a unique hash key in the URL. This unique key prevents people from modifying the URL and attempting to access restricted parts of your system. If the URL query does not match the unique hash key, Service Manager displays a warning message in the log file.

If you want to generate URL queries outside of Service Manager you can disable the unique hash key using the **querysecurity** parameter. Disabling this parameter in the Service Manager initialization file allows the server to accept URL queries without a unique hash key. If you disable the security hash, access to tables is controlled by the Document Engine. To restrict access to tables through the Document Engine, you must enable security features such as format control or application profiles that restrict access at the operator level.

Web tier URL format

Service Manager requires generated web tier URLs to use the following format.

```
http://myserver.mydomain.com:myport/SM/index.do?ctx=docEngine
&file=incidents&query=incident.id=%22SD1001%22
&queryHash=89472347&action=&title=Interaction%20;SD1001
```

URL portion	Required?	Description
http://myserver.mydomain.com:myport/SM/index.do	Yes	This portion of the URL specifies the web tier host name, port, and web client

URL portion	Required?	Description
		connection to use. Use <i>index.do</i> to have users connect to the standard web client. Use <i>ess.do</i> to have users connect to the self-service web client.
ctx=docEngine	Yes	This portion of the URL specifies that Service Manager use the Document Engine to fulfill the query.
&file=incidents	Yes	This portion of the URL specifies the table you want to query for records. You must use the file name for the table as specified in the database dictionary.
&query=incident.id=%22SD1001%22	Yes	This portion of the URL specifies the Service Manager query you want to use to search for records. You must URI encode the query string to prevent special characters from invalidating the URL. For example, use %22 instead of quotation marks.
&queryHash=89472347	No	This portion of the URL specifies the optional hash key used to encode the URL query. The hash key prevents users from modifying the URL query to view other portions of Service Manager. You can only include a hash key if you generate the URL from the makeSCWebURL method.
&action=	No	This portion of the URL specifies the Document Engine action you want the URL query to perform. By default, the URL query performs a search operation.
&title=Interaction%20;SD1001	No	This portion of the URL specifies the optional title of the query. You must URI encode the query string to prevent special characters from invalidating the URL. For example, use %20; instead of a space.

Windows client

HP Service Manager includes both Windows and Web clients. The Windows client is intended for administrators and implementers, while the Web client is suitable for most end users. The Windows client enables you to design forms and work with the dbdict; capabilities that are not available to the Web client user.

The Windows client includes the following features:

Interface

- A dynamic user interface that you can customize with preferences to change the way you interact with perspectives and views

- Windows client update capability

Client/server connection

- A Connection dialog that enables you to define multiple connections to different servers and saves them from one session to another

- Automatic connection for connection profiles with the automatic logon option enabled

System Navigator

- Favorites folder to save shortcuts to frequently used applications, forms, and queries

- Simplified access to forms, files, and records through the System Navigator

Applications

- Application-specific user interface elements

- A graphical workflow used by the Request and Change Management applications

- Charts and dashboards to show data graphically

Help

- Field help

- Help for end users and administrators

Other features

- UTF-8 (Unicode) support to display data in any language

- SOAP-based client/server communication

- Automatic reconnection of client/server sessions after brief network connection disruptions

Administration perspective

Note: Views and perspectives are available in the Windows client only.

The Service Manager Administration perspective contains several Service Manager administrative views. It is easy to show or hide administrative views by selecting or closing a single perspective instead of each individual view. You can use the Administration perspective to troubleshoot the user interface or monitor client/server traffic.

Within the Administration perspective, there are eight available views arranged in tab format. Each view displays different information.

- Messages
- Console
- Detail Data
- Detail Form
- List Data
- List Form
- Last Request
- Last Response

Console view

Note: Views are available in the Windows client only.

The console view displays the information sent to the system log, depending on the logging level. If you enable SOAP tracing from the Connection Dialog, SOAP messages also appear. If an error occurs, you may also see error messages or a stack trace from the Java Runtime Environment (JRE). If you are an administrator, you might want to monitor the console during normal activities instead of locating the sm.log file and opening it with a text editor. Ensure that you do the following:

- Specify the sm.log file in the **Window > Preferences > Service Manager > Logs** preferences.
- Ensure that the log file is accessible from your client.

Stopwatches view




Note: Views are available in the Windows client only.

The Stopwatches view is an administrative tool that lists information about the number of milliseconds required for some Service Manager client operations. This view shows the number of individual service desk interactions, the shortest time (Min), the longest (Max) time, the average, and the total time. It is helpful for troubleshooting.

The number of displayed operations varies. They are neither configurable or predictable because they depend on your Service Manager installation and current actions. The following table lists some operations that might appear in this view.

Operation	Purpose
Form.generate	The elapsed time to render the form using information sent by the server.
Generator.generate	The elapsed time required to retrieve a System Navigator branch from the server.
Navigator.getChildren	When you click a closed node in the System Navigator, the elapsed time required to expand that node and render content.
SOAP.transact	the elapsed time to exchange one message between the client and server. The stopwatch starts when the client sends the message and stops when it receives the server response. This time does not include message processing.
XSL.transform.jdom	The elapsed time for XSL transformation of forms and menus.

Click the appropriate icon to modify the Stopwatches view.

Icon	Function
	Refresh all stopwatches.
	Reset all stopwatches.
	Save current stopwatches state in the log file.

XML views

Note: Views are available in the Windows client only.

The Administrator perspective contains these views that display data exchanged between the client and server.

View name	What's in this view?	Use this view to	Related capability word
Detail Data	Displays all data in the records (in XML format) that populate the	Trace or debug Service Manager	Requires the Debug capability word.

View name	What's in this view?	Use this view to	Related capability word
	current form. For example, this view shows related query information, as well as RAD-defined variable values, associated displayoptions, and messages.	transactions.	
Detail Form	Displays the form fields (in XML format) in the current form.	Trace or debug Service Manager transactions, Dynamic View Dependencies (DVD), or display applications.	Requires the Debug capability word.
List Data	Displays all records (in XML format) that appear in the current record list.	Trace or debug Service Manager queries.	Requires the Debug capability word.
List Form	Displays the form fields (in XML format) for the current record list form.	Trace or debug Service Manager queries.	Requires the Debug capability word.
Last Request	Displays the XML-formatted action, modified data, and cursor position of the last request sent from the client to the server.	Trace or debug client/server communication.	Requires the Debug capability word.
Last Response	Displays the XML-formatted data of the last response sent from the server to the client.	Trace or debug client/server communication.	Requires the Debug capability word.

Administrative views

You can select Service Manager views individually to create a customized workbench or saved perspective.

Administrative and RAD Debugger views appear only if you have the Admin plug-in installed. Each view displays different information.

View name	Included in this perspective	What's in this view?	Use this view to
Console	Service Manager Administrator	Displays log messages for the client, depending on your log preferences.	Review log messages.

View name	Included in this perspective	What's in this view?	Use this view to
Detail Data	Service Manager Administrator	Displays all data in the records (in XML format) that populate the current form. This view shows related query and SLA information, as well as RAD-defined variable values, associated display options, and messages.	Trace or debug Service Manager transactions.
Detail Form	Service Manager Administrator	Displays the form fields (in XML format) that appear in the current form.	Trace or debug Service Manager transactions, Dynamic View Dependencies (DVD), or display applications.
Last Request	Service Manager Administrator	Displays the XML-formatted action sent from the client to the server.	Trace or debug client/server communication.
Last Response	Service Manager Administrator	Displays the XML-formatted response sent from the server to the client.	Trace or debug client/server communication.
List Data	Service Manager Administrator	Displays all records (in XML format) that appear in the current record list.	Trace or debug Service Manager queries.
List Form	Service Manager Administrator	Displays the form fields (in XML format) for the current record list form.	Trace or debug Service Manager queries.
Properties	Service Manager Administrator	Displays the properties of the current form.	Troubleshoot form problems.
RAD Debugger	Service Manager Administrator	Isolated RAD processes.	Troubleshoot RAD processes.
Stopwatches	Service Manager Administrator	Displays timing information about Service Manager processes.	Identify slow processes or opportunities for system tuning.
Dashboard	Service Manager Interface	Displays one or more charts.	Compare data visually.
Messages	Service Manager Interface	Displays client messages, depending on your log preferences.	Review log messages.

View name	Included in this perspective	What's in this view?	Use this view to
Secondary System Navigator	Service Manager Interface	Displays another view of the System Navigator at the bottom of the workbench. You can move this second view to another location.	Add another navigation view to access all Service Manager components.
System Navigator	Service Manager Interface	Displays a tree view of Service Manager applications, utilities, and data.	Access all Service Manager components.

Access an administrative view

Applies to User Roles:

Administrator

Note: Views are available in the Windows client only.

The administrative view provides additional panes to display different types of information that may be useful in a debug or problem solving situation for an administrator. In the out-of-box system, the messages and system navigator panes are standard views.

To access an administrative view:

1. Click **Window > Show View > Other**.
2. Expand the **Service Manager Administrator** folder. Choose one of these views:
 - o **Console**
 - o **Detail Data**
 - o **Detail Form**
 - o **Last Request**
 - o **Last Response**
 - o **List Data**
 - o **List Form**
 - o **Properties**
 - o **RAD Debugger**

- **Stopwatches**
- **Web Preview**

Preferences

You can customize the way you view the Windows client by changing settings in Preferences dialog.

If you are an administrator, you can configure a common set of Windows client preferences. For example, you can ensure all users have the same log file, security, and help server settings.

Administrators also have the option to set Windows client preferences in the sm.ini file. These preferences override any conflicting settings that appear in the Windows client Preferences dialog.

Administrators can configure a common set of Web client preferences in the ../WEB-INF/web.xml file. Thereafter, these preference settings apply to all Web client sessions.

If you are an end user, you can change default settings to customize your Windows client environment. For example, you can change some appearance, chart, and spell checking properties.

There is a Preferences filter available that enables you to filter preferences so that only a few preferences are displayed or a specific preference set (for example Appearance) is displayed. By typing the beginning letters or complete name of the preference setting in the Preferences filter text box, you can filter the list so only those preferences that match what you type appear in the list.

Set Windows client preferences

Applies to User roles:

All roles

To set Windows client preferences:

1. From the HP Service Manager menu bar, click **Window > Preferences**.
2. Expand the HP Service Manager Preferences filter:
3. Highlight the following filter settings to update your preferences:
 - **HP Service Manager**
 - **Appearance**

- **Charts**
 - **Logs**
 - **Security**
 - **Spell Checker**
4. Expand the Help Preferences filter.
 5. Highlight the following filter settings to update your Help preferences:
 - **Help**
 - **Help Server**
 6. Restart the client for these preferences to take effect.

Appearance preferences

This Preferences dialog enables you to customize the Windows client.

To access appearance preferences, from the HP Service Manager menu bar click **Window > Preferences > HP Service Manager > Appearance**.

Preference	Description	Default setting
Use server provided tab order	The default Windows tab order is from left to right, and top to bottom. When you specify a custom tab order with Forms Designer, the server follows this order.	False
Show Messages View when new messages occur	Enable a view to display messages.	False
Skip readonly controls when tabbing	Enable so that read-only fields will be skipped when tabbing through a form.	False
Prompt for save	Prompt to save information before you leave a form.	True
Show detail pane with record list	Display the record list at the top of a form.	True
Sort menu items alphabetically in the System Navigator	Display System Navigator menu items alphabetically.	True
Show context-sensitive help	Display context-sensitive help test information.	True

Preference	Description	Default setting
debug information		
Font	Specifies the primary font used by Service Manager. Hewlett-Packard delivers Service Manager with a default font for each language. For example, in English, Tahoma Regular is the default font. In Japanese, MS UI Gothic is the default font. Although you can change the font face and size, Hewlett-Packard recommends the default font for the best results. Font sizes larger than 8 points may produce unpredictable results.	Varies depending on the language pack installation
Percentage to increase form width	Specify a percentage to increase the width of the forms. Useful to display localized forms that may require more space to display translated text.	0

HP Service Manager preferences

This Preferences dialog enables you to customize the Windows client.

To access HP Service Manager preferences, click **Window > Preferences > HP Service Manager**.

Preference	Description	Default setting
Client printing	This preference allows users to access the client-side printing feature by using the Print icon in the Windows client. Web and Windows clients can also use the Print commands supplied by the web browser or Windows operating system.	True
Client side load/unload	This preference determines the destination of unload files. When enabled, HP Service Manager saves unload files on the client system. When disabled, HP Service Manager saves unload files on the server. Requires a client restart to take effect.	True
Use form caching	Form caching improves server responsiveness by reducing the need to load forms.	True
Record list request count	This preference specifies the number of records the client requests from the server when displaying a record list. The user can view additional records in the record list by advancing to the next page of records or by using the go to page option. The default value is set to maximize the server's response time to client record requests. The fewer records the client requests of the server the faster the server can answer the query.	32
Image path	This preference specifies the path to any custom images you want	None

Preference	Description	Default setting
	the Windows client to use. System administrators may use this preference to customize the Windows client to use custom branded images.	
Display names of missing images	This preference determines if the Windows client displays the name and of any missing images. This preference is intended to make customizing the Windows client easier.	False
DDE server name	This preference determines the name the Windows client uses when acting as a DDE server for integrations with DDE applications such as computer telephony integration (CTI).	HP_Service_Manager
Show line number	This preference displays line numbers in the comparison window when you perform application upgrades or apply application patches.	True

Log preferences

You can set log preferences to control how Service Manager logs and displays messages.

To access log preferences in the Windows client, click **Window > Preferences > Service Manager > Logs**.

Preference	Description	Default setting
Use a specific file	If you want to use a specific log file, type the path and name of the file or browse to its location.	The default log file is C:\Users\ <username>\ServiceManager\workspace\metadata.log.</username>
Message level for console view	Choose the severity of messages that appear in the Console view. Select these logging options: None Debug, Info, Warning, and Error messages Info, Warning, and Error messages Warning and Error messages Error only messages	None
Message level for log file	Choose the severity of messages that appear in the log file. Error only Debug, Info, Warning, and Error	Error only

Preference	Description	Default setting
	messages Info, Warning, and Error messages Warning and Error messages None	
Append exception trace to log option	Add the stack trace of the associated exception to the log.	False

Security preferences

Security preferences point to the location of the CA certificate keystore, client certificates, and private keys.

To access Security preferences in the Windows client, click **Window > Preferences > Service Manager > Security**.

Preference	Description	Default value
CA Certificates File	Specify the path to a keystore of Certification Authorities (CA) used to certify the client and server certificates.	Service Manager has an out-of-box example of this file in \<Web Tier>\WEB-INF\cacerts.
Client keystore file	Specify the path to the keystore containing the client certificate and associated private keys.	None. You must create this file.
Keystore password	Specify the password to access the keystore.	None.
JCE provider name	Specify the name of a FIPS-certified third-party JCE provider you use when configuring FIPS mode in the Windows client.	None
JCE provider class name	Specify the class name of a FIPS-certified third-party JCE provider you use when configuring FIPS mode in the Windows client.	None

Spell Checker preferences

These preferences specify how to customize the spell checker. To access Spell Checker preferences in the Windows client, click **Window > Preferences > Service Manager > Spell Checker**.

You can select or clear **General Options**, customize the dictionary, and choose one of these language dictionaries:

- American English
- British English
- French
- German
- Italian
- Spanish

Note: You must select a custom dictionary if you want to add new terms.

Spell checker custom dictionaries

You can configure the Service Manager Windows client to use a custom dictionary when running the spell checking utility. You can provide a custom dictionary for all Windows clients by saving the dictionary file on a network share and using the Client Packaging Utility to direct the Windows clients to the network path.

Change the spell checker dictionary

Applies to User Roles:

Administrator

You must have administrative access to the server operating system to use this procedure.

To change the spell checker dictionary:

1. Log in to the Windows Client.
2. Click **Windows > Preferences**.
The Preferences window opens.
3. Click **Spell Checker Options**.
The Spelling Preferences window opens.
4. From the **Language** box, select the language dictionary you want to use.

5. To define a custom dictionary, click **Browse**.
 - Type or browse to the path of your custom dictionary.
You can either create your own file or use the default custom dictionary file `customdict.tlx` provided in the `plugins\com.hp.ov.sm.client.eclipse.user_version_number\spellchecker` folder.
 - Click **OK**.
6. To add correction entries to the custom dictionary, click **Edit**.
The Edit user dictionary window opens.
 - To add a new correction entry to the dictionary, type the *misspelled* word in the **Words** field.
 - Type the *correction* in the **Other word** field.
 - Select an **Action** from the available list.
 - Click **Add word**.
 - Repeat these steps for each custom correction entry you want to add.
7. Select any other spelling preferences you want to use.
8. Click **OK**.

Help preferences

This Preferences dialog enables you to customize the way Service Manager help information is displayed.

To access Help preferences, click **Window > Preferences > Help**.

Preference	Description	Default setting
Use external browser	Opens the Service Manager help in your default web browser (for example, Internet Explorer). By default, Service Manager opens the help in a Help View window.	False
Open window context help in a dynamic help view	Opens window context help information in a dynamic help view.	True
Open window context help in an infopop	Opens window context help information in a pop-up information window.	False

Preference	Description	Default setting
Open dialog context help in dialog tray	Opens dialog context help information in the dialog tray.	True
Open dialog context help in an infopop	Opens dialog context help information in a pop-up information window.	False
Show all potential hits (faster)	Shows all records that fully or partially match your search criteria.	True
Show only actual hits	Shows only records that fully match your search criteria.	False

Help Server preferences

This Preferences dialog enables you to specify your Help Server settings so that you can access the online help documentation by clicking **Help > Help Contents** from the Windows client. Service Manager launches the Help Server using this URL: `http://<Help Server host name>:<Help Server port number>/<Help Server context>`.

To access Help Server preferences, click **Window > Preferences > Help > Help Server**.

Preference	Description	Default setting
Use a Help Server to access documentation	Enables you to access the online help from Service Manager. When selected, this preference requires the Service Manager help be deployed on a web server (for example, Apache). For information about how to deploy the Service Manager help on a web server, see the <i>Service Manager Interactive Installation Guide</i> .	False
Help Server host name	Specifies the host name (IP address or fully-qualified domain name) of the web server on which the Service Manager help is deployed. For example, if you have deployed the help on Apache, this is the Apache web server host name.	None
Help Server port number	Specifies the communications port of the web server on which the help is deployed.	None
Help Server context	Specifies the virtual directory name where the Service Manager help is installed. It excludes the web server's document directory path. For example, if the help is deployed in <code>C:/Apache/2.2/htdocs/sm_help</code> , the document directory path is <code>C:/Apache/2.2/htdocs/</code> and the virtual directory name is <code>sm_help</code> . Therefore, the context path is <code>sm_help</code> .	help

Preference	Description	Default setting
	If you use the default value, the Service Manager help must be deployed in, for example, the <Apache>/htdocs/help directory.	

Setting Windows client preferences from the server

You can define global Windows client preferences from the *sm.ini* file available in the *RUN* folder of your HP Service Manager server installation. The client parameters you define in this file override any individual client preferences. You must have administrative access to the server operating system to set Windows client preferences using client parameters.

Set Windows client preferences from the server

You must have administrative access to the server operating system to use this procedure.

To set Windows client preferences from the server:

1. Review the list of client parameters for Windows clients and determine which parameters you want to define from the server.
2. Open the *sm.ini* file in a text editor.
This file is located in the *RUN* folder of your HP Service Manager server installation.
3. Add or edit the client parameters for Windows clients.
4. Save the file.
All Windows clients that log in to the HP Service Manager server use the new client parameters you defined.

Views

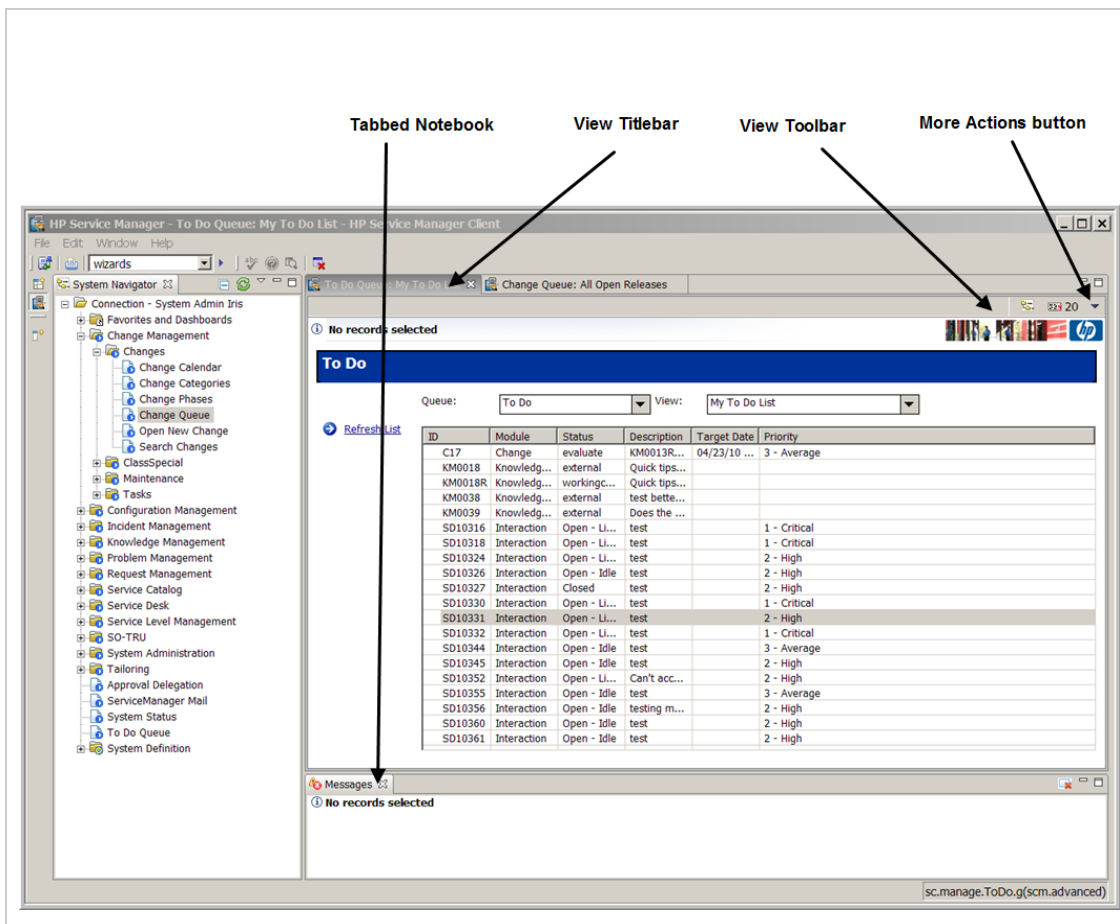
Views are a Windows-client feature that enable you to display editors and system information outside the editor, and provide alternate ways to access forms. For example, you can navigate the file system hierarchy with the System Navigator, and access and edit information about a form using the Properties view. Service Manager saves any changes that you make in a view immediately.

A Perspective can contain one or more views.

Views have menus and toolbars that affect only the items in the view. For example, if you click **Expand All** in the System Navigator, the tree expands but you see no change in the editor.

Most views also have a *More Actions menu* that displays a list of related Service Manager actions and a *context menu* that displays a list of related actions for the view. Right-click the view's titlebar to display the view's context menu; right-click the form or record list to display Service Manager's options menu.

You can save screen space and customize your perspective by stacking views. Drag one view over another to create a *Tabbed notebook*. You can also change the view sizes and create fast views. You cannot open more than one of each view at one time. If necessary, use the Secondary System Navigator to create a second instance of a view.



Component	Description
View Toolbar	Each view has an individual toolbar with buttons that support available functions.
View Titlebar	Click the X to remove the view from the workspace. Right-click the view's title bar to display the view's context menu.
Tabbed Notebook	Tabbed notebooks support stacked views. Click any tab to move that view to the top of the stack.
More Actions Menu	Click the down arrow to display the More Actions menu. Right-click the form or record list to display the Service Manager's More Actions menu.

Views

The following information identifies all of the views available in Service Manager.

Note: Views are available in the Windows client only.

Service Manager Administrator views

View	Description
Console	Displays real-time logging and debugger messages.
Detail Data	Displays all data in the records (in XML format) that populate the current form.
Detail Form	Displays the form fields (in XML format) that appear in the current form.
Last Request	Displays the XML-formatted action sent from the client to the server.
Last Response	Displays the XML-formatted response sent from the server to the client.
List Data	Displays all records (in XML format) that appear in the current record list.
List Form	Displays the form fields (in XML format) for the current record list form.
Properties	Displays the properties of the current form.
RAD Debugger	Displays isolated RAD processes for troubleshooting.
Stopwatches	Displays timing information about Service Manager processes.

Service Manager Interface views

View	Description
Dashboard	Displays one or more charts.
Messages	Displays client messages, depending on your log preferences.
System Navigator	Displays a tree view of Service Manager applications, utilities, and data.
Secondary System Navigator	Displays another view of the System Navigator at the bottom of the workbench. You can move this second view to another location.

Fast views

Note: Views are available in the Windows client only.

Fast views are hidden views that can quickly be made visible. They work the same as normal views, only when they are hidden they do not take up screen space on the Workbench window.

Once you have created a fast view, the icon for the view appears on the shortcut bar. You can look at the view by clicking its icon on the shortcut bar. As soon as you click somewhere else outside the view, it is hidden again.

To convert a fast view that has been maximized back to a regular sized view, select **Restore** from the context menu of the icon in the top left corner of the view. To reposition a fast view, drag the fast view's title bar (or close the fast view and then drag its button from the shortcut bar) and drop it on the workbench like a normal view.

Create a fast view

Applies to User Roles:

All roles

Note: Views are available in the Windows client only.

There are two ways to create a fast view.

- Using drag and drop.
- Using a menu operation available from the view System menu.

Create a fast view using drag and drop as follows.

1. Click the title bar of the view that you want and hold the mouse button down.
2. Drag the view to the **Fast View** bar and release the mouse button. By default the shortcut bar is located in the lower left corner of the window.

To create a fast view using the second approach, click on the button located on the left side of the Fast View bar which presents you with a selection of views. Choose one of these views to add it to the Fast View bar immediately.

Related objects

Related objects are data-model relationships created by a System Administrator for users of a specific database. Administrators can customize System views for users that specify the relationships between database objects and fields. These relationships can define one-to-one, many-to-one, or many-to-many relationships. One-to-one and many-to-one relationships are defined by a database JOIN between two corresponding tables for the data relationship. Many-to-many relationships are modeled by a relational database JOIN between the two corresponding tables and the cross-link table defining the relationship.

Workflow view

The workflow view is a graphic representation of the Change, Request, or Problem Management process definition, and which step each existing change request, quote, problem, or known error is at in that process.

Note: To successfully use the workflow features of Service Manager, make sure you have Java 1.5.0 or higher installed.

The workflow view has interactive features that enable you to expand the view of any component in the workflow. Service Manager displays graphic views of the workflow for these applications:

- Change Management change categories
- Change Management task categories
- Change Management change requests
- Problem Management problems
- Problem Management known errors
- Request Management quote categories
- Request Management quotes
- Request Management order categories
- Request Management orders

An administrator can view the defined workflow for any selected category. The administrator can also expand each phase in the workflow to view required approvals.

For change requests or quotes, a user can view the workflow to see:

- The current phase the change request or quote is at in the process.
- Who issued required approvals.
- The result of the approvals.
- Which phases are incomplete.

For problems and known errors, a user can view the workflow to see:

- The current phase the problem or known error is at in the process.
- Open and closed tasks.
- Which phases are incomplete.

Messages view

Service Manager notifies you when you have activity messages to review. For example, each time you add or change a record, open or close an incident, complete a Service Manager task, or an error occurs, Service Manager generates a message. These are the choices you have to view messages:

- Configure the Preference dialog to display the Message view by default.
Click **Window > Preferences > Service Manager > Appearance** and choose this preference:
Show message window when new messages arrive.
The Message view opens automatically when messages occur.
- Show the Messages view on demand.
Click **Window > Show View > Messages**.
- Show the Messages view as needed. When Service Manager sends a new message, an information icon appears.
Click the icon or the toolbar.
The Message window displays the new message and any others that appear during your Service Manager session.

The server can process different threads concurrently and each thread can generate a separate stream of messages. If you have a single client running, all messages generated by that client connection appear in the same Messages view. If you have more than one client connected, each client window has its own Messages view that might display only the messages affecting that connection.

Active notes

The Service Manager active notes feature notifies you when new activity messages appear in the Messages view. For example, each time you add or change a record, open or close an incident, complete a Service Manager task, or an error occurs, the active notes icon notifies you that a new message is available. To activate the active notes feature, you can set preferences to enable the Messages view to appear each time a new message occurs.




The Active Notes icon is flat when there is only one message and stacked when there are multiple messages.

View messages

Applies to User Roles:

All roles

Service Manager messages appear at the top of the detail window.

Icon	Icon name	Meaning
	Information	Information about the most recent action
	Warning	A warning about the most recent action
	Error	The most recent action caused an error to occur

To view all messages in a Service Manager session:

1. Click the message icon to open the Messages window.
2. To copy a message, select the message and use any standard Windows copy operation, such as Ctrl+C, to copy it to the clipboard.
3. Click the Close button to close the Messages view.
4. To erase all messages, click **Erase**.

Note: If you enable the Messages view from the Service Manager Preferences dialog, it appears automatically when messages occur.

Perspectives



Note: Perspectives are available in the Windows client only.

A perspective contains one or more views, and perhaps an editor, that provides all of the data and tools you need to complete a Service Manager task. Service Manager has a default perspective that is standard for accessing forms and data. The Administration perspective contains several Service Manager administrative views that help troubleshoot the user interface or monitor client/server traffic.

You can customize these standard perspectives, or create new ones according to your needs. For example, one user might want to add another view to all of the default Service Manager views. If you save this with a name, you can reuse it. Another user might want to see the same views, but docked or stacked differently within the Workbench. Building a customized perspective is similar to creating a personal Windows desktop.

The default Service Manager perspective opens the first time you log in to Service Manager. Click **Window > Open Perspective** to choose a default or customized perspective. You can also access perspectives from the icons on the left shortcut bar. When you choose a perspective for the first time, its icon appears on the shortcut bar and remains there until you end the session.

The following table lists available perspectives in Service Manager. You can define your own perspectives to customize the way you work with the Windows client.

Icon	Perspective name	Purpose	User
	Administration	Troubleshoot the user interface or monitor client/server traffic. Within the Administration perspective, there are eight available views arranged in tab format. Each view displays different information. For more information, see the Application Administration help.	Administrators
	Service Manager	The default perspective when you first log in. This perspective is for any Service Manager user and displays view and editor panes that help you access Service Manager applications.	Service desk operators and administrators

Create a new perspective

Applies to User Roles:

All roles

Note: Perspectives are available in the Windows client only.

To create a new perspective:

1. Start in an out-of-box perspective. The default perspective is the Service Manager perspective.
2. Arrange the views and editor according to where you want them to be.
3. Click **Window > Save Perspective As**.
4. Name the new perspective, then click **OK**.
5. Click the perspective icon on the shortcut bar. The new perspective is in the list of perspectives.

Open a list of perspectives

Applies to User Roles:

All roles

Note: Perspectives are available in the Windows client only.

To find a list of available perspectives:

1. From the Window menu click **Open Perspectives**. (You can also find the Open Perspectives shortcut on the Shortcut bar located on the left side of the Workbench.)
2. Click **Other**.

The out-of-box list of perspectives includes the following:

- Administration
- Service Manager (default)

Implementing version control for your tailoring

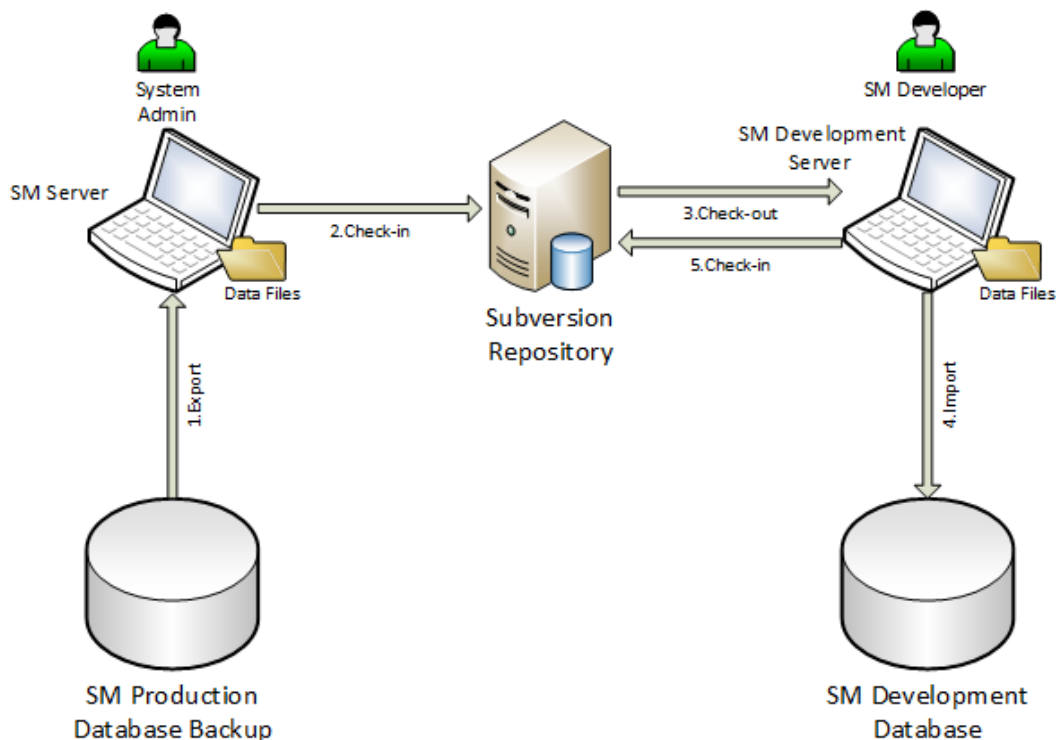
HP Service Manager provides comprehensive and fully-integrated IT service management through variously customized applications. Although Service Manager already provides an in-tool revision control based on the individual record, it has always been a challenge to implement version control for customization stacks. As of version 9.33, Service Manager can integrate with Apache Subversion (SVN) to provide a version control solution for your customized applications.

Note: This topic does not cover the basic concept of version control and the usage of Apache Subversion. Please refer to Apache Subversion homepage for details.

In short, it takes three steps to build up your version control process by using SVN:

1. A system administrator exports all the applications records from Service Manager and check in the exported files to an SVN server to create a baseline repository.
2. Developers check out the latest version of the applications records and import them into a local Service Manager development environment.
3. Developers customize the applications according to site-specific requirements, and check in the changes to the SVN server.

The following flow chart illustrates your version control process:



Version control process

The version control process includes the following:

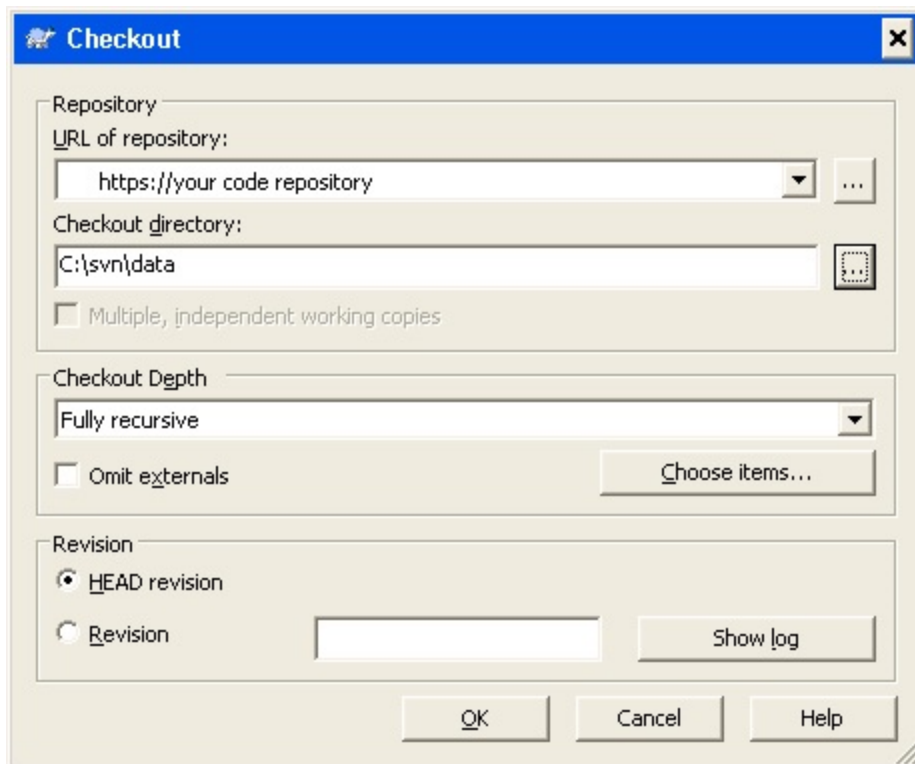
Code repository preparation by a system administrator

The first step of your version control process is that a system administrator creates a code repository on an SVN server. This enables you to use the repository to track all changes made to applications objects as you create, develop, or extend the HP Service Manager applications.

To leverage the SVN version control system, follow these steps:

1. Set up the SVN server and create the repository folder for the Service Manager applications data.
2. Set up the SVN client. To do this, follow these steps:
 - a. Create a folder in your local drive to check the SVN code repository in and out. For example, you create "c:\svn\data."

- b. Check out the empty repository to your local folder.



3. Configure Service Manager to use the repository. To do this, add the following two new lines to the `sm.ini` file in the `<Server Dir>/RUN/` directory:

```
svc_rootdir: <Your check-out Service Manager applications path, in this example, C:\svn\data>
```

```
svc_mode:2
```

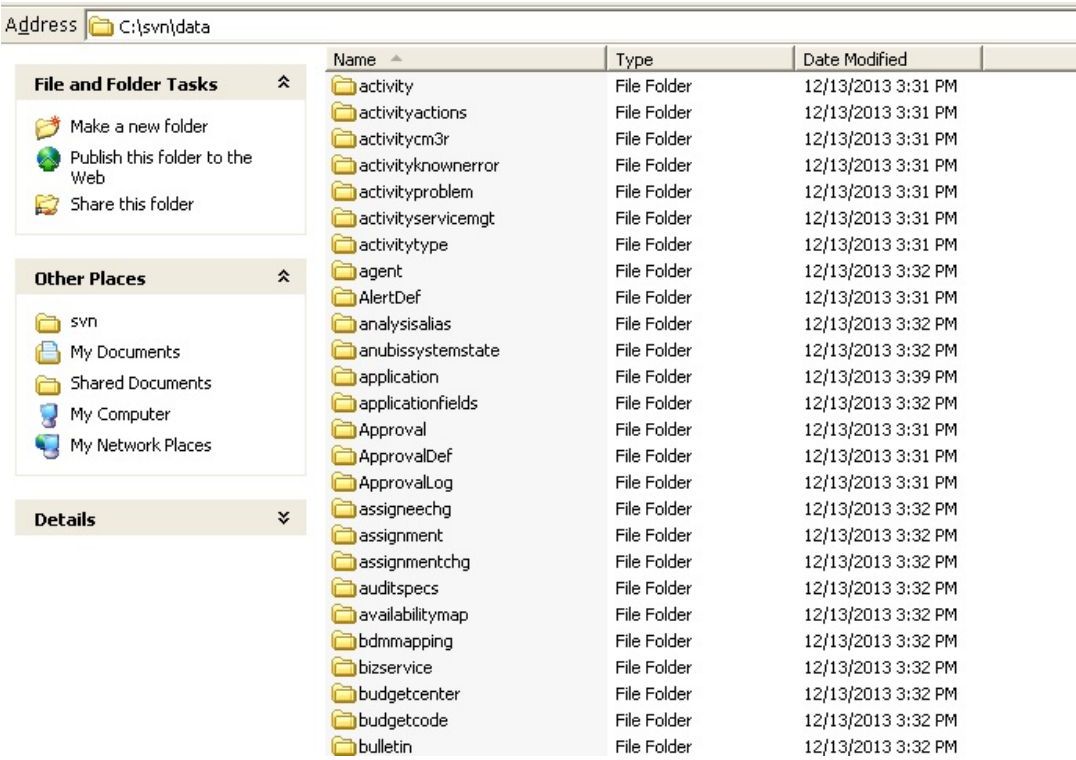
Note: Make sure that your Service Manager server is installed on the computer that has the SVN client installed.

4. Run the `sm -svc_export` command to export the Service Manager applications data as XML files to the `svc_rootdir` folder.

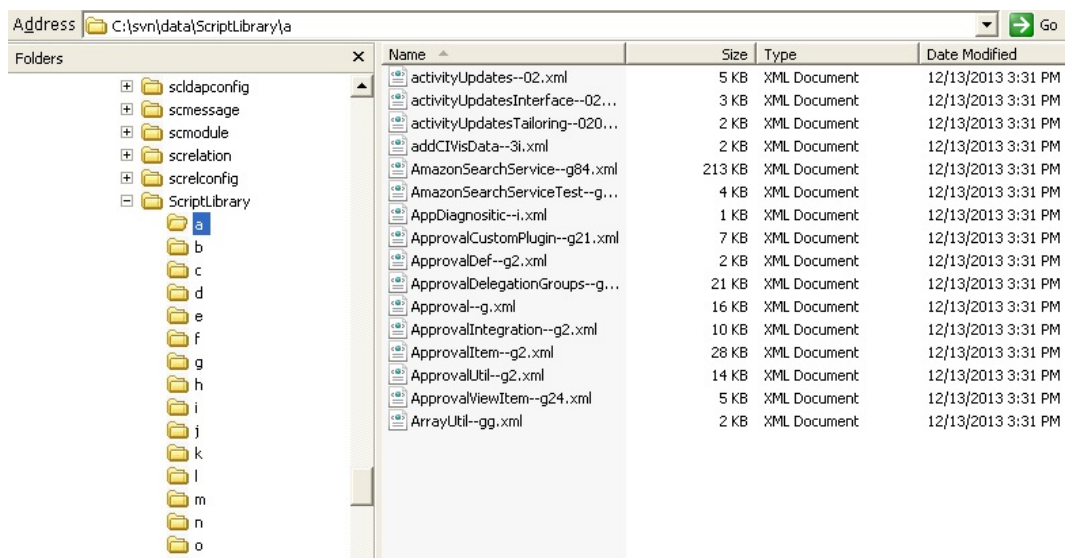
- Before you export the applications data records, be sure to stop the Service Manager server.

- For information about more parameters to export the applications data, see ["Parameters to import and export files"](#) on page 456.

Your output folder should resemble the following:



The XML files are exported to subfolders that are alphabetically-ordered by the unique key names of the applications, as shown in the following example for the ScriptLibrary folder:



5. Commit all the files to the SVN repository.

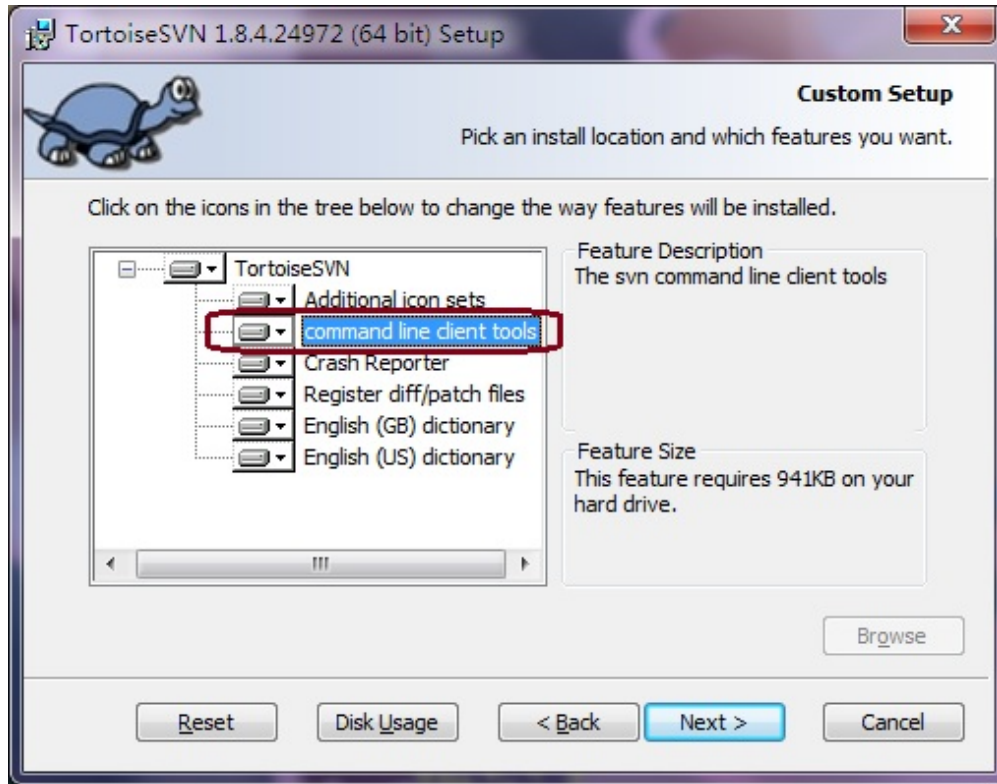
Note: You should exclude any business-related records when you commit the files, as these types of records require a great deal of capacity and decrease efficiency. These types of records include the following, but your organization’s records may differ:

Object	Table
Interaction	incidents
Incident	probsummary
Problem	rootcause
Problem Task	rootcausetask
Known Error	knownerror
Known Error Task	knownerrortask
Change	cm3r
Change Task	cm3t

Development environment preparation by developers

Note: Environment preparation is only required when a developer starts applications development for the first time or starts to use a new computer.

1. Set up your SVN client. To do this, choose the command line client tools in SVN. The following example displays the command line client tools feature in TortoiseSVN:



2. Create a folder in your local drive to check the SVN code repository in and out. For example, you create "c:\svn\data."
3. Set up Service Manager by adding the following two new lines to the `sm.ini` file in the `<Server Dir>/RUN/` directory:

```
svc_rootdir: <Your check-out Service Manager applications path, in this example, C:\svn\data>  
  
svc_mode:2
```

Note: Make sure that the Service Manager server is already installed on the same computer, and then restart the server to enable the added `svc` command.

4. Check out the applications data files from the SVN repository to your local folder.
5. Prepare a clean empty development database to import the applications data.

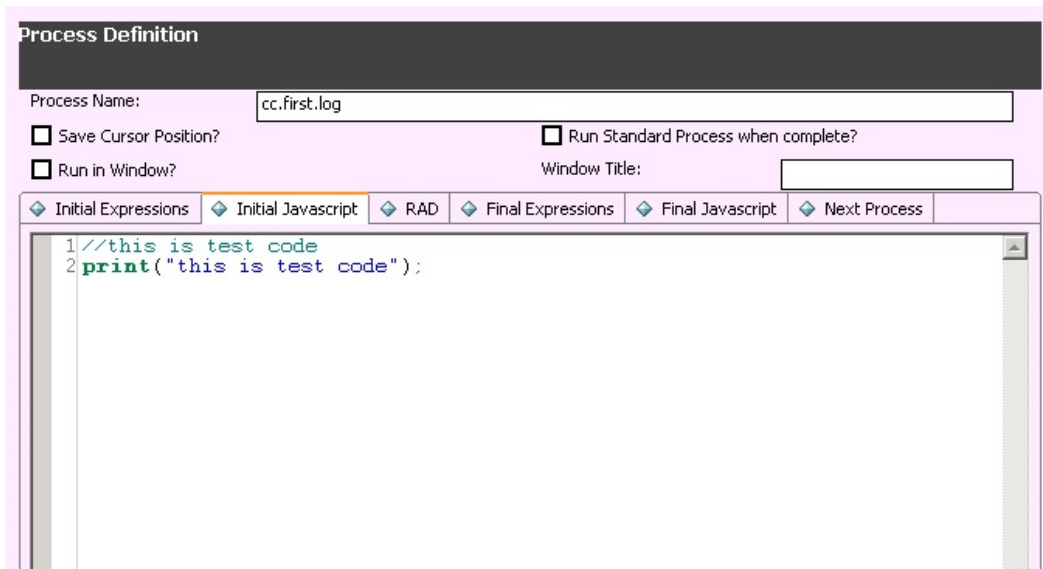
Development process by developers

1. Update all the source code to the latest version from the code repository.
2. Run the following command to import the XML files to the database:

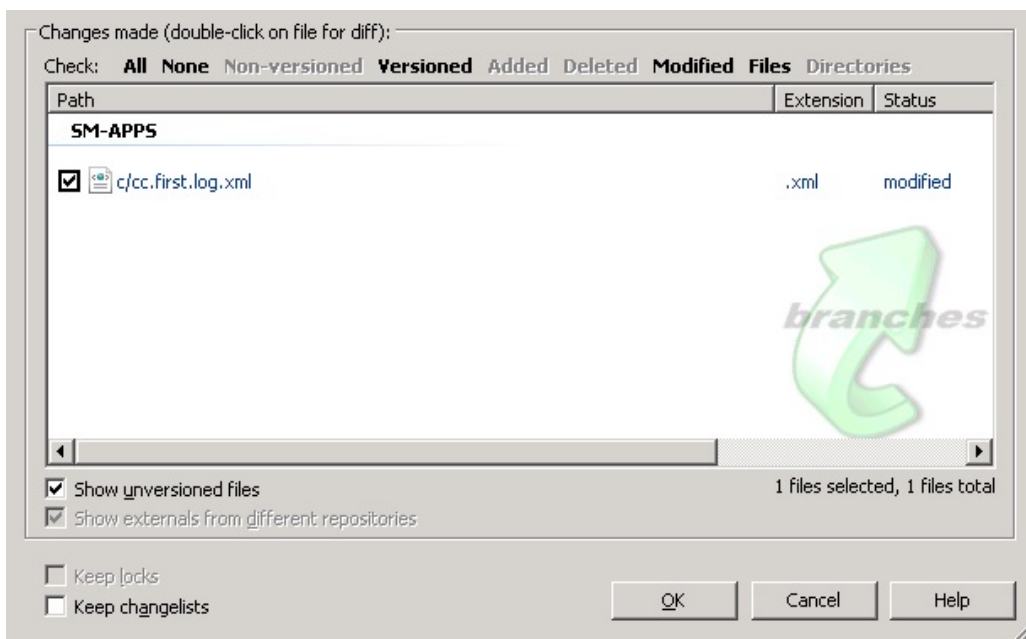
```
sm -svc_import -svc_cleanbuild:1
```

Note: For information about more parameters to import the XML files, see ["Parameters to import and export files" on page 456](#).

3. Modify the records by the Service Manager client and then save the changes to automatically synchronize the XML files in the `svc_rootdir` directory, as shown in the following example for the process record:



4. Check the modified code into the code repository. The commit window displays all the changed files.



Note: The commit window may not show all the changed files in the **SM-APPS** change list, and this problem occurs when the tables you have modified are not present in the `svc_include.props` property file in your `<Server Dir>/RUN` directory. To solve the problem, manually add the names of the tables to this file.

In this way, SVN retains a clear history of any changes made to a specific application.

Version control best practice

Create an operator record for each developer

Each developer should have their own operator record to log in to HP Service Manager rather than using the default System Administrator operator record. This helps to track code changes, because the value of the **sysmoduser** field of the `applicatiois` object is set as the currently logged-in user.

Streamline the code repository versions

Prior to any development, export the Service Manager 9.3x OOB database as the initial repository for future comparison and analysis. Then, export your organization's current, customized database as the

second-version repository.

Note: When you upgrade Service Manager to a later version and resolve all conflicts, you must export the complete upgraded database again as the new code repository.

Turn off this function when it is not needed

The source version control function decreases system performance. Therefore, you should uncomment the source version control configuration lines in the `sm.ini` file when it is not needed.

Applications data file represented in XML

Exporting applications from HP Service Manager converts the applications records to data files represented in XML, with the applications object name in the `<recordset table>` element and the unique id in the `<record id>` element. The following code is an example of the *Process* record:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<recordset table="Process">
  <record id="process="&quot;cc.first.log&quot;;" recordid="cc.first.log">
    <process type="string">cc.first.log/process</process>
    <pre.expressions sctype="array">
      <pre.expressions sctype="operator">$L.oldinc=$L.lastinc</pre.expressions>
      <pre.expressions sctype="operator">if ($L.action="lastincident") then
($L.golast="true";$L.action="log") else ($L.golast="false")</pre.expressions>
      <pre.expressions sctype="operator">$L.continue=true</pre.expressions>
      <pre.expressions sctype="operator">if ($L.action="logcatalog" and null(title
in $L.file)) then (title in $L.file=1 in description in $L.file)</pre.expressions>
    </pre.expressions>
    <post.expressions sctype="array">
      <post.expressions sctype="operator">if ($G.ess and $L.continue) then
($L.exit.when.done=true)</post.expressions>
    </post.expressions>
    <rad sctype="array">
      <rad sctype="structure">
        <application type="string">sc.get.sla</application>
        <names sctype="array">
          <names sctype="string">file</names>
          <names sctype="string">number1</names>
          <names sctype="string">numbers</names>
        </names>
        <values sctype="array">
          <values sctype="string">$L.file</values>
          <values sctype="string">contract.id in $L.file</values>
          <values sctype="string">agreement.ids in $L.file</values>
        </values>
      </rad>
    </rad>
  </record>
</recordset>
```

```

    </values>
    <rad.condition type="operator">enable in $G.sla.environment and
($L.action="complete" or $L.action="log" or $L.action="logcatalog") and
$L.mode="add"</rad.condition>
    <pre.rad.expressions sctype="array">
    <pre.rad.expressions sctype="operator">if ($affected.ci~="Other - See
Description") then (affected.item in $L.file=nullsub(affected.item in $L.file,
$affected.ci))</pre.rad.expressions>
    </pre.rad.expressions>
    <post.rad.expressions sctype="string"/>
</rad>
<rad sctype="structure">
  <application type="string">sla.check.sh.wrapper</application>
  <names sctype="array">
    <names sctype="string">number1</names>
    <names sctype="string">text</names>
  </names>
  <values sctype="array">
    <values sctype="string">1 in agreement.ids in $L.file</values>
    <values sctype="string">$L.exit</values>
  </values>
  <rad.condition type="operator">not (null(callback.contact in $L.file)) and
not (null(agreement.ids in $L.file)) and check.sh in $G.sla.environment and
(nullsub($L.overridden, false)=false or not (same($L.or.id, 1 in agreement.ids in
$L.file)))</rad.condition>
  <pre.rad.expressions sctype="array">
    <pre.rad.expressions sctype="operator">if (null(agreement.ids in $L.file)
and not (null(agreement.id in $L.file))) then (1 in agreement.ids in
$L.file=agreement.id in $L.file)</pre.rad.expressions>
  </pre.rad.expressions>
  <post.rad.expressions sctype="array">
    <post.rad.expressions sctype="operator">if ($L.exit="normal") then
($L.overridden=true;$L.or.id=1 in agreement.ids in $L.file)</post.rad.expressions>
    <post.rad.expressions sctype="operator">if ($L.exit="blocked") then
($L.string=scmsg(292, "pm", {nullsub(callback.contact in $L.file, "NULL"), nullsub
(1 in agreement.ids in $L.file,
"NULL")});$L.overridden=false;$L.or.id=NULL;$L.action="complete";resolution in
$L.file=resolution in $L.file+{$L.string};resolution.code in $L.file="Out of
Scope")</post.rad.expressions>
  </post.rad.expressions>
</rad>
<rad sctype="structure">
  <application type="string">getnumb</application>
  <names sctype="array">
    <names sctype="string">name</names>
    <names sctype="string">index</names>
    <names sctype="string">text</names>
  </names>
  <values sctype="array">

```

```

    <values sctype="string">number.record in $L.object</values>
    <values sctype="string">incident.id in $L.file</values>
    <values sctype="string">"string"</values>
  </values>
  <rad.condition type="operator">>null(incident.id in $L.file)</rad.condition>
  <pre.rad.expressions sctype="array">
    <pre.rad.expressions sctype="operator">if ($L.action={"log", "logexit"}
and open in $L.file~="Closed") then (first.call in $L.file=false)
</pre.rad.expressions>
    <pre.rad.expressions sctype="operator">if ($L.action={"associate",
"incident.change", "incident.request", "createchange", "createrequest", "create"})
then (first.call in $L.file=false;$L.notify.name="SM Add")</pre.rad.expressions>
    <pre.rad.expressions sctype="operator">if ($L.action="complete") then
($L.oldstatus=open in $L.file;open in $L.file="Closed";$L.notify.name="SM Close")
</pre.rad.expressions>
    <pre.rad.expressions sctype="operator">if ($L.action={"log", "logexit",
"save", "newcall"}) then (open in $L.file=nullsub(open in $L.file, "Open -
Callback");if ($L.bg=false) then ($L.notify.name="SM Add"))</pre.rad.expressions>
  </pre.rad.expressions>
  <post.rad.expressions sctype="string"/>
</rad>
<rad sctype="structure">
  <application type="string">us.consume.wrapper</application>
  <names sctype="array">
    <names sctype="string">index</names>
    <names sctype="string">name</names>
    <names sctype="string">boolean1</names>
    <names sctype="string">text</names>
  </names>
  <values sctype="array">
    <values sctype="string">contract.id in $L.file</values>
    <values sctype="string">"i"</values>
    <values sctype="string">$L.bg</values>
    <values sctype="string">$L.exit</values>
  </values>
  <rad.condition type="operator">enable in $G.cm.control and not (nullsub
(contract.consumed in $L.file, false)) and not (null(contract.id in $L.file))
</rad.condition>
  <pre.rad.expressions sctype="array">
    <pre.rad.expressions sctype="operator">if (null($G.src) or $G.src=false)
then ($L.exit="normal")</pre.rad.expressions>
  </pre.rad.expressions>
  <post.rad.expressions sctype="array">
    <post.rad.expressions sctype="operator">if ($L.exit="badval") then if
($L.action="complete") then (open in $L.file=$L.oldstatus)</post.rad.expressions>
    <post.rad.expressions sctype="operator">if ($L.exit="normal") then
(contract.consumed in $L.file=true)</post.rad.expressions>
  </post.rad.expressions>
</rad>

```

```

<rad sctype="structure">
  <application type="string">cc.save</application>
  <names sctype="array">
    <names sctype="string">file</names>
    <names sctype="string">name</names>
    <names sctype="string">record</names>
    <names sctype="string">boolean1</names>
    <names sctype="string">text</names>
  </names>
  <values sctype="array">
    <values sctype="string">${L.file}</values>
    <values sctype="string">${L.init.action}</values>
    <values sctype="string">${L.fc}</values>
    <values sctype="string">${L.bg}</values>
    <values sctype="string">${L.exit}</values>
  </values>
  <rad.condition type="operator">${L.exit}="normal"</rad.condition>
  <pre.rad.expressions sctype="array">
    <pre.rad.expressions
sctype="operator">${L.handle.save}=${L.handle.start}</pre.rad.expressions>
    <pre.rad.expressions sctype="operator">if (${L.handle.start}~0) then
(handle.time in ${L.file}=nullsub(handle.time in ${L.file}, '00:00:00');handle.time in
${L.file}=(handle.time in ${L.file}+tod()) - ${L.handle.start};${L.handle.start}=0)
</pre.rad.expressions>
    <pre.rad.expressions sctype="operator">${L.init.action}=nullsub
(${L.init.action}, "add")</pre.rad.expressions>
    <pre.rad.expressions sctype="operator">${L.pri.calc}=jscall
("PriorityCalc.getPriorityCalc")</pre.rad.expressions>
    </pre.rad.expressions>
    <post.rad.expressions sctype="array">
    <post.rad.expressions sctype="operator">if (${L.exit}="normal") then if
(${L.golast}="false") then (${L.lastinc}=incident.id in ${L.file})</post.rad.expressions>
    <post.rad.expressions sctype="operator">if (${L.exit}="badval") then
(${L.handle.start}=${L.handle.save};${L.continue}=false;if (${L.action}="complete") then
(open in ${L.file}=${L.oldstatus}))</post.rad.expressions>
    <post.rad.expressions sctype="operator">if (${L.exit}="normal" and ${G.ess})
then (${L.file.save}=${L.file})</post.rad.expressions>
    <post.rad.expressions sctype="operator">if (${L.exit}="normal" and not
(${G.ess}) and blank.call in ${G.sm.global.environment}) then (${L.format}=nullsub
(edit.format in ${G.sm.environment}, "SD.update.interaction"))</post.rad.expressions>
    </post.rad.expressions>
  </rad>
<rad sctype="structure">
  <application type="string">us.save.relation</application>
  <names sctype="array">
    <names sctype="string">name</names>
    <names sctype="string">prompt</names>
    <names sctype="string">query</names>
    <names sctype="string">string1</names>

```

```

</names>
<values sctype="array">
  <values sctype="string">$L.related.id</values>
  <values sctype="string">$L.related.filename</values>
  <values sctype="string">$L.depend.id</values>
  <values sctype="string">$L.depend.filename</values>
</values>
<rad.condition type="operator">not (null($L.related.id)) and not (null
($L.related.filename)) and $L.continue</rad.condition>
<pre.rad.expressions sctype="array">
  <pre.rad.expressions
sctype="operator">$L.depend.filename="incidents"</pre.rad.expressions>
  <pre.rad.expressions sctype="operator">$L.depend.id=str(incident.id in
$L.file)</pre.rad.expressions>
</pre.rad.expressions>
<post.rad.expressions sctype="array">
  <post.rad.expressions sctype="operator">if $L.continue then if (not
(null($L.related.id)) and not (null($L.related.filename))) then
($L.exit="normal";$L.exit.when.done=true)</post.rad.expressions>
  <post.rad.expressions sctype="operator">if $L.continue then
($L.related.id=NULL;$L.related.filename=NULL)</post.rad.expressions>
</post.rad.expressions>
</rad>
<rad sctype="structure">
  <application type="string">us.notify</application>
  <names sctype="array">
    <names sctype="string">name</names>
    <names sctype="string">record</names>
    <names sctype="string">second.file</names>
  </names>
  <values sctype="array">
    <values sctype="string">$L.notify.name</values>
    <values sctype="string">$L.file</values>
    <values sctype="string">$L.file.save</values>
  </values>
  <rad.condition type="operator">$L.exit="normal"</rad.condition>
  <pre.rad.expressions sctype="string"/>
  <post.rad.expressions sctype="array">
    <post.rad.expressions sctype="operator">if ($L.exit="normal" and not
($L.exit.when.done)) then if blank.call in $G.sm.global.environment then
($L.exit="restart";$L.void=rtecall("resetnotebook", $L.rc)) else
($L.exit.when.done=true)</post.rad.expressions>
    <post.rad.expressions sctype="operator">if ($L.continue and
$L.action="save" and not ($L.exit.when.done)) then ($L.exit="";$L.mode="browse")
</post.rad.expressions>
    <post.rad.expressions sctype="operator">if ($L.action="addcloseme") then
($L.exit="";$L.mode="browse")</post.rad.expressions>
    <post.rad.expressions sctype="operator">if ($L.continue and
$L.pmtapi=true) then ($L.exit="exit")</post.rad.expressions>

```

```

    </post.rad.expressions>
</rad>
<rad sctype="structure">
  <application type="string">cc.set.approval</application>
  <names sctype="array">
    <names sctype="string">file</names>
    <names sctype="string">record</names>
  </names>
  <values sctype="array">
    <values sctype="string">$L.file</values>
    <values sctype="string">$L.object</values>
  </values>
  <rad.condition type="boolean">>false</rad.condition>
  <pre.rad.expressions sctype="string"/>
  <post.rad.expressions sctype="string"/>
</rad>
</rad>
<next.process sctype="array">
  <next.process sctype="string">sm.close</next.process>
  <next.process sctype="string">cc.clone.relation</next.process>
  <next.process sctype="string">cc.screlate</next.process>
  <next.process sctype="string">sm.save</next.process>
</next.process>
<process.condition sctype="array">
  <process.condition sctype="operator">$L.continue and
$L.action="addcloseme"</process.condition>
  <process.condition sctype="operator">$L.continue and ($L.action="save" or
$L.mode="addclone") and not ($G.ess)</process.condition>
  <process.condition sctype="operator">$L.continue and
$L.action="complete"</process.condition>
  <process.condition sctype="operator">$G.ess and
$L.mode="addclone"</process.condition>
</process.condition>
<run.standard NullValue="1" type="boolean"/>
<sysmodcount type="decimal">9</sysmodcount>
<sysmoduser type="string">zhangqi</sysmoduser>
<sysmodtime type="dateTime">11/22/13 00:48:44</sysmodtime>
<save.cursor.position NullValue="1" type="boolean"/>
<run.in.window NullValue="1" type="boolean"/>
<window.name NullValue="1" type="string"/>
<javascript.pre NullValue="1" type="string"/>
<javascript.post type="string">if (vars.$G_src & & vars.$L_
exit=="blocked")
{
  system.functions.msg(vars.$L_string, 3);
}</javascript.post>
  <baseMethod NullValue="1" type="string"/>
</record>
</recordset>

```

Parameters to import and export files

The version control command line tool provides you the following parameters to import and export specific files. In Service Manager, a file is equivalent to a Relational Database Management System (RDBMS) table. For more information, see ["File" on page 1](#).

Parameters to export files

To export Service Manager files, you can run the `sm -svc_export` command with the following parameters, or add the following parameters in the `sm.ini` file.

Parameter	Description
<code>svc_files</code>	Specifies a list of files to export. Use commas to separate multiple files (no space before or after a comma). Service Manager exports all files in the system if the parameter is not set.
<code>svc_excludefiles</code>	Specifies a list of files that you want to exclude from the export operation. Use commas to separate multiple files (no space before or after a comma).

For example, if you want to export ScriptLibrary and formatctrl data only, add the following line in `sm.ini`:

```
svc_files:ScriptLibrary,formatctrl
```

Parameters to import files

To import Service Manager files, you can run the `sm -svc_import` command with the following parameters, or add the following parameters in the `sm.ini` file.

Parameter	Description
<code>svc_cleanbuild</code>	If the value of this parameter is set to 1 , all the files in the <code>svc_rootdir</code> directory are imported; if the value of this parameter is set to 0 or not set, the list of files to import is determined by the svc_files parameter.
<code>svc_files</code>	Specifies a list of files to import. Use commas to separate multiple files (no space before or after a comma). Note: This parameter works only if the svc_cleanbuild parameter is set to 0 or not set.
<code>svc_excludefiles</code>	Specifies the list of files that is excluded in the importing operation. Use commas to separate multiple files (no space before or after a comma).

For example, if you want to import only records in the ScriptLibrary and formatctrl folders under the svc_root directory, add the following lines in *sm.ini*:

```
svc_cleanbuild:0
```

```
svc_files:ScriptLibrary,formatctrl
```

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 Tailoring help topics for printing (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!

