

HP Defects and Requirements Exchange with HP Service Manager and HP Quality Center

Software Version: 1.02

Installation and Administration Guide

Document Release Date: March 2009

Software Release Date: March 2009



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 Notices

© Copyright 2008-2009 Hewlett-Packard Development Company, L.P.

Trademark Notices

AMD and the AMD logo are trademarks of Advanced Micro Devices, Inc.

Intel and Pentium are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

JAVA™ is a US trademark of Sun Microsystems, Inc.

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

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

UNIX® is a registered trademark of The Open Group.

Documentation Updates

This guide's title page contains the following identifying information:

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

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

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

This site requires that you register for an HP Passport and sign-in. To register for an HP Passport ID, go to:

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

Or click the **New users - please register** link 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

You can visit the HP Software Support Online web site at:

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

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

HP Software Support Online 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 HP Software Support web site to:

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

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

To find more information about access levels, go to:

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

To register for an HP Passport ID, go to:

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

Contents

1	Introduction	9
	Document Purpose	9
	Synchronization Concepts	9
	SM Change -> QC Defect	10
	SM Change -> QC Requirement	10
	SM Problem -> QC Defect	10
	QC Defect -> SM Problem	10
	SM Problem <-> QC Defect	10
	References	10
2	Planning Deployment	13
	Supported Products and Platforms	13
	Deployment Scenarios	13
	Data Types	14
	Deployment Tasks	14
	Release Package	14
3	Installing/Configuring QC Synchronizer	17
	Installing QCS	17
	Deploying Adapters	18
	Generating/Deploying Stub	18
	Copying SM Adapter Configuration Files	19
	Editing the Configuration Files	19
	Module Types	19
	Field Types	19
	Requirements	20
	SM Field Type and Definition Rule	21
	SM Change Management Example	21
	SM Problem Management Example	22
4	Configuring Links in Quality Center Synchronizer	25
	Create a Link	26
	Advanced Parameters	28
	QC Field <-> SM Field mappings	29
	Create Mapping	29
	General Mapping Requirements	29
	Matching Types	30
	List Value Mappings	31
	Constant -> SM field Mappings	32

5	Integration Account	33
	Creating a SM Integration Account	33
	Contact	33
	Profile Records	33
	Profile for Change Management	33
	Profile for Problem Management	34
	Operator	35
	Creating a QC Integration Account	36
	Create User	36
	Create Group	36
	Assign Permission	37
6	SM Change -> QC Defect	39
	Customizing Service Manager/ServiceCenter for Change Management	39
	Add Fields	39
	Specify Field External Access	40
	On Service Manager 7.0x/7.10	40
	On ServiceCenter	41
	Create Subform	42
	Add Subform to Form	43
	Add Format Control Calculations/Validations	44
	Customizing Quality Center Defects Module	45
	Add Fields	46
	Add Tabs	46
	Add Fields to Tabs	47
	Verify	48
	Configuring Links in QC Synchronizer	49
	Specify Endpoints / Type of Link	49
	Field Mappings	50
	QC Field <-> SM Field	50
	Events	52
	Test	53
7	SM Change -> QC Requirement	55
	Customizing Service Manager/ServiceCenter for Change Management	55
	Customizing Quality Center Requirements Module	55
	Add Fields	55
	Add Tabs	56
	Add Fields to Tabs	57
	Create Folder "SM Incoming Changes"	58
	Configuring Links in QC Synchronizer	59
	Specify Endpoints / Type of Link	59
	Field mappings	60
	QC Field <-> SM Field	61
	Events	61
	Test	62

8 SM Problem -> QC Defect	65
Customizing Service Manager/ServiceCenter for Problem Management	65
Add Fields	65
Specify Field External Access	66
On Service Manager 7.0x/7.10	66
On ServiceCenter	68
Create Subform	70
Add Subform to Form	71
Add Format Control Calculations/Validations	72
Customizing Quality Center Defects Module	73
Add Fields	73
Add Tabs	74
Add Fields to Tabs	75
Configuring Links in QC Synchronizer	76
Specify Endpoints / Type of Link	76
Field Mappings	76
Events	77
Test	77
9 QC Defect -> SM Problem	81
Customizing Service Manager/ServiceCenter for Problem Management	81
Add Fields	81
Specify Field External Access	82
On Service Manager 7.0x/7.10	82
On ServiceCenter	84
Create Subform	86
Add Subform to Form	86
Customizing Quality Center Defects Module	87
Add Fields	87
Add Tabs	88
Add Fields to Tabs	89
Create a View	90
Verify	91
Configuring Links in QC Synchronizer	91
Specify Endpoints / Type of Link	91
Filters	91
Field Mappings	92
QC Field <-> SM Field	93
Constants -> SM Fields	93
Events	94
Test	94
10 SM Problem <-> QC Defect	97
Customizing Service Manager/ServiceCenter for Problem Management	97
Add Fields	97
Specify Field External Access	98
On Service Manager 7.0x/7.10	98

On ServiceCenter	100
Create Subform	102
Add Subform to Form	103
Add Format Control Calculations/Validations	104
Customizing Quality Center Defects Module	106
Add Fields	106
Add Tabs	106
Add Fields to Tabs	107
Create a View	108
Verify	109
Configuring Links in QC Synchronizer	109
Specify Endpoints / Type of Link	109
Filters	109
Field Mappings	110
Events	111
Test	112
11 Upgrade	117
Upgrading to the Latest Release.	117
Backup Jar Files and Links	117
Deploy the Latest Adapters.	117
Upgrade User Stories	118
SM Change -> QC Defect	118
SM Change -> QC Requirement	118
QC Defect -> SM Problem	119
Post-Upgrade	121
A Error Messages.	123
Required Fields	123
Installation	124
Configuration	125
Runtime.	127
XML Validation.	129

1 Introduction

This introduction describes

- Document Purpose
- Synchronization Concepts
- References

Document Purpose

This document describes how to configure and deploy the integration components

- HP Service Manager / HP ServiceCenter (SM)
- HP Quality Center (QC)
- HP Quality Center Synchronizer (QCS)

This document then describes how to configure and test synchronization links between QC and SM.

- ▶ This document contains numerous examples that use the SM and QC default installation configuration and databases. Your particular configuration may differ significantly. The example synchronization configuration may also differ significantly from your requirements.

The target readers include HP Consultants and/or Application Administrators who must set up and maintain QCS, ensuring that QCS meets all user organization procedural requirements. This document assumes that the reader is an experienced user of either (but not both) SM or QC, and therefore describes the basics of both SM and QC.

Synchronization Concepts

This section provides a detailed introduction to basic synchronization concepts.

- SM Change -> QC Defect
- SM Change -> QC Requirement
- SM Problem -> QC Defect
- QC Defect -> SM Problem
- SM Problem <-> QC Defect

SM Change -> QC Defect

When a business owner enters a change request in SM and marks the change “Forward as defect”, a defect is created in QC. This informs the QA personnel that they should begin the QA process.

During the QA process, key information is synchronized from QC to SM. The integration administrator has the responsibility of determining the key information and specifying the information in the field mapping (using the provided integration tool) in order for the business owner to view updated (scheduled) information in the SM. The information includes the status of all changes in the testing cycle.

SM Change -> QC Requirement

The requirement synchronization features of SM and QC integration allow requirements found during the change management process to be systematically tracked by SM and QC.

When a business owner enters a change request in SM and marks it as “Forward as requirement”, a requirement is created in QC. This informs the QA personnel that they should begin the QA process.

During the QA process, key information is synchronized from QC to SM. The integration administrator has the responsibility of determining the key information and specifying the information in the field mapping (using the provided integration tool). This allows the business owner to view updated (scheduled) information in SM.

SM Problem -> QC Defect

After a problem is created, if the CPE engineer determines that there is bug with the problem after analyzing it, and the bug fixing work needs to be tracked, the CPE engineer triggers/initiates the creation of the QC CR ticket. When this problem is marked as "Synchronize with QC Defect", a defect is created in QC.

QC Defect -> SM Problem

The business process for defect management in QC supports creation of known errors in SM based on information in QC. However, in the current solution, the integration can only create a problem in SM from a defect in QC. A user must create the known error in SM manually from the problem in SM. Known errors are a source of information for informal knowledge articles in the Knowledge Base.

SM Problem <-> QC Defect

This user story is combination of [SM Problem -> QC Defect](#) and [QC Defect -> SM Problem](#).

References

- 1 *HP Quality Center Synchronizer User's Guide*
- 2 *HP Quality Center Administrator's Guide*

- 3 *HP Service Manager Installation Guide*
- 4 *HP Service Manager Online Help*
- 5 *Best Practices for Publishing and Consuming Web Services with ServiceCenter*

2 Planning Deployment

This chapter describes deployment planning

- Supported Products and Platforms
- Deployment Scenarios
- Data Types
- Deployment Tasks
- Release Package

Supported Products and Platforms

Supported products are shown in the following table.

Supported Product	Version
HP Quality Center Synchronizer	1.2, 1.3
HP Quality Center	9.2 patch 4 and above; 10
HP Service Manager	7.01, 7.02, 7.10
HP ServiceCenter	6.2.2 and above

In this integration solution, supported platforms for Quality Center Synchronizer include:

- Microsoft Windows 2000 with Service Pack 4 (32bit)
- Microsoft Windows XP with Service Pack 2 (32bit)
- Microsoft Windows 2003 Server with Service Pack 2 (32bit)

For Service Manager, supported platforms follow Service Manager's product support matrix.

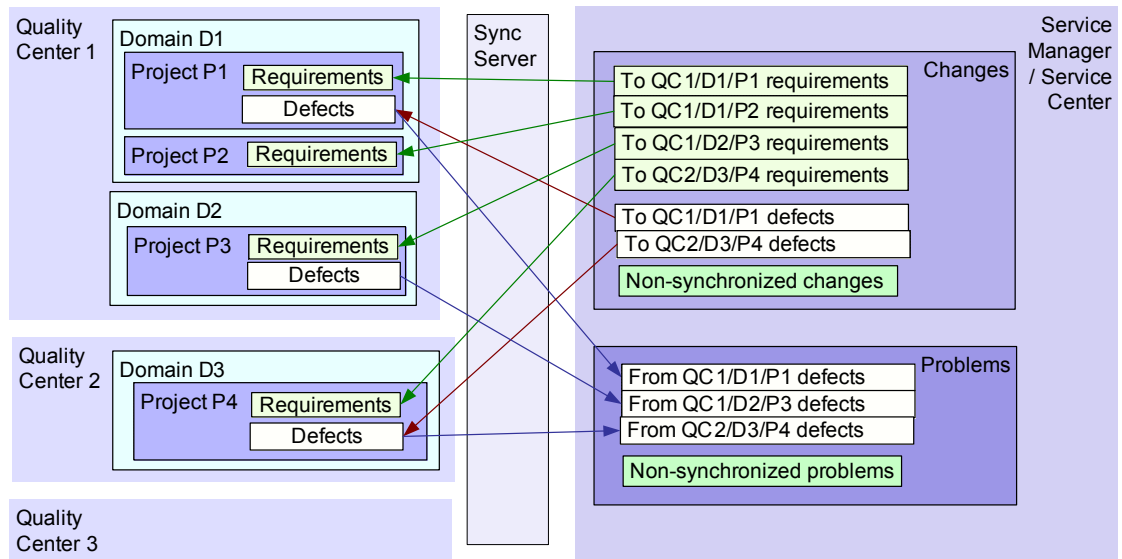
For Quality Center, supported platforms follow Quality Center's product support matrix.

Deployment Scenarios

The following are the deployment scenarios:

- A single SM server has a dedicated synchronizer.
- A single SM server can connect to multiple QC's.

This is shown in the following diagram.



Data Types

[Matching Types](#) on page 30 describes the data type requirements between QC, QCS, and SM.

Deployment Tasks

Deployment tasks include:

- Customizing Service Manager/Service Center
- Customizing Quality Center
- Installing/Configuring QC Synchronizer
- Configuring Links in Quality Center Synchronizer

Release Package

The release package is delivered as an executable self-extracting installer. Run the installer by double clicking it. The major contents are shown in the following table.

Directory	Contents
sm-adapter\adapter	Adapter and dependencies (except stub)
sm-adapter\ant	Build lib
sm-adapter\bin	Script to generate the stub
sm-adapter\doc	Release documents (including this document)

Directory	Contents
sm-adapter\jdk5	Sun JDK 1.5
sm-adapter\lib	Binary libraries required to generate the stub
sm-adapter\sample	Examples of WSDL and adapter configuration
sm-adapter\out-of-box	Out-of-box demo package

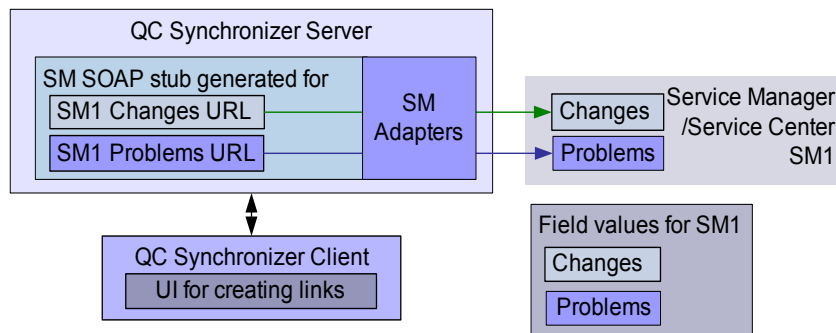
3 Installing/Configuring QC Synchronizer

The QC Synchronizer (QCS) allows centralized management of a set of tightly coupled one-to-one data synchronization links and provides an open and extensible platform for the development of new data synchronization adapters to entity repositories.

To install and configure the synchronizer, follow these steps.

- 1 Installing QCS
- 2 Deploying Adapters
- 3 Generating/Deploying Stub
- 4 Copying SM Adapter Configuration Files
- 5 Editing the Configuration Files

The following diagram shows an overview of the installation process.



Installing QCS

QCS is available from

<http://updates.merc-int.com/qualitycenter/qc90/sync/qcsynchronizer/index.html>

- ▶ • QC client is installed when you are logging into QC. The correct QC client should be installed on QCS server.
- QCS machine should have the same time zone with QC machine. See [QC must be in the same time zone as QCS](#) on page 125 for more information. Refer to the QC Synchronizer user guide for installation instructions (QCSyncUG.pdf). You can get the user guide from the installation package or from http://ovweb.external.hp.com/lpe/doc_serv/.



Make sure that the time difference in UTC between SM and QCS is within 5 minutes, otherwise the data might be lost during synchronization.

For example, SM server time is 2008-1-1 21:00:00 in UTC, then QCS server time must be between 2008-1-1 20:55:00 and 2008-1-1 21:05:00.

Deploying Adapters

Copy all files under `[release-package]\adapter` directory to `<QCS_Install_Dir>\adapters\lib` directory. Adapters include:

```
sm-adapter-XX.XX.XXX.jar (XX.XX.XXX is the the version number for the current
release)
sm-adapter-axis-1.4.jar
sm-adapter-commons-discovery-0.2.jar
sm-adapter-commons-lang-2.3.jar
sm-adapter-jaxrpc-1.1.jar
sm-adapter-jdom-1.1.jar
sm-adapter-saaj-1.2.jar
sm-adapter-wsdl4j-1.5.1.jar
sm-adapter-commons-codec-1.3.jar
sm-adapter-commons-httpclient-3.1.jar
```

Generating/Deploying Stub

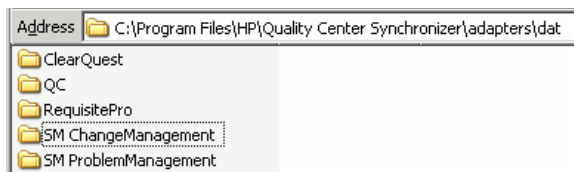
- 1 Start the SM service (stub generation requires access to SM).
- 2 Edit the following lines in `[release-package]\bin\build.properties` as required for access to SM:

```
#Set up WSDL URL, please change the URL to your actual SM server, eg,
http://<your-server>:<port>/.../<service-name>.wsdl
#Comment this line by this sign "#" if you do not generate stub jar for
change management module
sm.change.wsdl=http://localhost:13080/sc62server/PWS/
QCIntChangeService.wsdl

#Comment this line by this sign "#" if you do not generate stub jar for
problem management module
sm.problem.wsdl=http://localhost:13080/sc62server/PWS/
QCIntProblemService.wsdl
```
- 3 Run the script `build.bat` from command line (check the console output for errors). The stub `[release-package]\build\sm-adapter-ws-client.jar` is generated.
- 4 Copy the stub to `<QCS_Install_Dir>\adapters\lib` directory.

Copying SM Adapter Configuration Files

- 1 Start/restart QCS, go to **Start** → **All Programs** → **HP Quality Center Synchronizer** → **Start/Stop Synchronizer**. The directories `<QCS_Install_Dir>\adapters\dat\SM ChangeManagement` and `SM ProblemManagement` appear after the synchronizer service starts (this can take up to a minute).



- 2 Copy the file `[release-package]\sample\configuration_file_default.xml` to the following folders
 - SM ChangeManagement
 - SM ProblemManagement

Editing the Configuration Files

Edit the files as described below. The files will later be specified when creating links.

Module Types

There are two module types for this configuration file: `change` or `problem`.

If the module name is `change`, it means that this module is for Change Management; if it is `problem`, it means that this module is for Problem Management.

For example:

```
<itg:module name="change">
```

You can define one module or two in this file. But duplicate definition is not permitted.

Field Types

Table 1 Field XML Element Specification

Parameter	Description
name	Field name. This name should be the same as Caption enabled in SM/SC WSDL. This field is required.
type	Field type. Its value can be "String"/"Number"/"Date"/"Single_Value_List"/"Multi_Value_List". This field is required.

Table 1 Field XML Element Specification

Parameter	Description
readonly	Indicates whether the field is read-only. Its value can be "true" or "false". This field is optional. Default is "false".
required	Indicates whether the field is mandatory, recommended or optional. This field is optional. Default value for the field is optional.
length	The length of the field in SM endpoint. This field is optional. The length is unlimited if not specified.

The configuration file is an XML file that provides Change/Problem field values to the SM adapter. These values include

- Field name (the caption of a field in SM WSDL configuration form, such as Status, Priority)
- Field types
 - String
 - Number
 - Date
 - Single_Value_List
 - Multi_Value_List
- List types
 - Array (multi-value list)
 - Single-value list
- For a value list, the mapping of the value in the database and the exposed caption (for a type other than a value list type, the adapter automatically determines the desired data type).

Requirements

- Default field configuration is readable and writable with unlimited length.
- Default field configuration for a Single_Value_List or a Multi-Value_List must be explicitly specified.
- Read-only field must be explicitly specified.
- You must specify the type and read/write explicitly only for a Single_Value_List/ Multi_Value_List.
- If a field is not configured, then the field is read/write with unlimited length.
- A list or multi-list field may contain item elements. For each item specify the value and display text in the form `<itg:item value="$value">$display text</itg:item>`.
- If the field in WSDL is an Array then it must be mapped to Multi_Value_List.
- If the QC field is User_List, then you can only specify String or Single_Value_List for the corresponding SM field.

- If the field is read only you must set the attribute `readonly` as `true`.
- If the field attribute `required` is `mandatory`, then the field is mandatory for creation of a new entity.
- If the field has a length limitation (attribute `length`). Values from other endpoints could be truncated to match this limitation.

SM Field Type and Definition Rule

There are restrictions on data type and field type combination. Define field type in the configuration file according to such rules.

SM/SC DB data type		Field type on Form	WSDL data type	Field Definition Rule
SM 7.0x/7.10	SC 6.2			
Date/time	Date/time	Date	DateTimeType	Optional. Permitted type is "Date".
Number	Decimal	Decimal Text	<Empty> DecimalType IntType	Optional. Permitted type is "Number".
Logical	Boolean	Check Box Radio Button	<Empty> BooleanType	Optional. Permitted type is "String".
Character	Text	Text TextArea Combo Box Comfill	<Empty> StringType	Optional. Permitted types include "String" and "Single_Value_List".
Array	Array	Text Area	<Empty>	Optional. Permitted types include "String" and "Single_Value_List".
Array	Array	Combo Box Comfill Text	<Empty>	Required. Permitted type is "Multi_Value_List".

SM Change Management Example

The following is typical for SM change management.

```
<?xml version="1.0" encoding="UTF-8"?>
<itg:mapping xmlns:itg="http://www.hp.com/smci/SMQCIntegration/config">
<itg:module name="change">
  <itg:field name="Urgency" type="Single_Value_List" readonly="false"
required="mandatory" length="50">
    <itg:items>
      <itg:item value="1">1 - Critical</itg:item>
      <itg:item value="2">2 - High</itg:item>
      <itg:item value="3">3 - Average</itg:item>
    </itg:items>
  </itg:field>
```

```
</itg:module>
</itg:mapping>
```

SM Problem Management Example

The following is the included configuration_file_default.xml for SM problem management.

```
<?xml version="1.0" encoding="UTF-8"?>
<itg:mapping xmlns:itg="http://www.hp.com/smci/SMQCIntegration/config">

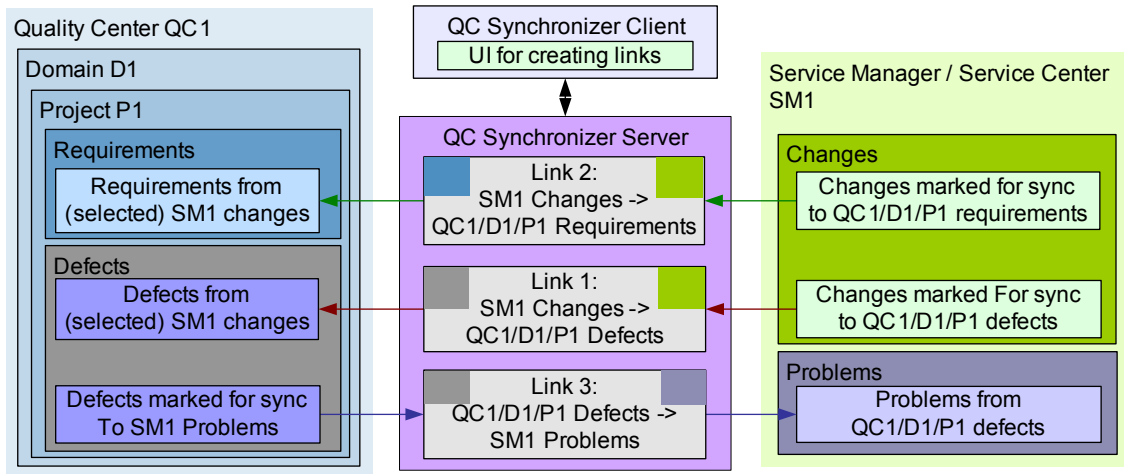
  <itg:module name="problem">
    <itg:field name="Status" type="Single_Value_List"
      required="mandatory">
      <itg:items>
        <itg:item value="Open">Open</itg:item>
        <itg:item value="Accepted">Accepted</itg:item>
        <itg:item value="Work In Progress">Work In Progress</itg:item>
        <itg:item value="Pending Vendor">Pending Vendor</itg:item>
        <itg:item value="Pending User">Pending User</itg:item>
        <itg:item value="Rejected">Rejected</itg:item>
        <itg:item value="Deferred">Deferred</itg:item>
      </itg:items>
    </itg:field>
    <itg:field name="AssignmentGroup" type="Single_Value_List"
      required="mandatory">
      <itg:items>
        <itg:item value="Application">Application</itg:item>
        <itg:item value="Network">Network</itg:item>
      </itg:items>
    </itg:field>
    <itg:field name="Service" type="Single_Value_List" required="mandatory">
      <itg:items>
        <itg:item value="Applications">Applications</itg:item>
        <itg:item value="Service Management">Service Management</itg:item>
      </itg:items>
    </itg:field>
    <itg:field name="Title" type="String" required="mandatory" length="50"/>
      <itg:field name="Description" type="String" required="mandatory"/>
    <itg:field name="Area" type="Single_Value_List" required="mandatory">
      <itg:items>
        <itg:item value="data">data</itg:item>
      </itg:items>
    </itg:field>
    <itg:field name="Subarea" type="Single_Value_List" required="mandatory">
      <itg:items>
        <itg:item value="data or file corrupted">data or file
        corrupted</itg:item>
      </itg:items>
    </itg:field>
    <itg:field name="Impact" type="Single_Value_List" readonly="false"
      required="mandatory">
      <itg:items>
        <itg:item value="1">1 - Enterprise</itg:item>
        <itg:item value="2">2 - Site/Dept</itg:item>
```

```
        <itg:item value="3">3 - Multiple Users</itg:item>
        <itg:item value="4">4 - User</itg:item>
    </itg:items>
</itg:field>
<itg:field name="Urgency" type="Single_Value_List" readonly="false"
        required="mandatory">
    <itg:items>
        <itg:item value="1">1 - Critical</itg:item>
        <itg:item value="2">2 - High</itg:item>
        <itg:item value="3">3 - Average</itg:item>
        <itg:item value="4">4 - Low</itg:item>
    </itg:items>
</itg:field>
</itg:module>
</itg:mapping>
```


4 Configuring Links in Quality Center Synchronizer

This chapter shows how to configure and test links.

The following diagram summarizes link configuration.



You need to create synchronization links in QC Synchronizer between two endpoints. Each endpoint is an application or system containing data that is synchronized by the synchronizer. A link defines which entities are included in the synchronization, and how the synchronization is performed.

This section describes aspects of link creation that are common to all three types of links.

- [Create a Link](#)
- [QC Field <-> SM Field mappings](#)
- [List Value Mappings](#)
- [Constant -> SM field Mappings](#)

Filters are only required for QC Defect -> SM Problem (see [Filters](#) on page 91). The events settings determine what QCS does in response to specified events. Events must be specified for all three link types.

Create a Link

The following table summarizes the properties required in the wizard. Have this data available before starting the wizard.

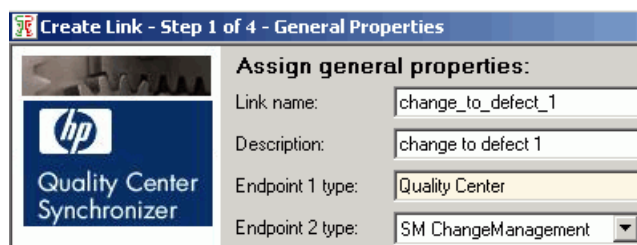
- ▶ A link can not be duplicated. For example, if a link already exists for SMServer1/Changes -> QCServer1/Doman1/Project1/Defects, then a second link between these two entities can not be created.

Table 2 Wizard Link Properties

End Point	Parameter	Requirements
QC	Username	
QC	Password	
QC	Server URL	
QC	Domain	
QC	Project	
SM	User name	
SM	Password	
SM	Service URL	http://<service_manager_host>:<port>/sc62server/PWS/QCIntChangeService.wsdl or http://<service_manager_host>:<port>/sc62server/PWS/QCIntProblemService.wsdl
SM	Adapter Configuration (SM field values) filename	Empty or the adapter data folder file (see Copying SM Adapter Configuration Files on page 19).
SM	QCProject	Required (because of an adapter limitation). Format is qc_host/qc_domain/qc_project

To create a link, perform the following steps:

- 1 Click **Link / Create**. The “Step 1: Assign general properties” dialog appears.
- 2 Enter the required information (the following example is for SM Change -> QC Defect).



- 3 Click **Next**. The “Step 2: Assign QC endpoint connection properties” dialog appears.

- 4 Enter the required information (see table below for details).

Parameter	Value
ServerURL	http://localhost:8080/qcbin
Domain	DEFAULT
Project	Demo

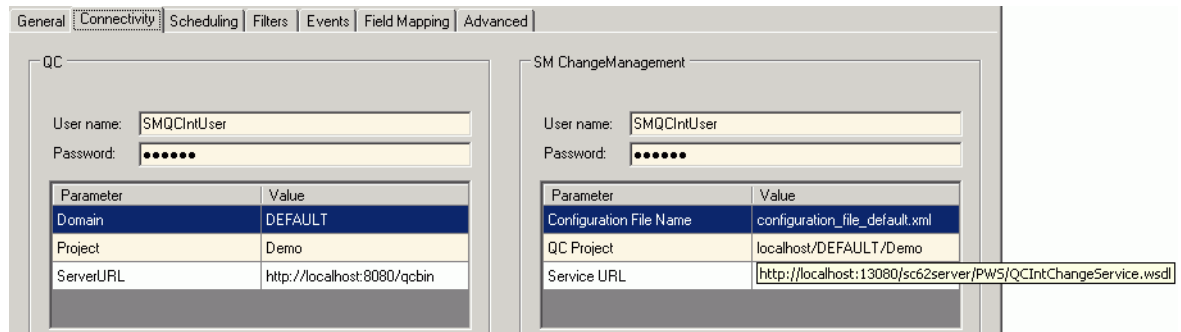
- 5 Click **Next**. One of the following appears:
 - “Step 3: Assign SM ChangeManagement endpoint connection properties”
 - “Step 3: Assign SM ProblemManagement endpoint connection properties”
- 6 Enter the required information (the following example is for SM Change -> QC Defect).

Parameter	Value
Service URL	http://localhost:13080/sc62server/PWS/QCIntChangeService.wsdl
Configuration File Name	configuration_file_default.xml
QC Project	localhost/DEFAULT/Demo

➤ QC Project has the same value as specified on SM customization.

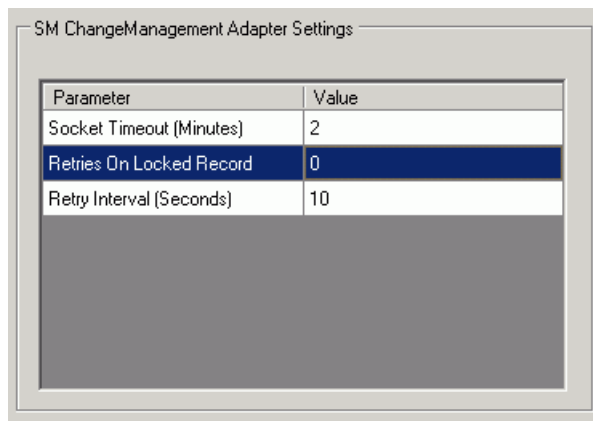
- 7 Click **Next**. If this is a change management link, then “Step 4: Select entity types” dialog appears.
- 8 Select one of the following:
 - **Change as Defect**
 - **Change as Requirement**

- 9 Click **Save**. The link is created.
- 10 Modify required settings in the Connectivity tab.



Advanced Parameters

Advanced parameters are shown on Advanced tab.



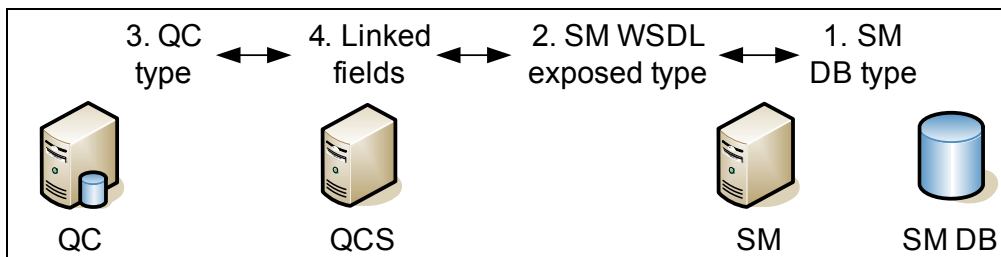
- Retries On Locked Record
When a record in SM endpoint is locked, it will cause the synchronization failure. Integration will retry to synchronize according to the value of this parameter. 0 means disabled. See details in [SM/QC locked record](#) section of [Appendix A, Known Issues and Limitations](#).
- Retry Interval (Seconds)
When retry feature is enabled, this parameter defines the retry interval. The retry interval must be an integer between 1 and 10. See details in [SM/QC locked record](#) section of [Appendix A, Known Issues and Limitations](#).
- Socket Timeout (Minutes)
Socket connection will be established during synchronization. If there are many records matching the filter in SM endpoint, retrieving list operation will cost some time, which might cause timeout of socket connection. This parameter is used to define the socket timeout. Its scope: 0 ~ 120.

QC Field <-> SM Field mappings

This section describes how to map QC fields and SM fields.

- [Create Mapping](#)
- [General Mapping Requirements](#)
- [Matching Types](#)

The following diagram shows the field mapping chain.



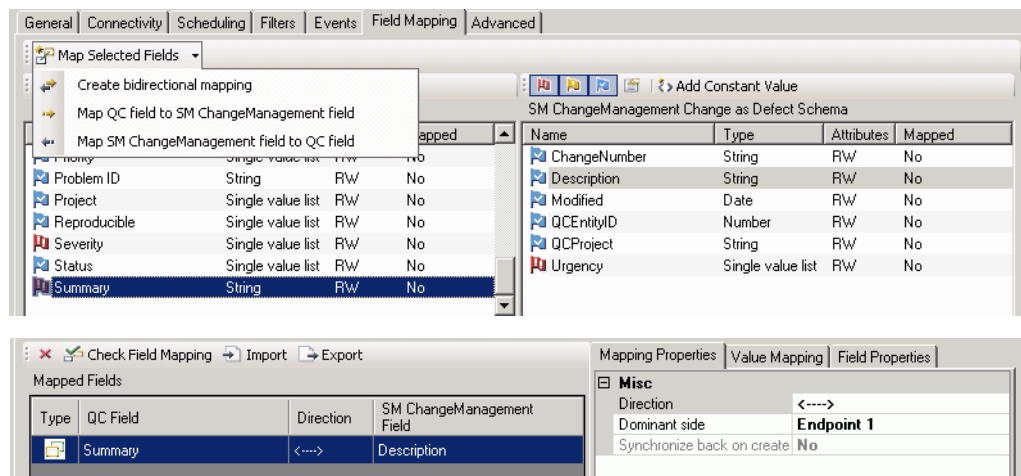
For examples of field mappings, see

- SM Change to QC Defect, [QC Field <-> SM Field](#) on page 50
- SM Change to QC Requirement, [QC Field <-> SM Field](#) on page 61
- QC Defect to SM Problem, [QC Field <-> SM Field](#) on page 93

Create Mapping

To map fields, follow these steps.

- 1 Select a field on each side.
- 2 From “Map Selected Fields” select the direction.



General Mapping Requirements

When creating field links, keep the following limitations in mind:

- If you change the mappings you must do a full synchronization to ensure synchronization of historical data. Otherwise, your historical data can not be synchronized correctly and you may get errors in the next incremental synchronization.
- A field in one endpoint can be mapped to only one field in the other endpoint.
- Mandatory fields must be mapped. If a null value is written to a mandatory field, an error will occur at runtime.
- If you map string fields with different maximum lengths, during synchronization a string value in the source endpoint will be truncated as necessary if it exceeds the maximum length of the corresponding field.

Matching Types

The following table shows the allowed data type combinations. Highlighted entries are demonstrated in examples in this document.

Table 3 Data Type Combinations

QC Data type	QCS QC Type	Dir	QCS SM type	WSDL data type	Field type on form in SM/SC	SM DB data type	
						SM	SC
Number	Number	<->	Number	DecimalType or IntType ^a	Decimal or Text	Number	Decimal
String	String	<->	String	BooleanType	Check Box or Radio Button	Logical	Boolean
Date ^b	Date	<->	Date	DateTimeType (required)	Date	Date/time	
String	String	<->	String	StringType ^e	Text, TextArea, Combo Box or Comfill	Character	Text
Memo	Memo/String	<->	String	StringType ^e	Text, TextArea, Combo Box or Comfill	Character	Text
User List ^d	User List	->	String	StringType ^e	Text, TextArea, Combo Box or Comfill	Character	Text
Lookup List	Single-value list	<->	Single value list/ String	StringType ^e	Text, TextArea, Combo Box or Comfill	Character	Text
String	String	<->	String	StringType	TextArea	Array ^c	
Memo	Memo/String	<->	String	StringType	TextArea	Array ^c	
Lookup List	Single-value List	<->	String	StringType	TextArea	Array ^c	
User List ^d	User List	<->	String	StringType	TextArea	Array ^c	

Table 3 Data Type Combinations (cont'd)

QC Data type	QCS QC Type	Dir	QCS SM type	WSDL data type	Field type on form in SM/SC	SM DB data type	
						SM	SC
Lookup List	Multi-value List	<->	Multi-value List	StringType	Text, Comfill or Combo Box	Array ^c	

Notes:

- a IntType supports a data range from -2,147,483,648 to 2,147,483,647.
- b QC data only supports Yr/Mo/Dt.
- c Only an array of characters is supported.
- d Write to the QC field `User_List` only if SM has exactly the same users (including logins, names, etc.) as QC. An incorrect entry can cause serious problems in QC. You can read from QC `User_List` field and write to SM `String` type field only if the field in SM is NOT a field with SM logins.
- e It is recommended to leave this field blank. Otherwise "Invalid byte 2 of 3-byte UTF-8 sequence" might occur if certain I18N characters are synchronized.

➤ No need to explicitly specify WSDL data type on WSDL configuration for all types except for Date type. For details, see *Best Practices for Publishing and Consuming Web Services with ServiceCenter*.

List Value Mappings

This section describes how to map values for multi-valued lists. QCS does not have access to the values of SM multi-values lists, and therefore the values must be specified in an XML file. Some list fields also require mapping of available values (as shown in the following diagram).

The screenshot shows the 'Field Mapping' tab in the Quality Center Synchronizer. It displays two schemas: 'QC Defect Schema' and 'SM ChangeManagement Change as Defect Schema'. The 'QC Defect Schema' includes fields like Priority, Problem ID, Project, Reproducible, Severity, Status, and Summary. The 'SM ChangeManagement Change as Defect Schema' includes fields like ChangeNumber, Description, Modified, QCEntityID, QCProject, and Urgency. Below the schemas, the 'Mapped Fields' section shows a mapping between 'Severity' and 'Urgency'. The 'Mapping properties' table is circled in red and contains the following data:

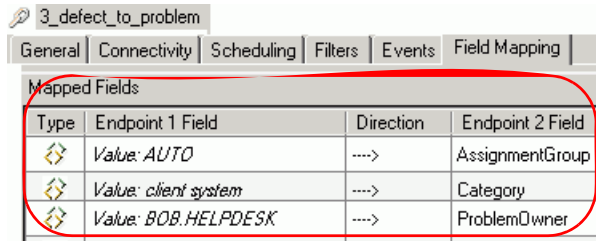
Endpoint 1 value	Direction	Endpoint 2 value
5-Urgent	<---->	1 - Critical
4-Very High	<---->	2 - High
3-High	<---->	3 - Average
2-Medium	<---->	4 - Low

Constant -> SM field Mappings

For examples of constant -> SM field mappings see

- [Constants -> SM Fields](#) on page 93)

The following diagram shows an example of constant -> SM field mapping.



The screenshot shows a software interface with a tabbed menu at the top: 'General', 'Connectivity', 'Scheduling', 'Filters', 'Events', and 'Field Mapping'. The 'Field Mapping' tab is active. Below the tabs is a table titled 'Mapped Fields' with four columns: 'Type', 'Endpoint 1 Field', 'Direction', and 'Endpoint 2 Field'. The table contains three rows of mappings, each with a double-headed arrow icon in the 'Type' column. A red oval highlights the entire table area.

Type	Endpoint 1 Field	Direction	Endpoint 2 Field
↔	Value: <i>AUTO</i>	---->	AssignmentGroup
↔	Value: <i>client system</i>	---->	Category
↔	Value: <i>BOB.HELPDESK</i>	---->	ProblemOwner

5 Integration Account

Creating a SM Integration Account

The integration account is equivalent to the operator in Service Manager for exclusive use with this solution.

Contact

Create a contact for the integration administrator (**Menu Navigation** → **System Administration** → **Base System configuration** → **Contacts** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Support** → **Contacts** on ServiceCenter).

Page	Field	Value
Contact Information	Contact Name	<Administrator's name>
Contact Information	Full Name	<Administrator's full name>

Profile Records

Profile for Change Management

▶ Follow this step if you deploy user story "Change->Requirement" and "Change->Defect".

Profile records grant specific rights and privileges to the integration account to enable Change Management.

On Service Manager 7.0x/7.10

Create a change management profile record (**Menu Navigation** → **System Administration** → **Ongoing Maintenance** → **Profiles**) with the parameters shown in the following table.

No	Tab Page	Field	Value	Comment
1		Profile Name	CMProfile_QCInt	
2		Profile Area	Changes	
3	Security/Rights	Update	Always	

No	Tab Page	Field	Value	Comment
4	Security/Rights	View	Yes	Check Box
5	Security/Rights	Reopen	Yes	Check Box
6	Query	Query Options/All	Yes	Check Box

On ServiceCenter

Create a change management profile record (**Menu Navigation** → **Services** → **Change Management** → **Maintenance** → **Profiles**) with the parameters shown in the following table.

No	Tab Page	Field	Value	Comment
1		Profile Name	CMProfile_QCInt	
2		Profile Area	Changes	
3	Basic/Basic Options	Open	Yes	Check Box
4	Basic/Basic Options	Reopen	Yes	Check Box
5	Basic/Basic Options	Save	Yes	Check Box
6	Query/Query Options	All	Yes	Check Box

Profile for Problem Management

- ▶ Follow this step if you deploy user story "Problem<->Defect"/"Problem->Defect"/"Problem<-Defect".

Profile records grant specific rights and privileges to the integration account to enable Problem Management.

On Service Manager 7.0x/7.10

Create a problem management profile record (**Menu Navigation** → **System Administration** → **Ongoing Maintenance** → **Profiles**) with the parameters shown in the following table.

No	Tab Page	Field	Value	Memo	Remarks
1		Profile Name	PMProfile_QCInt		
2	Problems/Security/Rights	New	Yes	Check Box	This parameter is not required for user story "Problem->Defect"
3	Problems/Security/Rights	Close	Yes	Check Box	
4	Problems/Security/Rights	Update	Always		
5	Problems/Security/Rights	Reopen	Yes	Check Box	

On ServiceCenter

Create a problem management profile record (**Menu Navigation** → **Services** → **Problem Management** → **Administration** → **User Profiles**) with the parameters shown in the following table.

No	Tab Page	Field	Value	Memo	Remarks
1		Profile Name	PMProfile_QCInt		
2	Problem Details	Browse	Yes	Check Box	
3	Problem Details	Open	Yes	Check Box	This parameter is not required for user story "Problem->Defect"
4	Problem Details	Update	Yes	Check Box	
5	Problem Details	Reopen	Yes	Check Box	

Operator

The operator record identifies the logon name, password, and other settings for each SM operator. Create the required operator records (**Menu Navigation** → **System Administration** → **Ongoing Maintenance** → **Operators** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Utilities** → **Administration** → **Security** → **User Administration** → **Search for Operators** on ServiceCenter) with the parameters shown in [Table 4](#).

Table 4 Operator Record Parameters

No	Page	Field	Value	Remarks
1	General	Logon Name	SMQCIntUser	
2	General	Full Name	QC Integration Default Account	
3	General	Contact ID	<Integration administrator's account in SM>	The contact created in the previous section.
4	Security	Unlimited Sessions	Yes	Check Box
5	Security	Password	<Your password>	
6	Startup	Execute Capabilities	SOAP API	
7	Login Profile	Time Zone	Greenwich/ Universal (or create a time zone with no time difference or DST switch in Database Manager)	

Table 4 Operator Record Parameters (cont'd)

No	Page	Field	Value	Remarks
8	Login Profile	Date Format	yy/mm/dd	The date format can not be changed (changing it will cause loss of all data during synchronization).
9	Startup	Execute Capabilities	ChMAdmin	Set these two paramters if you deploy user stories "Change->Defect" and "Change->Requirement".
10	General/ Application Profiles	Change Profiles	CMProfile_QCInt	
11	Startup	Execute Capabilities	ProbAdmin	Set these two paramters if you deploy user story "Problem<->Defect"/ "Problem->Defect"/ "Defect->Problem".
12	General/ Application Profiles	Problem Profile	PMProfile_QCInt	

Creating a QC Integration Account

To create an integration account, follow these steps.

- 1 Create User
- 2 Create Group
- 3 Assign Permission

Create User

To create a user, follow these steps.

- 1 Log in to the "Quality Center - Site Administration" using Quality Center site administrator account.
- 2 In the **Site Users** tab, create and configure integration account **SMQCIntUser** (including the User Name and password).
- 3 In the **Site Projects** tab, choose the project from the list.
- 4 Click the **Project Users** tab in the right panel and click **Add From The Users List**.
- 5 Add the configured user **SMQCIntUser** to the project.
- 6 Log off.

Create Group

- 1 Log on to the QC project using project administrator account.
- 2 Click **TOOLS** → **Customize...**

- 3 Select **Groups**.
- 4 Click **New**.
- 5 Enter name **SMIntegration**.
- 6 For Create As: select **Viewer**.
- 7 Click **OK**.
- 8 Select **Yes** to create the user group.

Assign Permission

- 1 Click **Change** and assign permissions for the user group as shown in the following table.

— Change-> Requirement

Requirement	Add Requirement/Modify Requirement	<input checked="" type="checkbox"/> Add Requirement <input checked="" type="checkbox"/> Modify Requirement <input type="checkbox"/> Delete Requirement <input type="checkbox"/> Add Tests To Coverage <input type="checkbox"/> Remove Tests From Coverage <input type="checkbox"/> Add Requirement Traceability <input checked="" type="checkbox"/> Modify Requirement Traceability <input type="checkbox"/> Remove Requirement Traceability <input checked="" type="checkbox"/> Risk-Based Quality Management
-------------	------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

— Change->Defect

Defects	Add Defects/Modify Defects	<input checked="" type="checkbox"/> Add Defect <input checked="" type="checkbox"/> Modify Defect <input type="checkbox"/> Delete Defect <input type="checkbox"/> Add Defect Link <input checked="" type="checkbox"/> Modify Defect Link <input type="checkbox"/> Remove Defect Link
---------	----------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

— Problem->Defect

Defects	Add Defects/Modify Defects	<input checked="" type="checkbox"/> Add Defect <input checked="" type="checkbox"/> Modify Defect <input type="checkbox"/> Delete Defect <input type="checkbox"/> Add Defect Link <input checked="" type="checkbox"/> Modify Defect Link <input type="checkbox"/> Remove Defect Link
---------	----------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

— Problem<-Defect

Defects	Add Defects/Modify Defects	<input type="checkbox"/> Add Defect <input checked="" type="checkbox"/> Modify Defect <input type="checkbox"/> Delete Defect <input type="checkbox"/> Add Defect Link <input checked="" type="checkbox"/> Modify Defect Link <input type="checkbox"/> Remove Defect Link
Administration	Add Public Favorite Views/Modify Public Favorite Views/Delete Public Favorite Views/Add Private Favorite Views/Modify Private Favorite Views/Delete Private Favorite Views	<input checked="" type="checkbox"/> Add Public Favorite Views <input checked="" type="checkbox"/> Modify Public Favorite Views <input checked="" type="checkbox"/> Delete Public Favorite Views <input checked="" type="checkbox"/> Add Private Favorite Views <input checked="" type="checkbox"/> Modify Private Favorite Views <input checked="" type="checkbox"/> Delete Private Favorite Views

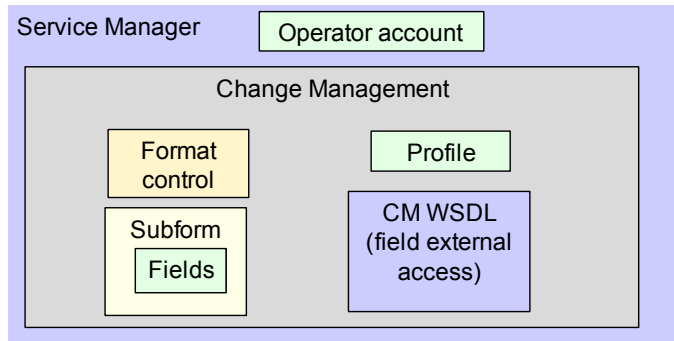
— Problem<->Defect

Defects	Add Defects/Modify Defects	<input checked="" type="checkbox"/> Add Defect <input checked="" type="checkbox"/> Modify Defect <input type="checkbox"/> Delete Defect <input type="checkbox"/> Add Defect Link <input checked="" type="checkbox"/> Modify Defect Link <input type="checkbox"/> Remove Defect Link
Administration	Add Public Favorite Views/Modify Public Favorite Views/Delete Public Favorite Views/Add Private Favorite Views/Modify Private Favorite Views/Delete Private Favorite Views	<input checked="" type="checkbox"/> Add Public Favorite Views <input checked="" type="checkbox"/> Modify Public Favorite Views <input checked="" type="checkbox"/> Delete Public Favorite Views <input checked="" type="checkbox"/> Add Private Favorite Views <input checked="" type="checkbox"/> Modify Private Favorite Views <input checked="" type="checkbox"/> Delete Private Favorite Views

- 2 Add the integration user SMQCIntUser to group SMIntegration.
- 3 Save and close. The integration account is created.

6 SM Change -> QC Defect

Customizing Service Manager/ServiceCenter for Change Management



To customize SM for change management, follow these steps

- 1 Add Fields
- 2 Specify Field External Access
- 3 Create Subform
- 4 Add Subform to Form
- 5 Add Format Control Calculations/Validations

Add Fields

Add the following fields to table cm3r (**System Definition** → **Tables** → **cm3r**). The values shown are required (do not change).

Field	Type	
	Service Manager 7.0x/7.10	ServiceCenter
qcintegration.type	Character	Text
qcintegration.id	Number	Decimal
qcintegration.project	Character	Text

► The data type requirements for SM fields are described in [Matching Types](#) on page 30.

Specify Field External Access

On Service Manager 7.0x/7.10

- 1 Create a customized External Access Definition QCIntChangeService (**Menu Navigation** → **Tailoring** → **WSDL configuration** on Service Manager 7.0x; **Menu Navigation** → **Tailoring** → **Web Services** → **WSDL Configuration** on Service Manager 7.10) with

- Service Name: QCIntChangeService
- Name: cm3r
- Object Name: QCIntChange
- Allowed Actions: save / Action Names: Update

► The above values are required (DO NOT change).

This is shown in the following figure.

object.name
Change
QCIntChange

External Access Definition

Service Name:

Name: Object Name:

Allowed Actions | Expressions | Fields

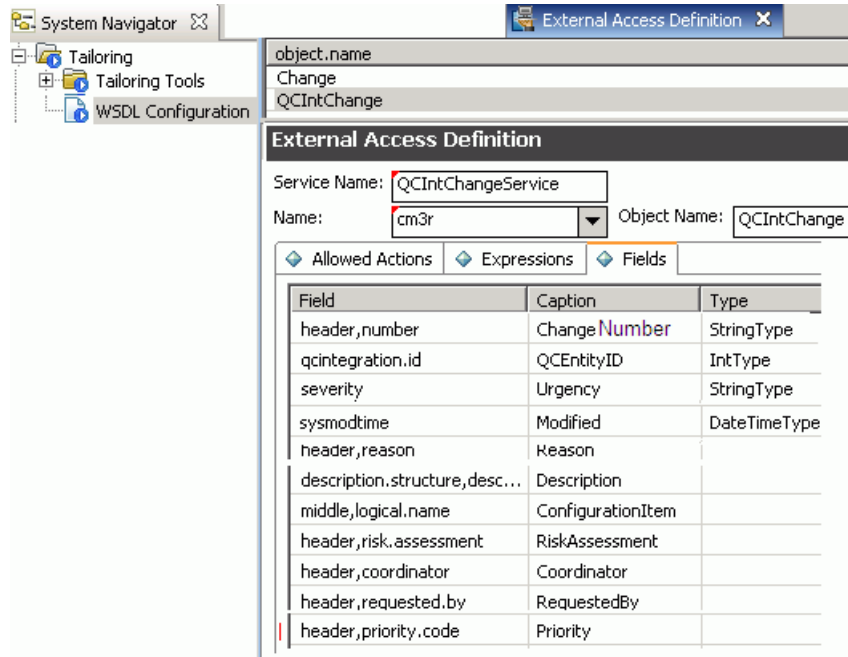
Allowed Actions	Action Names	Action Type
save	Update	

- 2 Enable required fields in the web service.

Field	Caption	Type
header,number	ChangeNumber	StringType
qcintegration.id	QCEntityID	IntType
sysmodtime	Modified	DateTimeType

► The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on). The above values are required (do not change).

This is shown in the following diagram.



On ServiceCenter

All fields of ServiceCenter change entity or problem entity can be exposed in Web service by modifying the WSDL configuration. In ServiceCenter, you can modify the WSDL configuration by changing “Web Services API properties” setting of the field in the table definition.

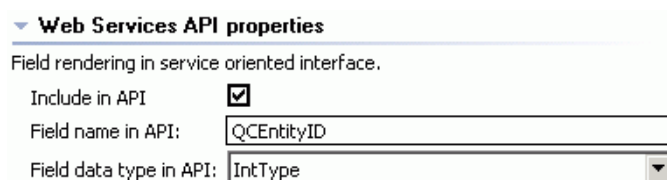
➤ Restart the ServiceCenter server whenever you made changes to the WSDL configuration.

- 1 Modify settings of the following fields in **System Definition** → **Tables** → **cm3r** → **Fields** and **keys definitions for cm3r table**:

No.	Field	Include in API	Field name in API	Field data type in API
1	header,number	Y	ChangeNumber	StringType
2	qcintegration.id	Y	QCEntityID	IntType
3	sysmodtime	Y	Modified	DateTimeType

➤ The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on).

This is shown in the following figure.



- Go to **Menu navigation** → **Toolkit** → **WSDL Configuration** and search for table **cm3r**, then update the External Access Definition as follows based on the table **cm3r**.

No.	Field	Value
1	Service Name	QCIntChangeService
2	Object Name	QCIntChange
3	Allowed Actions	save/Update (Action Names)

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Object Name:

Allowed Actions
 Expressions
 Data Policy

Allowed Actions	Action Names
save	Update

Create Subform

- Create Global list.

Create a global list (**Menu Navigation** → **Tailoring** → **Tailoring Tools** → **Global Lists** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Utilities** → **Tools** → **Global Lists** on ServiceCenter) with following parameters:

No.	Parameter	Value	Remarks
1	List Name	SMQC Integration CM Project List	
2	Regen Every	1 00:00:00	
3	Build List on Startup?	Yes	Check box
4	List Variable	\$G.qcintegration.change.project	
5	User Defined List?	Yes	Check box
6	Value List	{"server1/domain1/project1", "server2/domain2/project2"}	Change to the values for your system Note: No spaces between slashes.

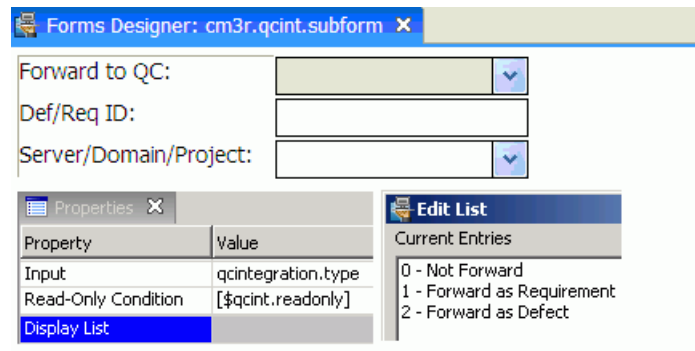
Save this global list and click **Rebuild Global List** from menu.

- Create subform.

Create subform **cm3r.qcint.subform** without the Form Wizard (**Menu Navigation** → **Tailoring** → **Forms Designer** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Toolkit** → **Forms Designer** on ServiceCenter) with the following components on the canvas.

Component	Properties
Label	Caption: "Forward to QC:"
Combo Box	<ul style="list-style-type: none"> Input: "qcintegration.type" Value List: "0;1;2" Display List: "0 - Not Forward;1 - Forward as Requirement;2 - Forward as Defect" Select Only: "Yes" Read-Only Condition: "[\$qcint.type.readonly]"
Label	Caption: "Def/Req ID:"
Text	<ul style="list-style-type: none"> Input: "qcintegration.id" Read-Only: "Yes"
Label	Caption: "Server/Domain/Project:"
Combo Box	<ul style="list-style-type: none"> Input: "qcintegration.project" Value List: "\$G.qcintegration.change.project" Read-Only Condition: "[\$qcint.project.readonly]" Mandatory Condition: "[qcintegration.type]>0"

This is shown in the following diagram.

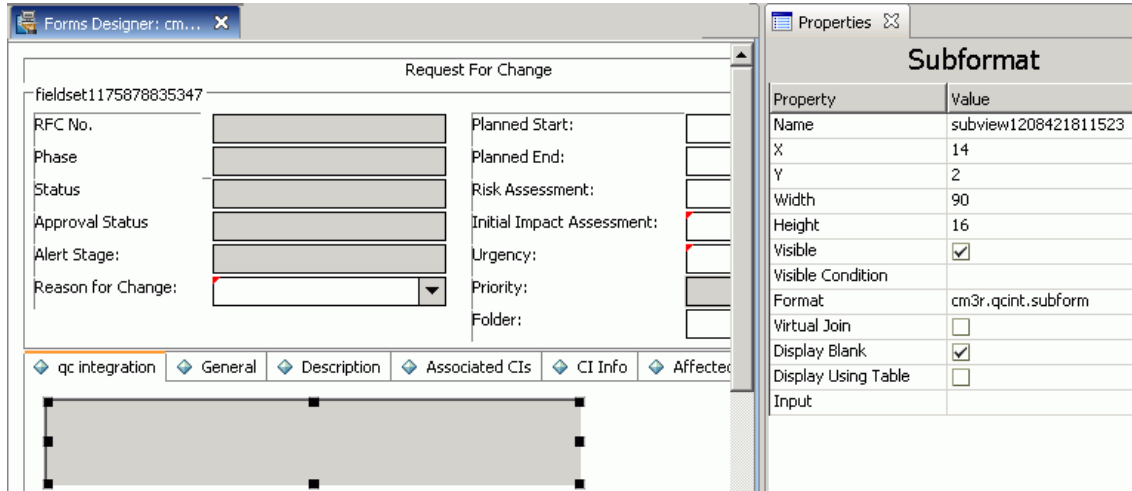


Add Subform to Form

To add a subform to a form, follow these steps.

- 1 Open the form of a phase of a category via the Forms Designer (cm3r.rfc.build.g is used as an example in ServiceCenter 6.2/Service Manager 7.0x).
- 2 Add a notebook tab with caption "QC Integration".
- 3 Add a subform to the new tab with format "cm3r.qcint.subform".

The following diagram shows the form.

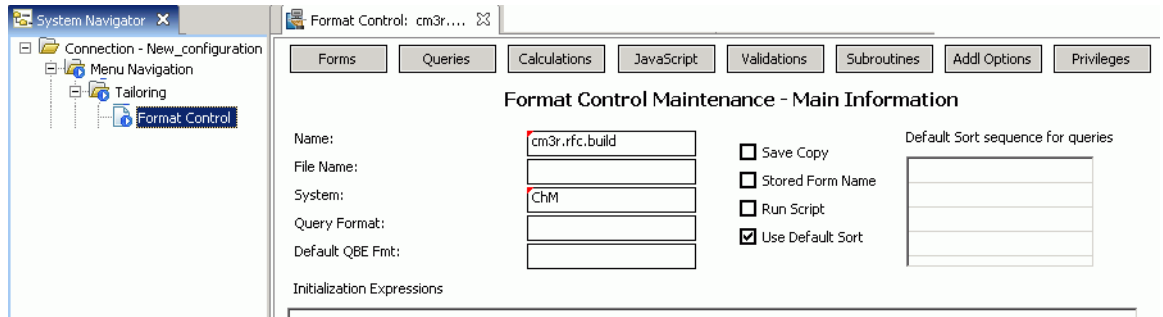


4 Save the changes.

▶ If error message "Format 'cm3r.qcint.subform' not found (display, show.rio)" appears, log out and then log back into your client to enable the subform.

Add Format Control Calculations/Validations

1 Open the form format control of the previous change form by clicking **Format Control** from menu (cm3r.rfc.build is used as an example in ServiceCenter 6.2/Service Manager 7.0x).



2 Click **Calculations**.

3 Add two records.

Parameter	Display	Initial	Calculation
1	true	true	<code>\$qcint.type.readonly=2;if (qcintegration.type in \$file~=0) then (\$qcint.type.readonly=1)</code>
2	true	true	<code>\$qcint.project.readonly=2;if (qcintegration.type in \$file~=0 and not null(qcintegration.project in \$file)) then (\$qcint.project.readonly=1)</code>

- ▶ When you copy the calculation into the records, make sure that the calculation is in one line, also note that there is a space between lines in this table. For example, the calculation for Parameter 1 is:
`$qcint.type.readonly=2;if (qcintegration.type in $file~=0) then ($qcint.type.readonly=1)`

The change calculations are shown in the following diagram.

- 4 Click **Validations**.
- 5 Add one record with the following parameters.

Table 5 Change Validation Record

No	Parameter	Value
1	Validation	not null(qcintegration.project in \$file)
2	Message	The Server/Domain/Project is required.
3	Add	qcintegration.type in \$file~=0
4	Update	qcintegration.type in \$file~=0
5	Set Focus to	qcintegration.project

The change validation record is shown in the following diagram.

- 6 Save the changes.

Customizing Quality Center Defects Module

To customize Quality Center Defects module, follow these steps:

- 1 Add Fields
- 2 Add Tabs
- 3 Add Fields to Tabs

4 Verify

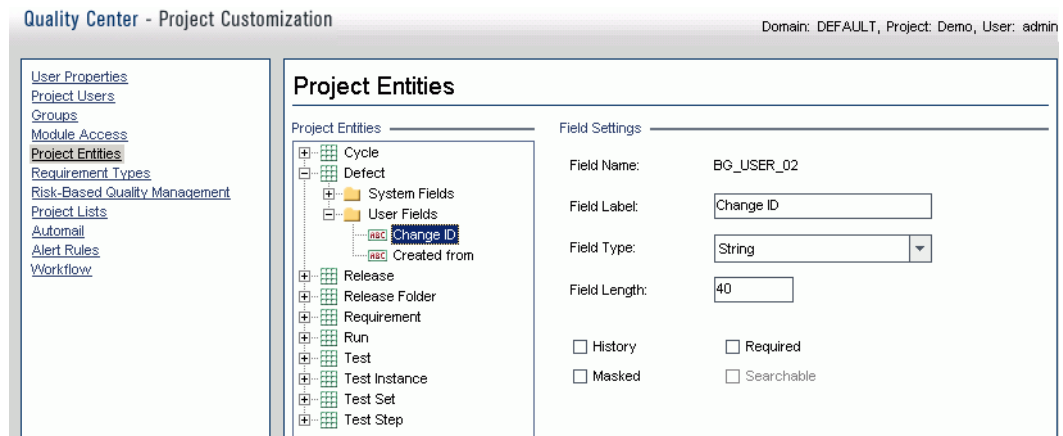
Add Fields

To add the required fields for defect customization, follow these steps.

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Add the following fields for the defect entity in project entities (XX XY are sequential numbers auto-generated by QC).

Field Name	Field Label	Field Type
BG_USER_XX	Change ID	String
BG_USER_XY	Created from	String

The following diagram shows an example project entity.



- The data type requirements for QC fields is described in [Matching Types](#) on page 30.

Add Tabs

To add tabs to the defect form and show fields on these tabs, follow these steps.

- 1 In "QC - Project Customization" click **Workflow**.
- 2 Click **Workflow** → **Script Editor**.

- 3 Choose **Defects module script**.

Quality Center - Project Customization

[User Properties](#)

[Project Users](#)

[Groups](#)

[Module Access](#)

[Project Entities](#)

[Requirement Types](#)

[Risk-Based Quality Man](#)

[Project Lists](#)

[Automail](#)

[Alert Rules](#)

[Workflow](#)

Workflow

[Script Generator - Add Defect Field Customization](#)
Enables you to customize the fields displayed for each user group in the Add Defects dialog box. You can also specify field order and whether a field is required.

[Script Generator - Defect Details Field Customization](#)
Enables you to customize the fields displayed for each user group in the Defect Details dialog box. You can also specify field order and whether a field is required.

[Script Editor](#)
Enables you to write VBScript code for all Quality Center modules. You can also use the Script Editor to modify the scripts generated by the above tools.

Script Editor

Script Editor Toolbar Button Editor

Workflow Scripts

- + Common script
- + Requirements module script
- + Test Plan module script
- + Test Lab module script
- + Manual Runner script
- + Defects module script

- GetNewBugPageName
- GetDetailsPageName
- Bug_New
- Bug_MoveTo
- Bug_FieldCanChange
- Bug_FieldChange
- Bug_CanPost
- Bug_CanDelete
- Bug_AfterPost
- SetFieldApp

- 4 Add the following code to the **GetNewBugPageName** event procedure (which is triggered before QC opens the Add Defect dialog box).

```
select case PageNum
  case "2"
    GetNewBugPageName = "SM Integration (New)"
  end select
```

➤ 2 specifies tab 2 (the second tab). For a new bug, the tab name is **SM Integration (New)**.

- 5 Add the following code to the **GetDetailsPageName** event procedure (which is triggered before QC displays Defect Details dialog box).

```
select case PageNum
  case "2"
    GetDetailsPageName = "SM Integration (Details)"
  end select
```

➤ 2 specifies tab 2 (the second tab). For an existing defect, the tab name is **SM Integration (Details)**.

Add Fields to Tabs

- 1 If WizardFieldCust_Details and WizardFieldCust_Add are not found in the list, then do the following to generate these two methods.
 - a **Script Generator - Add Defect Field Customization**
 - b **Script Generator - Defect Details Field Customization**



- 2 Add the following code to the WizardFieldCust_Details event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
```

The parameters are

- Field name (BG_USER_XX, where XX = two digits)
- Visible (True)
- Required (False)
- Page number (start from 0)
- View order (start from 0)

- 3 Add the following code to the WizardFieldCust_Add event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
```

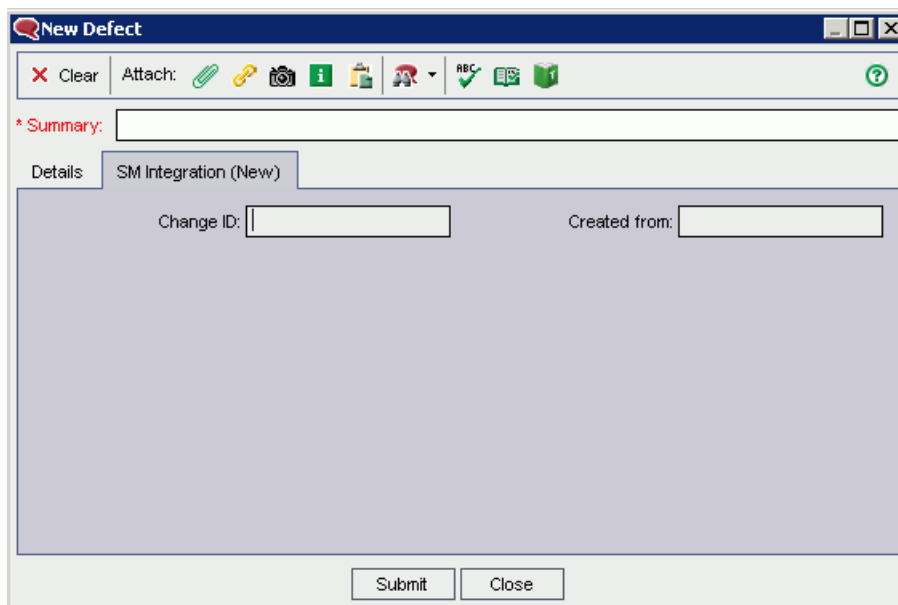
- 4 Set the **ReadOnly** fields by adding the following lines to Bug_New and Bug_Moveto subroutines:

```
Bug_Fields.Field("BG_USER_XX").IsReadOnly=True
Bug_Fields.Field("BG_USER_XY").IsReadOnly=True
```

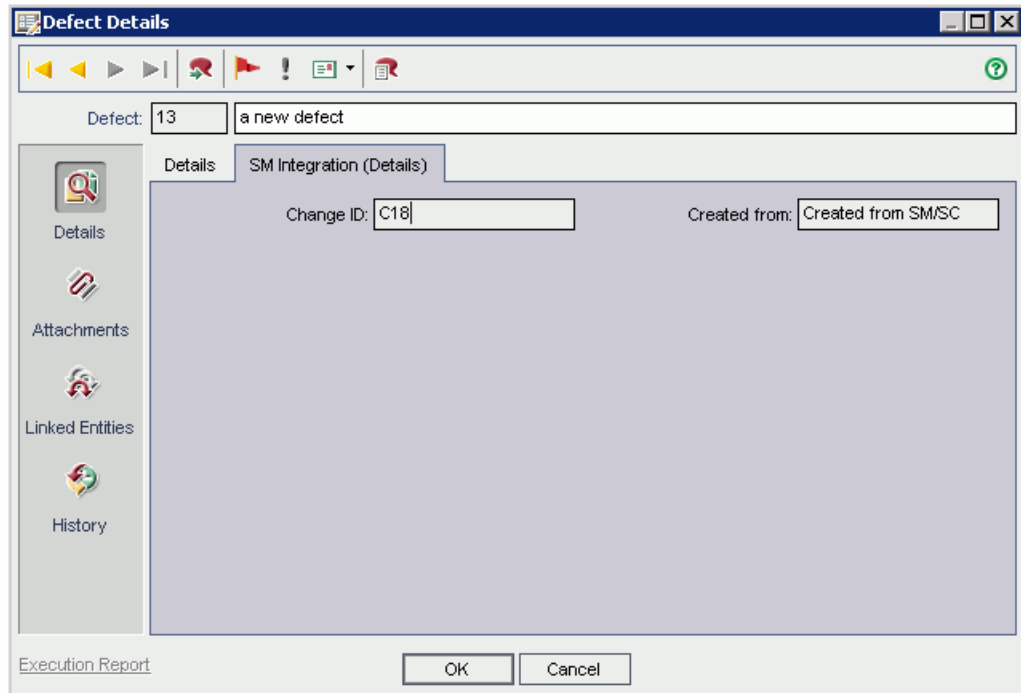
- 5 Exit customization (save changes).

Verify

- 1 Create a new defect. The dialog box should have a new tab titled **SM Integration (New)** with two fields.



- 2 Open an existing defect. The second tab is titled "SM Integration (Detail)", "Change ID" and "Created from" fields are always readonly.



Configuring Links in QC Synchronizer

This section describes how to create and test a link.

- Specify Endpoints / Type of Link
- Field Mappings
- Events
- Test

Specify Endpoints / Type of Link

Specify the connection properties as described in [Create a Link](#) on page 26 with the following settings specific for this type of link:

- 1 Step 1 endpoint 2 type = **SM ChangeManagement**.
- 2 Step 3 service URL =
**http://<service_manager_host>:<port>/sc62server/PWS/
QCIntChangeService.wsdl**
- 3 Step 4 select entity types = **Change as Defect**.

Field Mappings

Basic field mappings are summarized below:

QC	Direction	SM	Constant value	Remarks
Change ID	<-	ChangeNumber		
Defect ID	->	QCEntityID		Synchronize back on create: Yes
Created from			Created from SM/SC	

Example field mappings are shown in following figure.

Type	QC Field	Direction	SM ChangeManagement Field
	Severity	<---->	Urgency
	Change ID	<----	ChangeNumber
	Defect ID	---->	QCEntityID
	Summary	<---->	Description
	Created from	<----	Value: Created from SM/SC

QC Field <-> SM Field

The following table summarizes QC <-> SM field mappings. The highlighted rows are the required mappings.

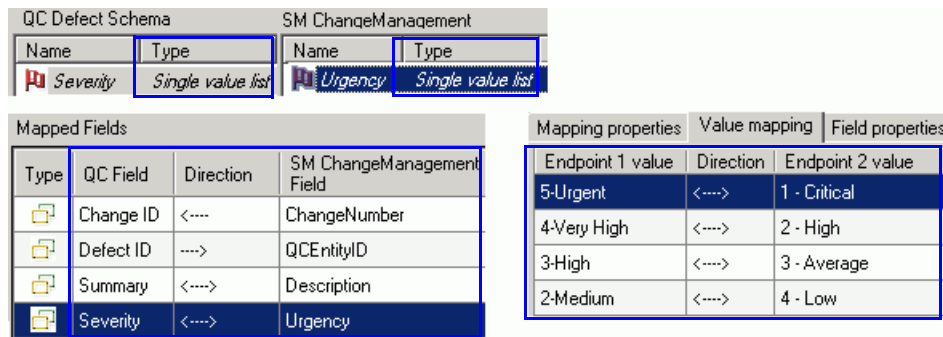
Table 6 SM Change -> QC Defect Mappings

QC Len	QC DB Name	QC Type	QC/QCS Label	QCS QC Type	Dir	QCS SM type	QCS Name / SM WSDL Caption	SM WSDL Type	SM DB Name / SM WSDL field	SM DB type		SM Len
										SM7	SC6	
40	BG_USER_02 ^c	String	Change ID	String	<-	String	ChangeNumber	StringType	header,number	Char	Text	100
10	BG_BUG_ID	Number	Defect ID ^a	Number	->	Number	QCEntityID ^b	IntType	qcintegration.id	Num	Decimal	xx
255	BG_SUMMARY	String	Summary	String	<->	String	Description	StringType	description.structure,description	Char	Text	xx
70	BG_SEVERITY	Lookup List	Severity	Single value list	<->	Single value list	Urgency	Character	severity	Char	Text	40

Notes:

- a QC defect ID is assigned in QC only after saving a new defect.
- b Check for QCEntityID mapping property Synchronize back on create.
- c The numeric suffix on generated names may differ on your system.
- d The length may differ on your system.

If you specify value mapping, for example, Severity <-> Urgency, you can specify as follows:



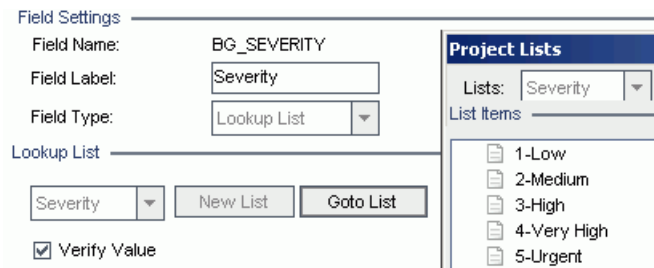
The following table summarizes the mappings you just created between the single value lists (which have their own direction).

Table 7 SM Change -> QC Defect List Value Mappings

QC Len	QC DB Name	QC Type / Lookup list values	QC/QCS Label	QCS QC Type	Dir			QCS SM type	QCS Name/ SM WSDL Caption	SM WSDL Type	SM DB Name / SM WSDL Field	SM DB type	SM Len
					QC value (from lookup list) ^a	value map dir	SM field value (from SM Adapter xml config file for SM Change Management) ^b						
70	BG_SEVERITY	Lookup List	Severity	Single value list	<->			Single value list	Urgency	String Type	severity	Char (SM7) or Text (SC6)	40
					5-Urgent	<->	1-Critical						
					4-VeryHigh	<->	2-High						
					3-High	<->	3-Average						
					2-Medium	<->	4-Low						

Notes:

a Lookup list is created in QC.



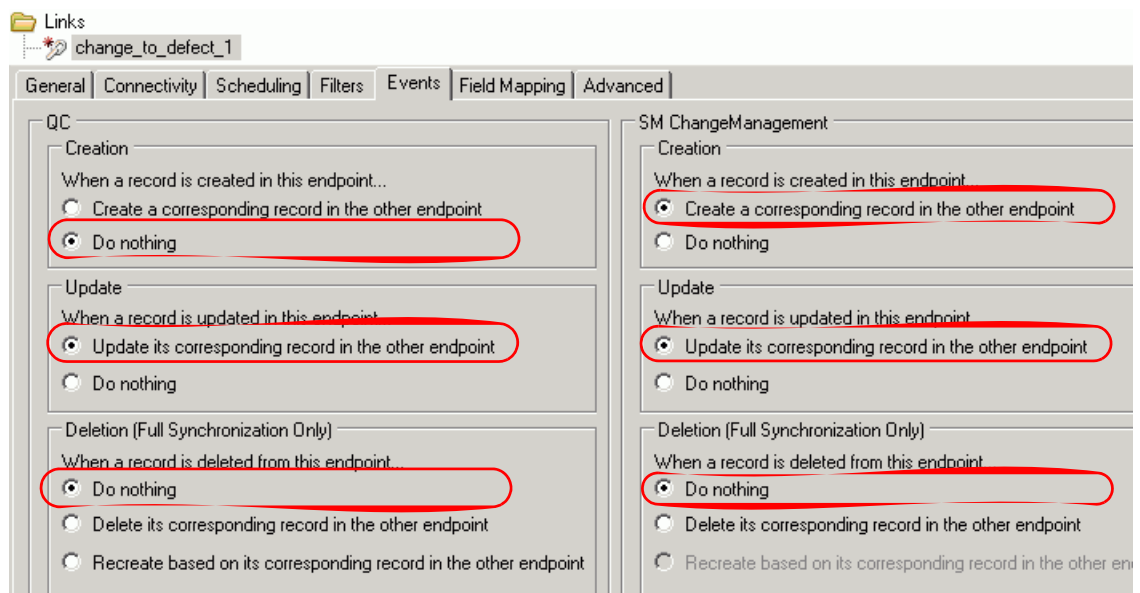
b The XML file is in <QCS_Install_Dir>\adapters\dat\SM ChangeManagement\configuration_file_default.xml (see [SM Change Management Example](#) on page 21).

Events

The following table lists events.

Events Tab Settings	QC Action (Event)	SM Action (Event)
Creation	Do nothing.	Create a corresponding record in the other endpoint.
Update	Update its corresponding record in the other endpoint.	Update its corresponding record in the other endpoint.
Deletion	Do nothing.	Do nothing.

The following diagram shows the settings.

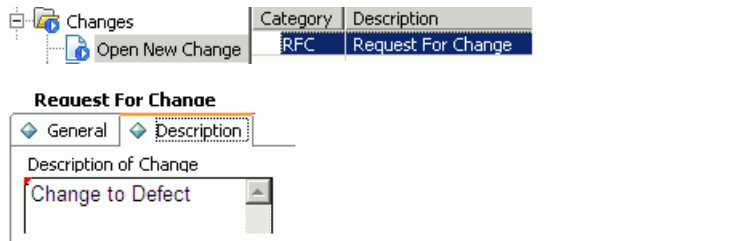


Test

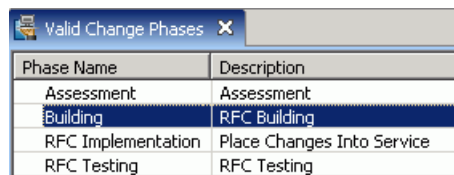
To test the link, follow these steps:

▶ The following is just an example. The exact steps required on your system may differ significantly. The phase in which the tab for QC Integration appears may be different on your system.

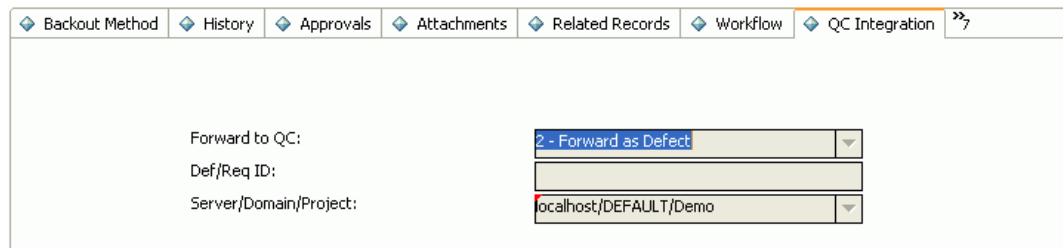
- 1 Save the configuration (an integrity check is automatically run).
- 2 Click **Enable Link**.
- 3 Create a Service Manager change (the category of the change depends on each Service Manager customizations; RFC is used in ServiceCenter 6.2/Service Manager 7.0x as an example).



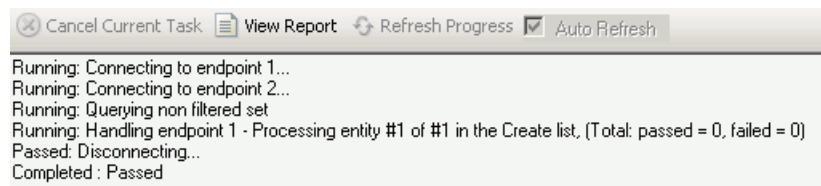
- 4 Change the phase to **Building**. The “QC Integration” tab appears.



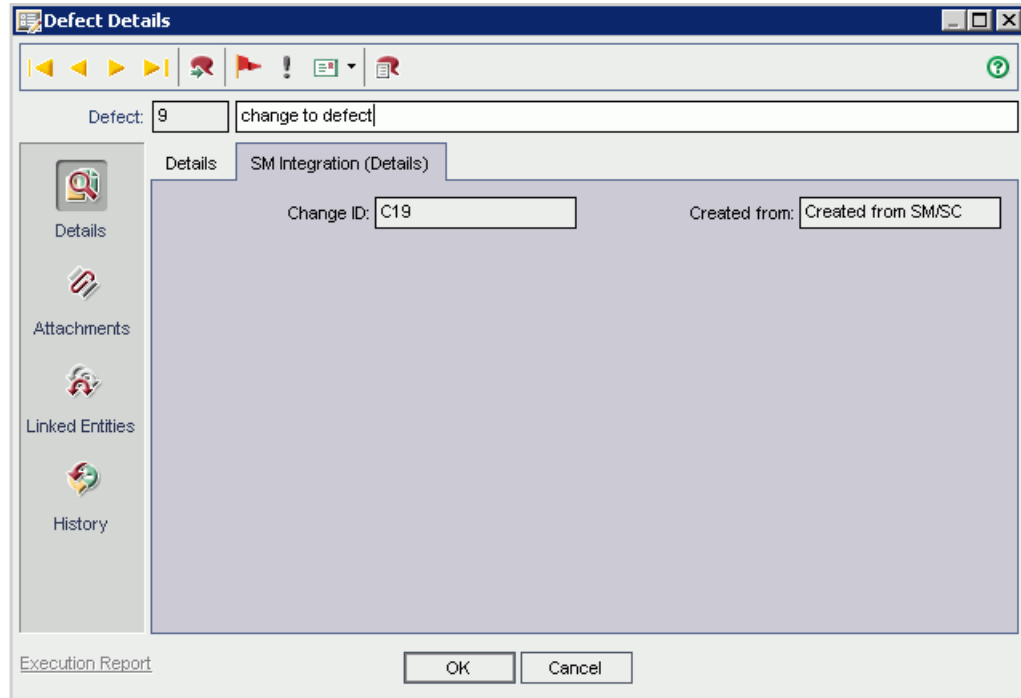
- 5 Select a value for **Server/Domain/Project** field and select **Forward as Defect** for Forward to QC field.



- 6 Synchronize.



7 View the defect in QC.



7 SM Change -> QC Requirement

Customizing Service Manager/ServiceCenter for Change Management

See Customizing Service Manager/ServiceCenter for Change Management section of Chapter 6, SM Change -> QC Defect.

Customizing Quality Center Requirements Module

To customize QC for requirements management, follow these steps


- 1 Add Fields
- 2 Add Tabs
- 3 Add Fields to Tabs
- 4 Create Folder "SM Incoming Changes"

Add Fields

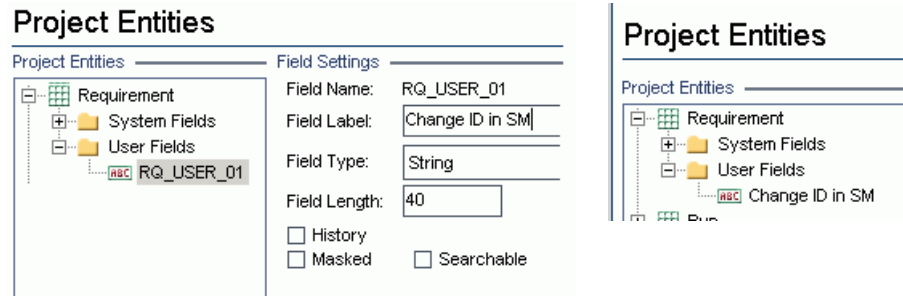
To add required fields for requirement customization, follow these steps.

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Add the following fields for the requirement entity in project entities (XX XY are sequential numbers auto-generated by QC).

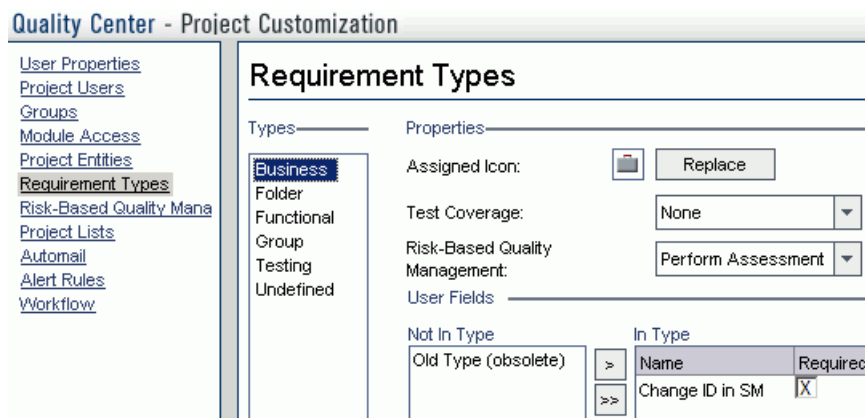
Field Name	Field Label	Field Type
RQ_USER_XX	Change ID	String
RQ_USER_XY	Created from	String

 The data type requirements for QC fields is described in [Matching Types](#) on page 30.

This is shown in the following diagram.



- In Requirement Types add fields "Change ID"/"Created from" to the Business type requirement. Business type is the default requirement type for incoming requirements (other types can be used).



Add Tabs

To add tabs to the Requirement form and display the fields on these tabs, click **Workflow** → **Script Editor**. Add the following code to the requirement module.

- For a new Requirement, the tab label is "SM Integration (New)". For an existing Requirement, the tab label is "SM Integration (Details)". 2 specifies tab 2 (the second tab). If N tabs exist, then the number of a new tab should be N+1. This function is called when an existing requirement is shown in the dialog box.

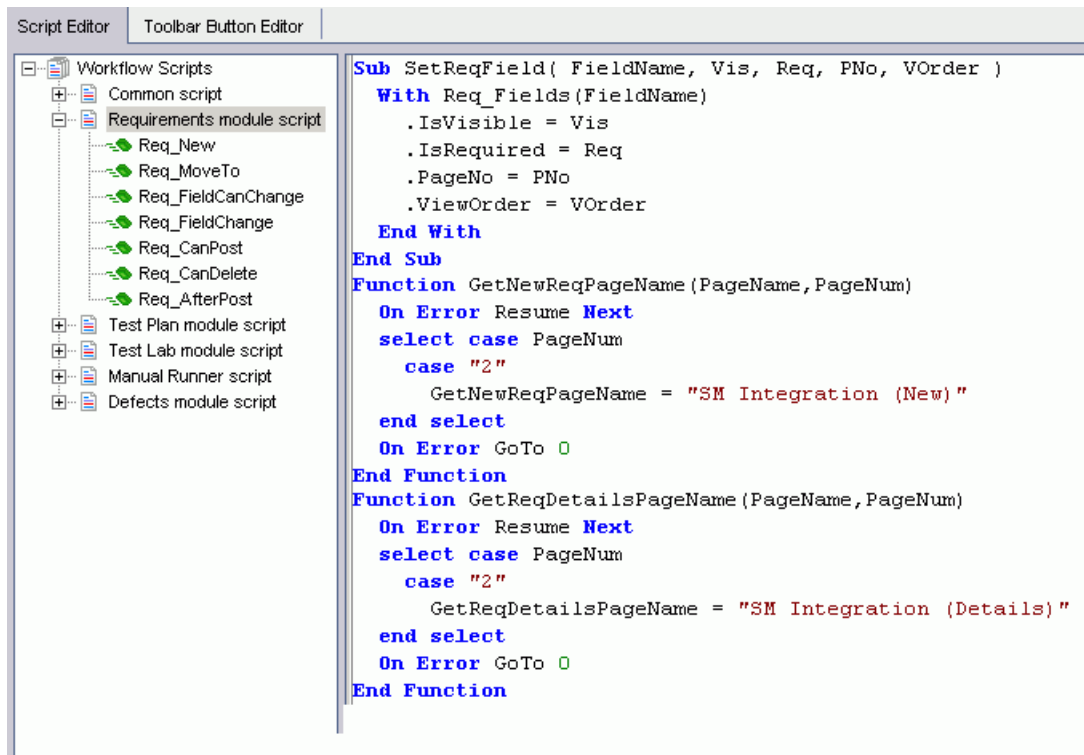
```
Sub SetReqField( FieldName, Vis, Req, PNo, VOrder )
    With Req_Fields(FieldName)
        .IsVisible = Vis
        .IsRequired = Req
        .PageNo = PNo
        .ViewOrder = VOrder
    End With
End Sub
Function GetNewReqPageName (PageName, PageNum)
    On Error Resume Next
    select case PageNum
        case "2"
            GetNewReqPageName = "SM Integration (New)"
```

```

    end select
    On Error GoTo 0
End Function
Function GetReqDetailsPageName (PageName, PageNum)
    On Error Resume Next
    select case PageNum
        case "2"
            GetReqDetailsPageName = "SM Integration (Details)"
    end select
    On Error GoTo 0
End Function

```

The resulting script is shown in the following diagram.



Add Fields to Tabs

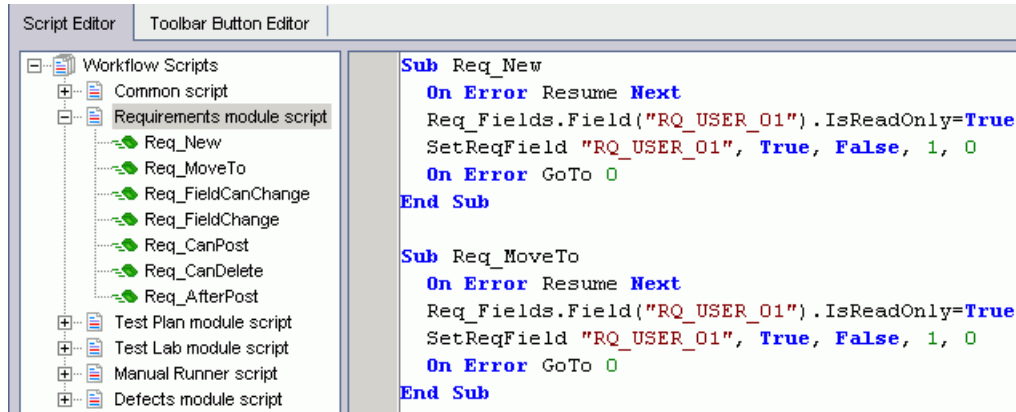
To set the fields as read only and place the fields on the tabs, in the Script Editor for the Requirements module script, add the following code to Req_New and Req_Moveto (Req_New is called when a new Requirement is created; Req_Moveto is called when an existing Requirement is opened).

```

Req_Fields.Field("RQ_USER_XX").IsReadOnly=True
Req_Fields.Field("RQ_USER_XY").IsReadOnly=True
SetReqField "RQ_USER_XX", True, False, 1, 0
SetReqField "RQ_USER_XY", True, False, 1, 1

```

This is shown in the following diagram.



Create Folder “SM Incoming Changes”

To create the folder for the requirements created from SM changes, follow these steps.

- 1 From the menu, select **Requirements / New Folder**.
- 2 Set the folder name to **SM Incoming Changes**.

The screenshot shows the Requirements application window with a menu bar (Requirements, Edit, View, Favorites, Analysis) and a toolbar. Below the toolbar is a table listing folders:

Name	Direct Cover Status	Req ID
Requirements	----	0
SM Incoming Changes	----	1

Configuring Links in QC Synchronizer

This section describes how to create and test a link.

- Specify Endpoints / Type of Link
- Field mappings
- Events
- Test

Specify Endpoints / Type of Link

Specify the connection properties as described in [Create a Link](#) on page 26 with the following settings specific for this type of link:

- 1 Step 1 endpoint 2 type = **SM ChangeManagement**.
- 2 Step 3 service URL =
http://<service_manager_host>:<port>/sc62server/PWS/QCIntChangeService.wsdl
- 3 Step 4 select entity types = **Change as Requirement**.
- 4 Specify the incoming requirement folder as shown in the following figure.

General | Connectivity | Scheduling | Filters | Subtype Mapping | Advanced

QC

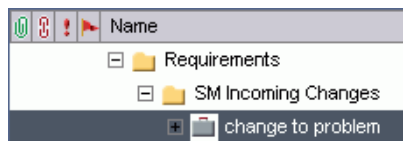
User name: SMQCIntUser
Password: ●●●●●●

Parameter	Value
Domain	DEFAULT
Project	Demo
ServerURL	http://localhost:8080/qcbin

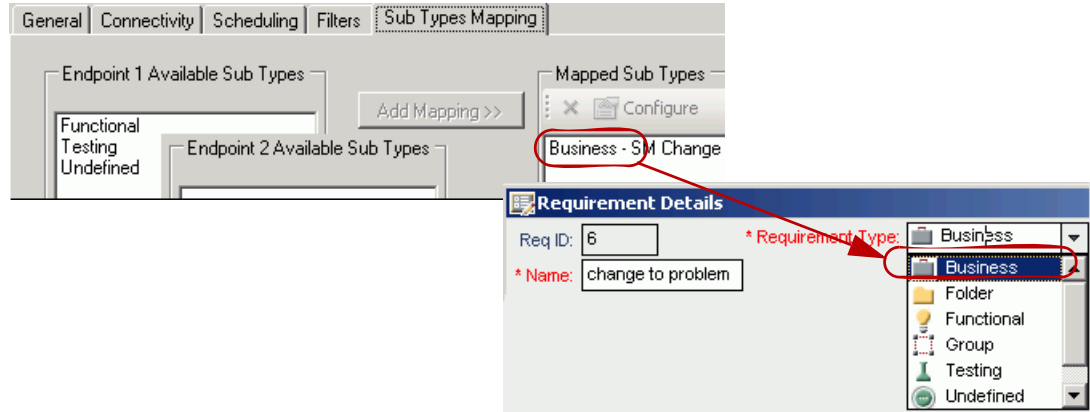
Check Connectivity

Use alternate root folder:
Requirements\ <your_specified_incoming_requirement_folder>

Requirements will be created in the specified folder in QC.



- 5 In the “Sub types mapping” tab, specify the type of requirements created from changes.



Field mappings

Basic field mappings are summarized below:

QC	Direction	SM	Constant value	Remarks
Change ID	<-	ChangeNumber		
Req ID	->	QCEntityID		Synchronize back on create: Yes
Created from			Created from SM/SC	

Example field mappings are shown in the following figure:

Mapped Fields			
Type	QC Field	Direction	SM ChangeManagement Field
	Name	<---->	Description
	Change ID	<---->	ChangeNumber
	Req ID	---->	QCEntityID
	Created from	<---->	Value: Created from SM/SC

QC Field <-> SM Field

The following table summarizes the QC <-> SM field mappings. The highlighted rows are the required mappings.

Table 8 SM Change -> QC Requirement Mappings

QC Len	QC DB Name	QC Type	QC/QCS Label	QCS QC Type	Dir	QCS SM type	QCS Name / SM WSDL Caption	SM WSDL Type	SM DB Name	SM DB type		SM Len
										SM7	SC6	
40	RQ_USER_01	String	Change ID	String	<-	String	ChangeNumber	StringType	header,number	Char	Text	100
10	RQ_REQ_ID	Number	Req ID (1)	Number	->	Number	QCEntityID	IntType	qcintegration.id	Num	Decimal	xx
255	RQ_REQ_COMMENT	Memo	Description	String	<->	String	Description	--	description.structure,description	Char	Text	xx



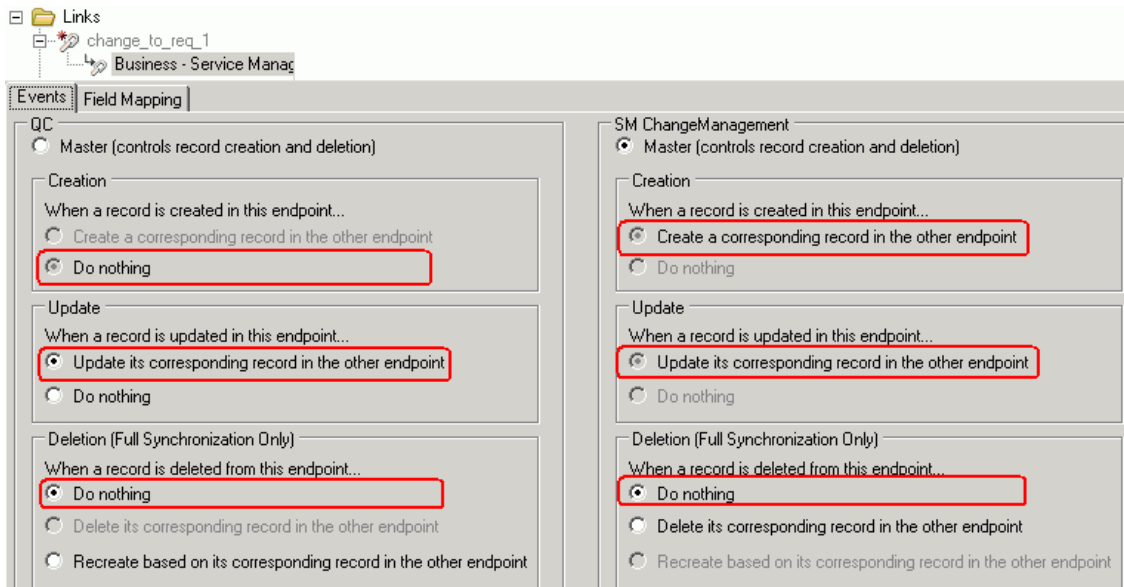
Check for QCEntityID mapping property **Synchronize back on create**.

Events

The following table lists the events.

Events Tab Settings	QC Action (Event)	SM Action (Event)
Creation	Do nothing.	Create a corresponding record in the other endpoint.
Update	Update its corresponding record in the other endpoint.	Update its corresponding record in the other endpoint.
Deletion	Do nothing.	Do nothing.

The following diagram shows the settings.



Test

To test the link, follow these steps.

▶ The following is just an example. The exact steps required on your system may differ significantly. The phase in which the tab for QC Integration appears may be different on your system.

- 1 Save the configuration (an integrity check is automatically run).
- 2 Click **Enable Link**.
- 3 Create a Service Manager change (the category of the change depends on each Service Manager customizations; RFC is used in this example).

The screenshot shows a table with columns 'Category' and 'Description'. The 'RFC' category is selected, with 'Request For Change' in the description. Below the table is a form titled 'Request For Change' with tabs for 'General' and 'Description'. The 'Description' tab is active, showing a text area with the text 'Change to Requirement'.

- 4 Change the phase to **Building**. The “QC Integration tab” appears.

The screenshot shows a table titled 'Valid Change Phases' with columns 'Phase Name' and 'Description'. The 'Building' phase is selected, with 'RFC Building' in the description.

Phase Name	Description
Assessment	Assessment
Building	RFC Building
RFC Implementation	Place Changes Into Service
RFC Testing	RFC Testing

- 5 Select **Forward as Requirement**.

The screenshot shows the 'QC Integration' tab in the form. It contains three fields: 'Forward to QC:' with a dropdown menu set to '1 - Forward as Requirement', 'Def/Req ID:' with an empty text box, and 'Server/Domain/Project:' with a dropdown menu set to 'localhost/DEFAULT/Demo'.

- 6 Synchronize.

The screenshot shows a progress bar with buttons for 'Cancel Current Task', 'View Report', 'Refresh Progress', and 'Auto Refresh'. Below the progress bar is a log of the synchronization process:

```
Running: Connecting to endpoint 1...
Running: Connecting to endpoint 2...
Running: Querying non filtered set
Running: Handling endpoint 1 - Processing entity #1 of #1 in the Create list, (Total: passed = 0, failed = 0)
Passed: Disconnecting...
Completed : Passed
```

7 View the requirement in QC.

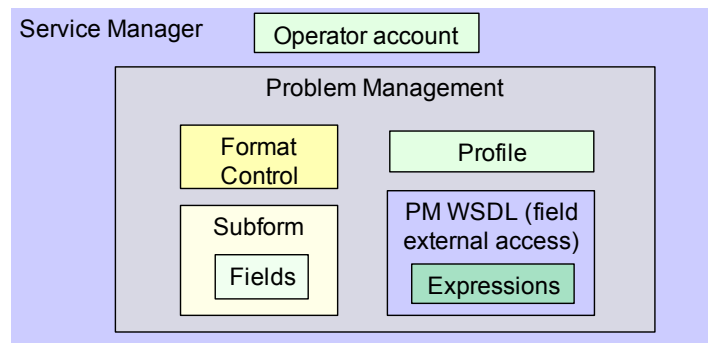
The screenshot displays a software interface with a sidebar on the left containing icons for Releases, Requirements, Test Plan, Test Lab, and Defects. The main window has a menu bar with 'Requirements', 'Edit', 'View', 'Favorites', and 'Analysis'. Below the menu is a toolbar with various icons. A table lists requirements:

Name	Direct Cover Status	Req ID	Author
Requirements	-----	0	
SM Incoming Changes	-----	1	admin
change - requirement	-----	2	

Below the table is a 'Requirement Details' window. It shows 'Req ID: 2', '* Name: change - requirement', and '* Requirement Type: Business'. The 'Details' tab is active, showing 'SM Integration (Details)' with 'Change ID: 18' and 'Created from: Created from SM/SC'. A sidebar on the left of the details window includes 'Details' and 'Requirements Traceability'.

8 SM Problem -> QC Defect

Customizing Service Manager/ServiceCenter for Problem Management



To customize problem management, follow these steps:

- 1 Add Fields
- 2 Specify Field External Access
- 3 Create Subform
- 4 Add Subform to Form
- 5 Add Format Control Calculations/Validations

Add Fields

Add the following fields to the table "rootcause" (**Menu Navigation** → **Tailoring** → **Database Dictionary**). The values shown are required (do not change).

Field	Type	
	Service Manager 7.0x/7.10	ServiceCenter
qcintegration.type	Character	Text
qcintegration.id	Number	Decimal
qcintegration.project	Character	Text

▶ The data type requirements for SM fields are described in [Matching Types](#) on page 30.

Specify Field External Access

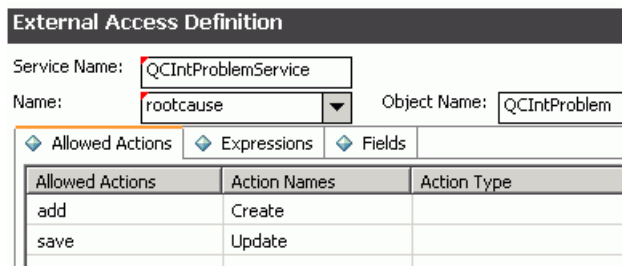
On Service Manager 7.0x/7.10

- 1 Create a customized External Access Definition QCIntProblemService (**Menu Navigation** → **Tailoring** → **WSDL configuration** on Service Manager 7.0x; **Menu Navigation** → **Tailoring** → **Web Services** → **WSDL configuration** on Service Manager 7.10).

- Service Name: QCIntProblemService
- Name: rootcause
- Object Name: QCIntProblem
- Allowed Action: add / Create
- Allow Action: save / Action Name: Update

► The above values are required (do not change).

This is shown in the following diagram.



External Access Definition

Service Name:

Name: Object Name:

Allowed Actions Expressions Fields

Allowed Actions	Action Names	Action Type
add	Create	
save	Update	

- 2 Enable the required fields in the web service.

Field	Caption	Type	Remarks
id	ProblemID	StringType	
sysmodtime	Modified	DateTimeType	
qcintegration.id	QCEntityID	IntType	

► The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on). The above values are required (do not change).

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Allowed Actions Expressions Fields

Field	Caption	Type
qcintegration.id	QCEntityID	IntType
id	ProblemID	StringType
sysmodtime	Modified	DateTimeType
qcintegration.project	QCProject	StringType
incident.category	Category	StringType
subrcatentrv	SubCategory	StringType

3 Define Web service expression.

➤ Problem Management requires an activity update provided with each save and for better flow, this activity update will be hardcoded with the following expressions.

No	Expression
1	cleanup(\$pm.activity);cleanup(\$rc.update);if same(update in \$L.file, update in \$L.file.save) then (\$L.need.to.update=true)
2	\$rc.update=update in \$L.file;if (denu(\$rc.update)={}) then (\$rc.update={"QC update sent"})
3	if (\$L.need.to.update=true) then (\$rc.update={"QC update sent"})
4	update in \$L.file=update in \$L.file.save

➤ Expressions 1~4 are for fixing the web services Problem Management update issue. For more information, see *SCR 41399*.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Object Name:

Allowed Actions Expressions Fields

Expressions

```
cleanup($pm.activity);cleanup($rc.update);if same(update in $L.file, update in $L.file.save) then ($L.need.to.update=true)
$rc.update=update in $L.file;if (denu($rc.update)={}) then ($rc.update={"QC update sent"})
if ($L.need.to.update=true) then ($rc.update={"QC update sent"})
update in $L.file=update in $L.file.save
current.phase in $L.file="Problem Investigation and Diagnosis"
category in $L.file="ITIL"
```

On ServiceCenter

- 1 Change the settings of these fields in **System Definition** → **Tables** → **rootcause** → **Fields and keys definitions for the rootcause table**.

No.	Field	Include in API	Field name in API	Field data type in API	Remarks
1	id	Y	ProblemID	StringType	
2	sysmodtime	Y	Modified	DateTimeType	
3	qcintegration.id	Y	QCEntityID	IntType	



The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on).

- 2 Fill in the Process Name **rca.save** in **Menu navigation** → **Utilities** → **Tools** → **Document Engine** → **Process**, and then click **Search**.
- 3 Change the name to **rca.qcupdate** and click **Add**, then append lines in **Initial Expressions** tab and click **Save**.

No.	Expression
1	cleanup(\$pm.activity);cleanup(\$rc.update);if same(update in \$L.file, update in \$L.file.save) then (\$L.need.to.update=true)
2	\$rc.update=update in \$L.file;if (denu(\$rc.update)={}) then (\$rc.update={"QC update sent"})
3	if (\$L.need.to.update=true) then (\$rc.update={"QC update sent"})
4	update in \$L.file=update in \$L.file.save

Process Definition

Process Name:

Save Cursor Position? Run Standard Process when complete?

Run in Window? Window Title:

◆ Initial Expressions ◆ Initial Javascript ◆ RAD ◆ Final Expressions ◆ Final Javascript ◆ Next Process

```

if same(nullsub(full.name in $G.rc.environment, full.name in $G.rc.global.environment), true) then ($L.operator=nullsub($lo.ufname, nullsu...
$L.stamp=str(tod()+")"+"+$L.operator+");"
if ($rc.update={} or $rc.update={""}) then ($rc.update=NULL);if ($kne.update={} or $kne.update={""}) then ($kne.update=NULL);if ($p...
if (update in $L.file=NULL) then (update in $L.file={""})
if ($G.bg and not null($G.bg.activity.type) then ($rc.update=nullsub($rc.update, $G.bg.activity.text);$rc.update=nullsub($rc.update, "E...
if (filename($L.file)~="rootcausetask") then if (status in $L.file="Past Due" and expected.resolution.time in $L.file>tod()) then (status in $L...

$L.save.status=status in $L.file
if (status in $L.file="Open" or status in $L.file="Reopened" and $reopen.flag=false) then (status in $L.file="Updated")
$reopen.flag=false

cleanup($pm.activity);cleanup($rc.update);if same(update in $L.file, update in $L.file.save) then ($L.need.to.update=true)
$rc.update=update in $L.file;if (denull($rc.update)={}) then ($rc.update={"QC update sent"})
if ($L.need.to.update=true) then ($rc.update={"QC update sent"})
update in $L.file=update in $L.file.save

```

- 4 Fill in the state name **rca.view** in **Menu navigation** → **Utilities** → **Tools** → **Document Engine** → **States**, click **Search**. Add one record in **Non-base methods** table, then click **Save**.

Display Action	Process Name	Condition	Save First
qcupdate	rca.qcupdate	\$L.mode~="close" and \$L.mode~#"add"	false

- 5 Search for the name **rootcause** in **Menu navigation** → **Toolkit** → **WSDL Configuration**, then update the External Access Definition as follows based on the table **rootcause**.

No.	Field	Value
1	Service Name	QCIntProblemService
2	Object Name	QCIntProblem
3	Allowed Actions	add/Create (Action Names)
4	Allowed Actions	qcupdate/Update (Action Names)

-  Delete all the Allowed Actions without Action Name.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Object Name:

Allowed Actions
 Expressions
 Data Policy

Allowed Actions	Action Names
add	Create
qcupdate	Update

Create Subform

- 1 Create Global List.

Create a global list (**Menu Navigation** → **Tailoring** → **Tailoring Tools** → **Global Lists** on Service Manager 7.0x/7.10; **Menu navigation** → **Utilities** → **Tools** → **Global Lists** on ServiceCenter) with following parameters:

No.	Parameter	Value	Remarks
1	List Name	SMQC Integration PM Project List	
2	Regen Every	1 00:00:00	
3	Build List on Startup?	Yes	Check box
4	List Variable	\$G.qcintegration.problem.project	
5	User Defined List?	Yes	Check box
6	Value List	{"server1/domain1/project1", "server2/domain2/project2"}	Change to the values for your system Note: No spaces between slashes.

Save this global list and click **Rebuild Global List** from menu.

- 2 Create a subform.

Create subform **pm.qcint.subform** without the Form Wizard (**Menu Navigation** → **Tailoring** → **Forms Designer** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Toolkit** → **Forms Designer** on ServiceCenter) with the following components.

No.	Component	Properties
1	Label	Caption: "Synchronize with QC:"
2	Combo Box	Input: "qcintegration.type" Value List: "0;1;" Display List: "0 - Not Synchronize;1 - Synchronize with QC Defect" Select Only: "Yes" Read-Only Condition: "[\$qcint.type.readonly]"
3	Label	Caption: "Defect ID:"

No.	Component	Properties
4	Text	Input: "qcintegration.id" Read-Only: "Yes"
5	Label	Caption: "Server/Domain/Project:"
6	Combo Box	Input: "qcintegration.project" Value List: "\$G.qcintegration.problem.project" Read-Only Condition: "[\$qcint.project.readonly]" Mandatory Condition: "[qcintegration.type]>0"

This is shown in the following diagram.

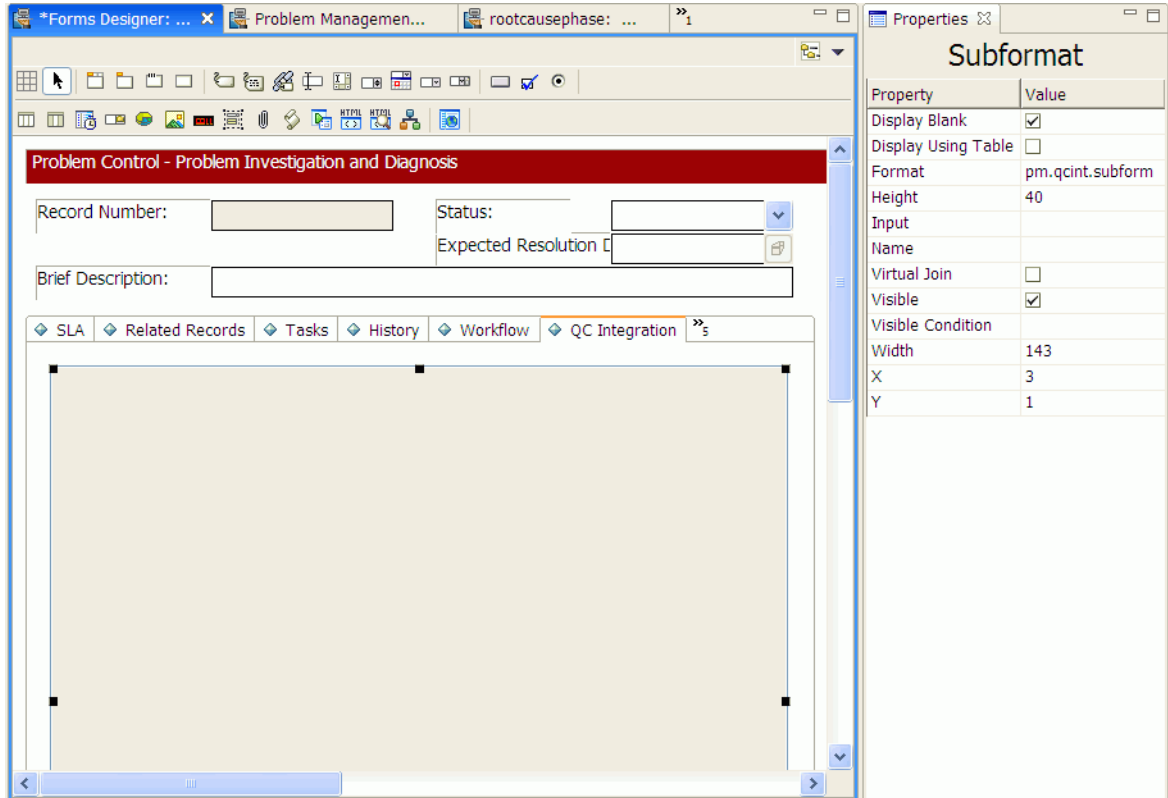
The screenshot shows a form designer window with a toolbar at the top containing icons for grid, mouse, folder, document, undo, redo, copy, paste, zoom, and other controls. Below the toolbar, there are four form fields arranged vertically:

- Synchronize with QC:** A dropdown menu with a downward arrow.
- Defect ID:** A text input field.
- Server/Domain/Project:** A dropdown menu with a downward arrow.
- Created from:** A text input field.

Add Subform to Form

- 1 Open the default form of one phase of Problem Management via Forms Designer (PM.pc.ident.and.class is used as an example in ServiceCenter 6.2/Service Manager 7.0x).
- 2 Add a Notebook Tab with caption "QC Integration".

- 3 Add a Subform to the new tab with format "pm.qcint.subform".



- 4 Save the changes.

➤ If the error message "Format 'pm.qcint.subform' not found (display, show.rio)" appears, then restart the SM server to enable the subform.

Add Format Control Calculations/Validations

- 1 Open the corresponding format control of the form, such as PM.pc.ident.and.class.
- 2 Click **Calculations**.
- 3 Add two records with the following values:

Record	Parameter	Value
1	display	true
	initial	true
	calculation	<code>\$qcint.type.readonly=2;if (qcintegration.type in \$file~=0) then (\$qcint.type.readonly=1)</code>
2	display	true
	initial	true
	calculation	<code>\$qcint.project.readonly=2;if (qcintegration.type in \$file~=0 and not null(qcintegration.project in \$file)) then (\$qcint.project.readonly=1)</code>

- 4 Click **Validations**.
- 5 Add one record with the following values:

No.	Parameter	Value
1	Validation	not null(qcintegration.project in \$file)
2	Message	The Server/Domain/Project is required.
3	Add	qcintegration.type in \$file~=0
4	Update	qcintegration.type in \$file~=0
5	Set Focus to	qcintegration.project

- 6 Save your changes.

Customizing Quality Center Defects Module

To customize Quality Center Defects module, follow these steps:

- 1 Add Fields
- 2 Add Tabs
- 3 Add Fields to Tabs

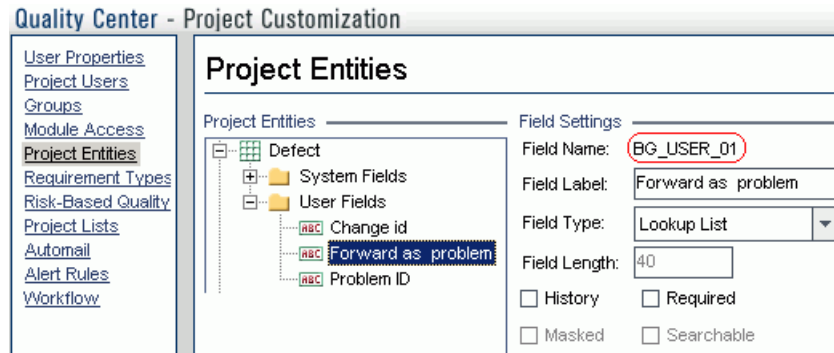
Add Fields

To add the required fields for defect customization, follow these steps.

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Add the following fields for the defect entity in project entities (XX XY are sequential numbers auto-generated by QC).

Field Name	Field Label	Field Type	Remarks
BG_USER_XX	Problem ID	String	
BG_USER_XY	Created from	String	

The following diagram shows an example project entity.

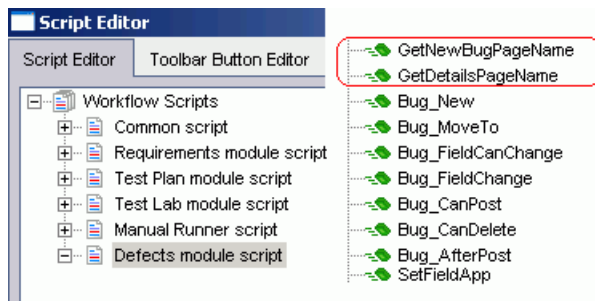
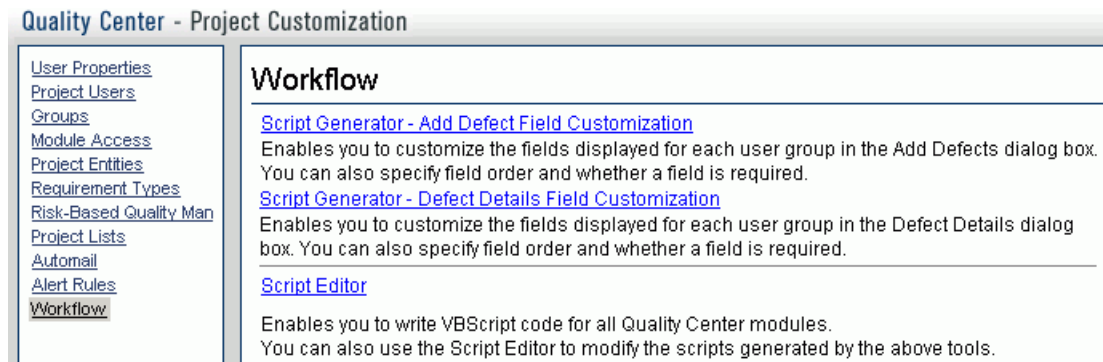


▶ The data type requirements for QC fields is described in [Matching Types](#) on page 30.

Add Tabs

To add tabs to the defect form and show fields on these tabs, follow these steps.

- 1 In "QC - Project Customization" click **Workflow**.
- 2 Click **Workflow** → **Script Editor**.
- 3 Choose **Defects module script**.



- 4 Add the following code to the **GetNewBugPageName** event procedure (which is triggered before QC opens the Add Defect dialog box).

```
select case PageNum
case "2"
    GetNewBugPageName = "SM Integration (New)"
```

```
end select
```

➤ 2 specifies tab 2 (the second tab). For a new bug, the tab name is **SM Integration (New)**.

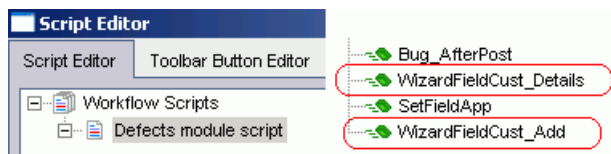
- 5 Add the following code to the **GetDetailsPageName** event procedure (which is triggered before QC displays Defect Details dialog box).

```
select case PageNum
  case "2"
    GetDetailsPageName = "SM Integration (Details)"
end select
```

➤ 2 specifies tab 2 (the second tab). For an existing defect, the tab name is **SM Integration (Details)**.

Add Fields to Tabs

- 1 If WizardFieldCust_Details and WizardFieldCust_Add are not found in the list, then do the following to generate these two methods.
 - a **Script Generator - Add Defect Field Customization**
 - b **Script Generator - Defect Details Field Customization**



- 2 Add the following code to the WizardFieldCust_Details event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
```

The parameters are

- Field name (BG_USER_XX, where XX = two digits)
- Visible (True)
- Required (False)
- Page number (start from 0)
- View order (start from 0)

- 3 Add the following code to the WizardFieldCust_Add event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
```

- 4 Set the **ReadOnly** fields by adding the following lines to Bug_New and Bug_Moveto subroutines:

```
Bug_Fields.Field("BG_USER_XY").IsReadOnly=True
Bug_Fields.Field("BG_USER_XZ").IsReadOnly=True
```

- 5 Exit customization (save changes).

Configuring Links in QC Synchronizer

This section describes how to create and test a link.

- Specify Endpoints / Type of Link
- Field Mappings
- Events
- Test

Specify Endpoints / Type of Link

Specify the connection properties as described in [Create a Link](#) on page 26 with the following settings specific for this type of link:






- 1 Step 1 endpoint 2 type = **SM ProblemManagement**.
- 2 Step 3 service URL =
http://<service_manager_host>:<port>/sc62server/PWS/QCIntProblemService.wsdl
- 3 Step 4 select entity types = **Problem by Defect** (this is the only available selection).

Field Mappings

Basic field mappings are summarized below:

QC	Directions	SM	Constant Value	Remarks
Problem ID	<-	ProblemID		
Defect ID	->	QCEntityID		Synchronize back on create: Yes
Created from			Created from SM/SC	

Example field mappings are shown in the following figure:

Mapped Fields			
Type	QC Field	Direction	SM ProblemManagement Field
	Summary	<--->	Description
	Defect ID	--->	QCEntityID
	Severity	<--->	Severity
	Problem ID	<--->	ProblemID
	Created from	<--->	Value: Created from SM/SC

Events

The following table lists three events:

Events Tab Settings	QC Action (Event)	SM Action (Event)
Creation	Do nothing.	Create a corresponding record in the other endpoint.
Update	Update its corresponding record in the other endpoint.	Update its corresponding record in the other endpoint.
Deletion	Do nothing.	Do nothing.

The following diagram shows the settings:

The diagram shows two side-by-side configuration panels. The left panel is titled 'QC' and the right panel is titled 'SM ProblemManagement'. Each panel has three sections: 'Creation', 'Update', and 'Deletion (Full Synchronization Only)'. In the QC panel, the 'Do nothing' radio button is selected for all three sections. In the SM ProblemManagement panel, the 'Create a corresponding record in the other endpoint' radio button is selected for 'Creation' and 'Update', and the 'Do nothing' radio button is selected for 'Deletion'. Red boxes highlight the selected radio buttons in each section.

Test

To test the link, follow these steps.

➤ The following is just an example. The exact steps required on your system may differ significantly. The phase in which the tab for QC Integration appears may be different on your system.

- 1 Save the configuration (an integrity check is automatically run).
- 2 Click **Enable Link**.

3 Create a problem and select "1 - Synchronize with QC Defect".

Problem PM0016 has been opened.

Problem Control - Problem Identification and Classification

Record Number: Status:
Expected Resolution Date:

Brief Description:

Classification Activities Attachments Related Records History Workflow **QC Integration**

Synchronize with QC:
Defect ID:
Server/Domain/Project:

4 Synchronize.

Cancel Current Task View Report Refresh Progress Auto Refresh

Running: Querying non-filtered set...
Running: Handling endpoint 1 - Processing entity #1 of #1 in the Create list. (Total: passed = 0, failed = 0)
Passed: Disconnecting...
Completed: Passed

5 View the problem in SM.

Problem Control - Problem Identification and Classification

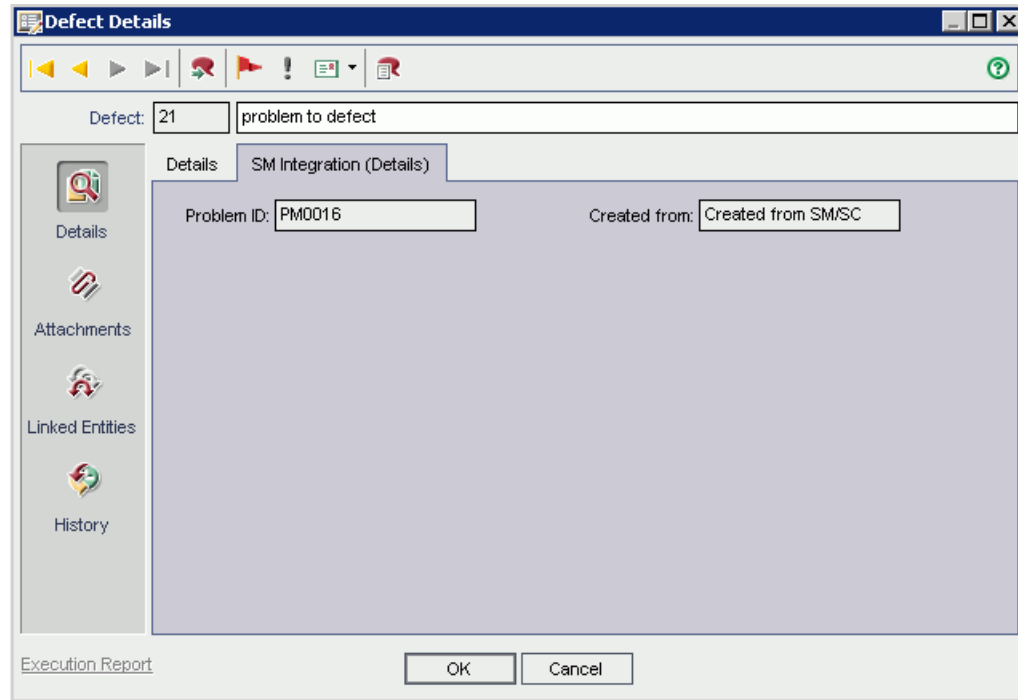
Record Number: Status:
Expected Resolution Date:

Brief Description:

Classification Activities Attachments Related Records History Workflow **QC Integration**

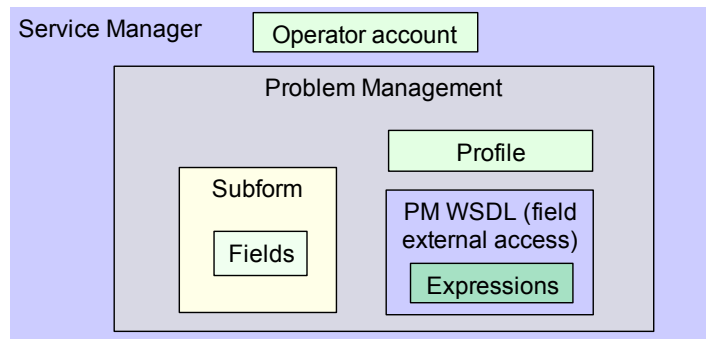
Synchronize with QC:
Defect ID:
Server/Domain/Project:

6 View the defect in QC.



9 QC Defect -> SM Problem

Customizing Service Manager/ServiceCenter for Problem Management



To customize problem management, follow these steps:

- 1 Add Fields
- 2 Specify Field External Access
- 3 Create Subform
- 4 Add Subform to Form

Add Fields

Add the following fields to the table "rootcause" (**Menu Navigation** → **Tailoring** → **Database Dictionary**). The values shown are required (do not change).

Field	Type	
	Service Manager 7.0x/7.10	ServiceCenter
qcintegration.type	Character	Text
qcintegration.id	Number	Decimal
qcintegration.project	Character	Text
qcintegration.created.from	Character	Text

► The data type requirements for SM fields are described in [Matching Types](#) on page 30.

Specify Field External Access

On Service Manager 7.0x/7.10

- 1 Create a customized External Access Definition QCIntProblemService (**Menu Navigation** → **Tailoring** → **WSDL configuration** on Service Manager 7.0x; **Menu Navigation** → **Tailoring** → **Web Services** → **WSDL configuration** on Service Manager 7.10).

- Service Name: QCIntProblemService
- Name: rootcause
- Object Name: QCIntProblem
- Allowed Action: add / Create
- Allow Action: save / Action Name: Update

➤ The above values are required (do not change).

This is shown in the following diagram.

External Access Definition

Service Name:

Name: Object Name:

Allowed Actions | Expressions | Fields

Allowed Actions	Action Names	Action Type
add	Create	
save	Update	

- 2 Enable the required fields in the web service.

Field	Caption	Type	Remarks
id	ProblemID	StringType	
sysmodtime	Modified	DateTimeType	
qcingtegration.id	QCEntityID	IntType	
qcingtegration.project	QCProject	StringType	
qcingtegration.type	QCIntegrationType	StringType	
qcingtegration.created .from	CreatedFrom	StringType	
current.phase	CurrentPhase	StringType	Optional for Service Manager 7.10.
category	WorkFlowType	StringType	Optional for Service Manager 7.10.

➤ The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on). The above values are required (do not change).

- ▶ Be sure to enable other required fields of Problem in the web service for your system.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Allowed Actions Expressions Fields

Field	Caption	Type
qcintegration.id	QCEntityID	IntType
id	ProblemID	StringType
sysmodtime	Modified	DateTimeType
qcintegration.project	QCProject	StringType
incident.category	Category	StringType
subrcatentrv	SubCategory	StringType

3 Define Web service expression.

- ▶ Problem Management requires an activity update provided with each save and for better flow, this activity update will be hardcoded with the following expressions.

No	Expression
1	cleanup(\$pm.activity);cleanup(\$rc.update);if same(update in \$L.file, update in \$L.file.save) then (\$L.need.to.update=true)
2	\$rc.update=update in \$L.file;if (denu(\$rc.update)={}) then (\$rc.update={"QC update sent"})
3	if (\$L.need.to.update=true) then (\$rc.update={"QC update sent"})
4	update in \$L.file=update in \$L.file.save

- ▶ Expressions 1~4 are for fixing the web services Problem Management update issue. For more information, see *SCR 41399*.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Object Name:

Allowed Actions Expressions Fields

Expressions

```
cleanup($pm.activity);cleanup($rc.update);if same(update in $L.file, update in $L.file.save) then ($L.need.to.update=true)
$rc.update=update in $L.file;if (denu($rc.update)={}) then ($rc.update={"QC update sent"})
if ($L.need.to.update=true) then ($rc.update={"QC update sent"})
update in $L.file=update in $L.file.save
current.phase in $L.file="Problem Investigation and Diagnosis"
category in $L.file="ITIL"
```

On ServiceCenter

- 1 Change the settings of these fields in **System Definition** → **Tables** → **rootcause** → **Fields and keys definitions for the rootcause table**.

No.	Field	Include in API	Field name in API	Field data type in API	Remarks
1	id	Y	ProblemID	StringType	
2	sysmodtime	Y	Modified	DateTimeType	
3	qcintegration.id	Y	QCEntityID	IntType	
4	qcintegration.p roject	Y	QCProject	StringType	
5	qcintegration.t ype	Y	QCIntegrationT ype	StringType	
6	qcintegration.cr eated.from	Y	CreatedFrom	StringType	
7	current.phase	Y	CurrentPhase	StringType	
8	category	Y	WorkflowType	StringType	



The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on).

- 2 Fill in the Process Name **rca.save** in **Menu navigation** → **Utilities** → **Tools** → **Document Engine** → **Process**, and then click **Search**.
- 3 Change the name to **rca.qcupdate** and click **Add**, then append lines in **Initial Expressions** tab and click **Save**.

No.	Expression
1	cleanup(\$pm.activity);cleanup(\$rc.update);if same(update in \$L.file, update in \$L.file.save) then (\$L.need.to.update=true)
2	\$rc.update=update in \$L.file;if (dnull(\$rc.update)={}) then (\$rc.update={"QC update sent"})
3	if (\$L.need.to.update=true) then (\$rc.update={"QC update sent"})
4	update in \$L.file=update in \$L.file.save

Process Definition

Process Name:

Save Cursor Position? Run Standard Process when complete?

Run in Window? Window Title:

◆ Initial Expressions ◆ Initial Javascript ◆ RAD ◆ Final Expressions ◆ Final Javascript ◆ Next Process

```

if same(nullsub(full.name in $G.rc.environment, full.name in $G.rc.global.environment), true) then ($L.operator=nullsub($lo.ufname, nullsu...
$L.stamp=str(tod())+" ("+$L.operator+");"
if ($rc.update={} or $rc.update="") then ($rc.update=NULL);if ($kne.update={} or $kne.update="") then ($kne.update=NULL);if ($p...
if (update in $L.file=NULL) then (update in $L.file="")
if ($G.bg and not null($G.bg.activity.type)) then ($rc.update=nullsub($rc.update, $G.bg.activity.text);$rc.update=nullsub($rc.update, "E...
if (filename($L.file)~="rootcausetask") then if (status in $L.file="Past Due" and expected.resolution.time in $L.file>tod()) then (status in $L...

$L.save.status=status in $L.file
if (status in $L.file="Open" or status in $L.file="Reopened" and $reopen.flag=false) then (status in $L.file="Updated")
$reopen.flag=false

cleanup($pm.activity);cleanup($rc.update);if same(update in $L.file, update in $L.file.save) then ($L.need.to.update=true)
$rc.update=update in $L.file;if (denull($rc.update)={}) then ($rc.update={"QC update sent"})
if ($L.need.to.update=true) then ($rc.update={"QC update sent"})
update in $L.file=update in $L.file.save

```

- 4 Fill in the state name **rca.view** in **Menu navigation** → **Utilities** → **Tools** → **Document Engine** → **States**, click **Search**. Add one record in **Non-base methods** table, then click **Save**.

Display Action	Process Name	Condition	Save First
qcupdate	rca.qcupdate	\$L.mode~="close" and \$L.mode~#"add"	false

- 5 Search for the name **rootcause** in **Menu navigation** → **Toolkit** → **WSDL Configuration**, then update the External Access Definition as follows based on the table **rootcause**.



No.	Field	Value
1	Service Name	QCIntProblemService
2	Object Name	QCIntProblem
3	Allowed Actions	add/Create (Action Names)
4	Allowed Actions	qcupdate/Update (Action Names)

-  Delete all the Allowed Actions without Action Name.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:  

Object Name:

Allowed Actions
 Expressions
 Data Policy

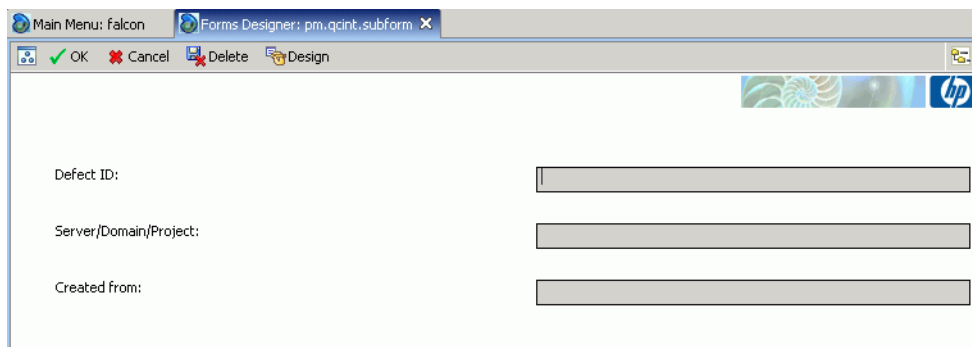
Allowed Actions	Action Names
add	Create
qcupdate	Update

Create Subform

Create subform **pm.qcint.subform** without the Form Wizard (**Menu Navigation** → **Tailoring** → **Forms Designer** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Toolkit** → **Forms Designer** on ServiceCenter) with the following components.

No.	Component	Properties
1	Label	Caption: "Defect ID:"
2	Text	Input: "qcintegration.id" Read-Only: "Yes"
3	Label	Caption: "Server/Domain/Project:"
4	Text	Input: "qcintegration.project" Read-Only: "Yes"
5	Label	Caption: "Created from:"
6	Text	Input: "qcintegration.created.from" Read-Only: "Yes"

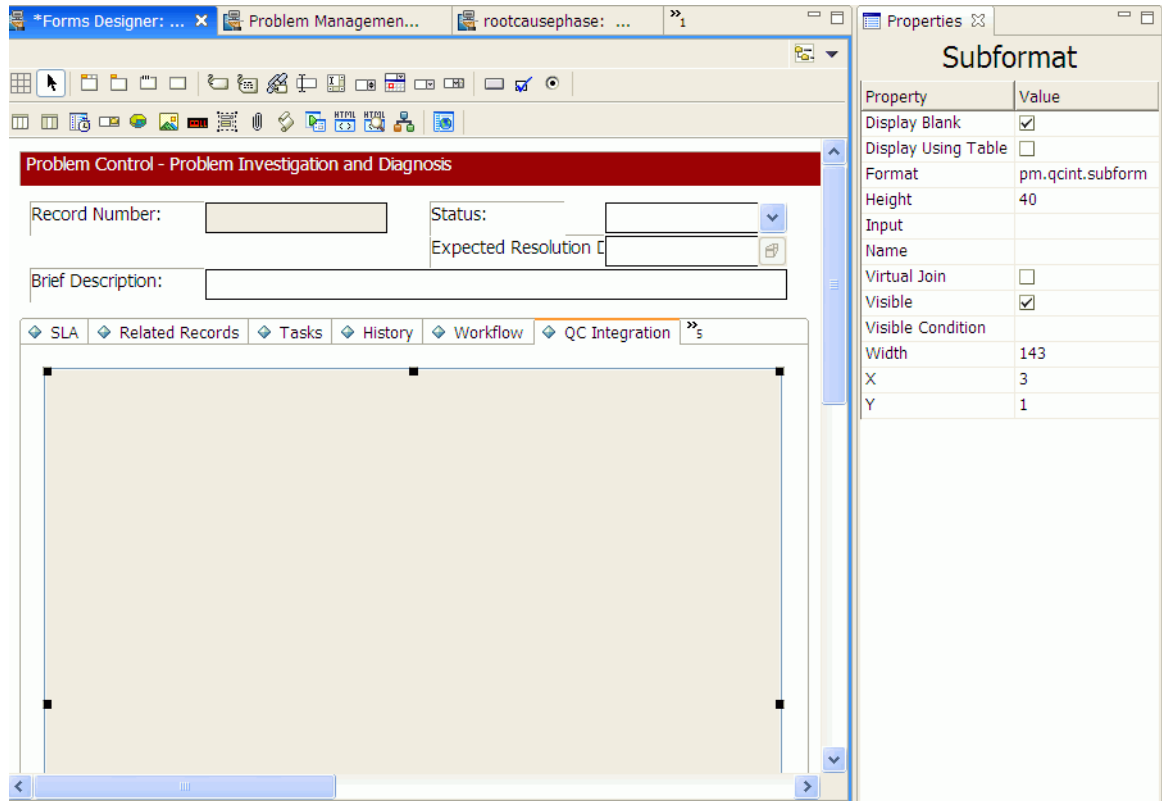
This is shown in the following diagram.



Add Subform to Form

- 1 Open the default form of one phase of Problem Management via Forms Designer (PM.pc.ident.and.class is used as an example in ServiceCenter 6.2/Service Manager 7.0x).
- 2 Add a Notebook Tab with caption "QC Integration".

- 3 Add a Subform to the new tab with format "pm.qcint.subform".



- 4 Save the changes.

➤ If the error message "Format 'pm.qcint.subform' not found (display, show.ric)" appears, then restart the SM server to enable the subform.

Customizing Quality Center Defects Module

To customize Quality Center Defects module, follow these steps:

- 1 Add Fields
- 2 Add Tabs
- 3 Add Fields to Tabs
- 4 Create a View

Add Fields

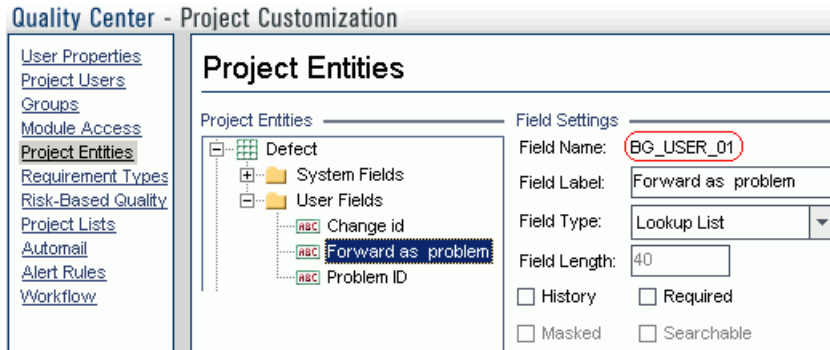
To add the required fields for defect customization, follow these steps.

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.

- 3 Add the following fields for the defect entity in project entities (XX XY are sequential numbers auto-generated by QC).

Field Name	Field Label	Field Type	Remarks
BG_USER_XX	Synchronize with SM Problem	Lookup List/YesNo	Select "Verify Value" checkbox
BG_USER_XY	Problem ID	String	

The following diagram shows an example project entity.



- The data type requirements for QC fields is described in [Matching Types](#) on page 30.

Add Tabs

To add tabs to the defect form and show fields on these tabs, follow these steps.

- 1 In "QC - Project Customization" click **Workflow**.
- 2 Click **Workflow** → **Script Editor**.

- 3 Choose **Defects module script**.

- 4 Add the following code to the **GetNewBugPageName** event procedure (which is triggered before QC opens the Add Defect dialog box).

```
select case PageNum
  case "2"
    GetNewBugPageName = "SM Integration (New)"
end select
```

- 2 specifies tab 2 (the second tab). For a new bug, the tab name is **SM Integration (New)**.

- 5 Add the following code to the **GetDetailsPageName** event procedure (which is triggered before QC displays Defect Details dialog box).

```
select case PageNum
  case "2"
    GetDetailsPageName = "SM Integration (Details)"
end select
```

- 2 specifies tab 2 (the second tab). For an existing defect, the tab name is **SM Integration (Details)**.

Add Fields to Tabs

- 1 If WizardFieldCust_Details and WizardFieldCust_Add are not found in the list, then do the following to generate these two methods.
 - a **Script Generator - Add Defect Field Customization**
 - b **Script Generator - Defect Details Field Customization**



- 2 Add the following code to the WizardFieldCust_Details event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
```

The parameters are

- Field name (BG_USER_XX, where XX = two digits)
- Visible (True)
- Required (False)
- Page number (start from 0)
- View order (start from 0)

- 3 Add the following code to the WizardFieldCust_Add event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
```

- 4 Set the **ReadOnly** fields by adding the following lines to Bug_New and Bug_Moveto subroutines:

```
if (Bug_Fields("BG_USER_XX").Value="Y") then
    Bug_Fields("BG_USER_XX").IsReadOnly=True
end if
Bug_Fields.Field("BG_USER_XY").IsReadOnly=True
```

The if loop above marks the field "Synchronize with SM Problem" as read only after selected and saved.

- 5 Exit customization (save changes).

Create a View

- 1 Log on to QC as the integration account - SMQCIntUser.
- 2 In the Defects module, click **View / Filter/Sort / Set Filters/Sort**. The purpose of this view is to let QC Synchronizer correctly filter those defects to be synchronized to SM as problems.
- 3 Set **Synchronize with SM Problem** to **Y**.
- 4 Add a view to Favorites:
 - Name: SMIntegrationView
 - Location: private



In the QC Synchronizer this view will be selected as the QC data filter. QC defects can not be forwarded to SM without the filter.

Verify

Details | SM Integration (Details)

Synchronize with SM Problem: Problem ID: PM0017

Created from: Created from SM/SC

Configuring Links in QC Synchronizer

This section describes how to create and test a link.

- [Specify Endpoints / Type of Link](#)
- [Filters](#)
- [Field Mappings](#)
- [Events](#)
- [Test](#)

Specify Endpoints / Type of Link

Specify the connection properties as described in [Create a Link](#) on page 26 with the following settings specific for this type of link:

- 1 Step 1 endpoint 2 type = **SM ProblemManagement**.
- 2 Step 3 service URL =
http://<service_manager_host>:<port>/sc62server/PWS/QCIntProblemService.wsdl
- 3 Step 4 select entity types = **Problem by Defect** (this is the only available selection).

Filters

In the Filters tab, select filter **SMIntegrationView** for QC endpoint. If the filter is not available, see [Create a View](#) on page 90.

QC

No Filter

Use filter (for creation events):

Private: SMIntegrationView

SM ProblemManagement

No Filter

Use filter (for creation events):

Field Mappings

Basic field mappings are summarized below:

QC	Dir	SM	Constant Value	Remarks
Problem ID	<-	ProblemID		
Defect ID	->	QCEntityID		Synchronize back on create: Yes
		QCIntegrationType	1	
		CreatedFrom	Created from QC	
QCProject	<-		(your setup)	This constant value should be the same as that for "QC Project" parameter in Connectivity tab.
	->	CurrentPhase	XXX	Note: Replace XXX with a valid phase name, such as "Problem Investigation and Diagnosis". This field mapping is optional for Service Manager 7.10.
	->	WorkFlowType	YYY	Note: Replace YYY with a valid category name, such as "ITIL" for demo data of SM 7.0x/SC 6.2; "BPPM" is for demo data of Service Manager 7.10. This field mapping is optional for Service Manager 7.10.

Example field mappings are shown in the following figure:

Mapped Fields			
Type	QC Field	Direction	SM ProblemManagement Field
	Summary	<---->	Description
	Defect ID	---->	QCEntityID
	Severity	<---->	Severity
	Problem ID	<---->	ProblemID
	Value: Created from Quality C...	---->	CreatedFrom
	Value: 1	---->	QCIntegrationType
	Value: AUTO	---->	AssignmentGroup
	Value: BOB.HELPDESK	---->	ProblemOwner
	Value: client system	---->	Category
	Value: software	---->	SubCategory
	Value: email client	---->	ProductType
	Value: outlook	---->	ProblemType
	Value: 4 - User	---->	Impact
	Value: Problem Identification ...	---->	CurrentPhase
	Value: localhost/DEFAULT/...	---->	QCProject
	Value: ITIL	---->	WorkFlowType

QC Field <-> SM Field

The following table summarizes the field mappings. The highlighted rows are the required mappings.

Table 9 QC Defect -> SM Problem Mappings

QC Len	QC DB Name	QC Type	QC/QCS Label	QCS QC Type	Dir	QCS SM type	QCS Name/ SM WSDL Caption	SM WSDL Type	SM DB Name / SM WSDL Field	SM DB type		SM Len
										SM7	SC6	
40	BG_USER_03	String	Problem ID	String	<-	String	Problem ID	StringType	id	Char	Text	100
10	BG_BUG_ID	Number	Defect ID	Number	->	Number	QCEntityID	IntType	qcintegration.id	Num	Decimal	xx
255	BG_SUMMARY	String	Summary	String	<->	String	Description	--	description	Char	Text	xx
70	BG_SEVERITY	Lookup List	Severity	Single value list	<->	Single value list	Severity	StringType	severity	Char	Text	40

Constants -> SM Fields

In order to create a problem in SM, constant values should be specified in field mapping. The constant values vary with different SM versions and SM customizations. Example constant field mappings for ServiceCenter are summarized below:

QCS SM constant value	Dir	QCS Name/SM WSDL Caption	SM WSDL Type	SM DB type		SM DB Name / SM WSDL Field	SM Len
				SM7	SC6		
1	->	QCIntegrationType	StringType	Char	Text	qcintegration.type	60
Created from Quality Center	->	CreatedFrom	StringType	Char	Text	qcintegration.created.from	60
(your setup)	->	QCProject	StringType	Char	Text	qcintegration.project	60
	->	Modified	DateTimeType	Date/time		sysmodtime	xx
AUTO	->	AssignmentGroup	StringType	Char	Text	assignment	50
client system	->	Category (1)	StringType	Char	Text	incident.category	40
BOB.HELPDESK	->	ProblemOwner	StringType	Char	Text	ticket.owner	40
exchange	->	ProblemType (1)	StringType	Char	Text	problem.type	40
email client	->	ProductType (1)	StringType	Char	Text	product.type	40
4 - User	->	Impact	StringType	Char	Text	initial.impact	50
software	->	SubCategory (1)	StringType	Char	Text	subcategory	40
ITIL	->	WorkFlowType	StringType	Char	Text	category	40
Problem Identification and Classification	->	CurrentPhase	StringType	Char	Text	current.phase	40

Events

The following table lists three events:

Events Tab Settings	QC Action (Event)	SM Action (Event)
Creation	Create a corresponding record in the other endpoint.	Do nothing.
Update	Update its corresponding record in the other endpoint.	Update its corresponding record in the other endpoint.
Deletion	Do nothing.	Do nothing.

The following diagram shows the settings.

The screenshot displays the 'Events' configuration window with two main panels: 'QC' and 'SM ProblemManagement'. Each panel has three sections: 'Creation', 'Update', and 'Deletion (Full Synchronization Only)'. In the 'QC' panel, the 'Creation' section has 'Create a corresponding record in the other endpoint' selected, 'Update' has 'Update its corresponding record in the other endpoint' selected, and 'Deletion' has 'Do nothing' selected. In the 'SM ProblemManagement' panel, the 'Creation' section has 'Do nothing' selected, 'Update' has 'Update its corresponding record in the other endpoint' selected, and 'Deletion' has 'Do nothing' selected. Red boxes highlight the selected radio buttons in each section.

Test

To test the link, follow these steps.

▶ The following is just an example. The exact steps required on your system may differ significantly. The phase in which the tab for QC Integration appears may be different on your system.

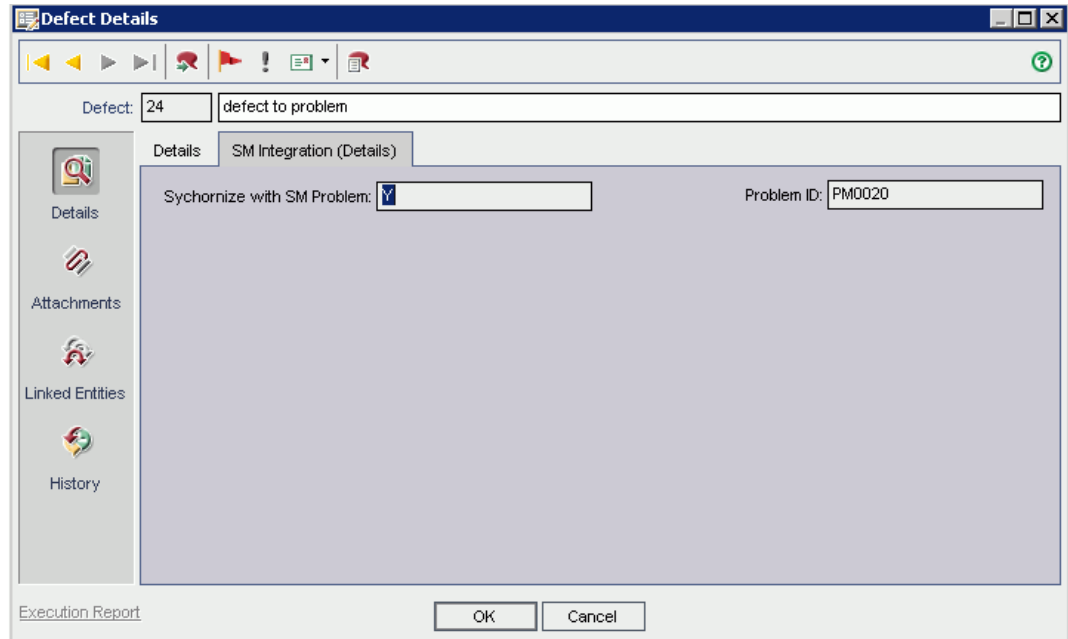
- 1 Save the configuration (an integrity check is automatically run).
- 2 Click **Enable Link**.

- 3 Create a defect and set "Synchronize with SM Problem" to true.

- 4 Synchronize.

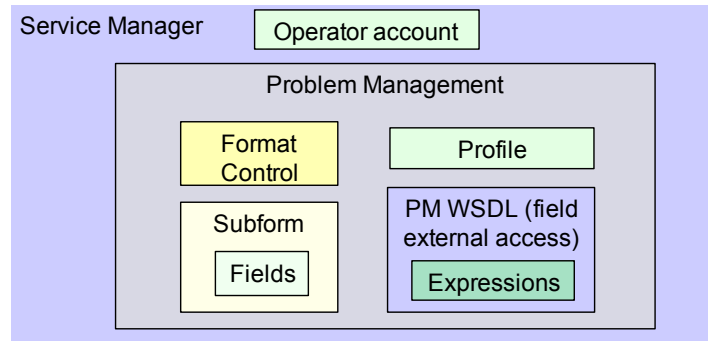
- 5 View the problem in SM.

6 View the defect in QC.



10 SM Problem <-> QC Defect

Customizing Service Manager/ServiceCenter for Problem Management



To customize problem management, follow these steps:

- 1 Add Fields
- 2 Specify Field External Access
- 3 Create Subform
- 4 Add Subform to Form
- 5 Add Format Control Calculations/Validations

Add Fields

Add the following fields to the table "rootcause" (**Menu Navigation** → **Tailoring** → **Database Dictionary**). The values shown are required (do not change).

Field	Type	
	Service Manager 7.0x/7.10	ServiceCenter
qcintegration.type	Character	Text
qcintegration.id	Number	Decimal
qcintegration.project	Character	Text
qcintegration.created.from	Character	Text

► The data type requirements for SM fields are described in [Matching Types](#) on page 30.

Specify Field External Access

On Service Manager 7.0x/7.10

- 1 Create a customized External Access Definition QCIntProblemService (**Menu Navigation** → **Tailoring** → **WSDL configuration** on Service Manager 7.0x; **Menu Navigation** → **Tailoring** → **Web Services** → **WSDL configuration** on Service Manager 7.10).

- Service Name: QCIntProblemService
- Name: rootcause
- Object Name: QCIntProblem
- Allowed Action: add / Create
- Allow Action: save / Action Name: Update

➤ The above values are required (do not change).

This is shown in the following diagram.

External Access Definition

Service Name:

Name: Object Name:

Allowed Actions | Expressions | Fields

Allowed Actions	Action Names	Action Type
add	Create	
save	Update	

- 2 Enable the required fields in the web service.

Field	Caption	Type	Remarks
id	ProblemID	StringType	
sysmodtime	Modified	DateTimeType	
qcintegration.id	QCEntityID	IntType	
qcintegration.project	QCProject	StringType	
qcintegration.type	QCIntegrationType	StringType	
qcintegration.created .from	CreatedFrom	StringType	
current.phase	CurrentPhase	StringType	Optional for Service Manager 7.10.
category	WorkFlowType	StringType	Optional for Service Manager 7.10.

➤ The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on). The above values are required (do not change).

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Allowed Actions Expressions Fields

Field	Caption	Type
qcintegration.id	QCEntityID	IntType
id	ProblemID	StringType
sysmodtime	Modified	DateTimeType
qcintegration.project	QCProject	StringType
incident.category	Category	StringType
subrcatcnrv	SubCategory	StringType

3 Define Web service expression.

➤ Problem Management requires an activity update provided with each save and for better flow, this activity update will be hardcoded with the following expressions.

No	Expression
1	cleanup(\$pm.activity);cleanup(\$rc.update);if same(update in \$L.file, update in \$L.file.save) then (\$L.need.to.update=true)
2	\$rc.update=update in \$L.file;if (denu(\$rc.update)={}) then (\$rc.update={"QC update sent"})
3	if (\$L.need.to.update=true) then (\$rc.update={"QC update sent"})
4	update in \$L.file=update in \$L.file.save

➤ Expressions 1~4 are for fixing the web services Problem Management update issue. For more information, see *SCR 41399*.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Object Name:

Allowed Actions Expressions Fields


Expressions

```
cleanup($pm.activity);cleanup($rc.update);if same(update in $L.file, update in $L.file.save) then ($L.need.to.update=true)
$rc.update=update in $L.file;if (denu($rc.update)={}) then ($rc.update={"QC update sent"})
if ($L.need.to.update=true) then ($rc.update={"QC update sent"})
update in $L.file=update in $L.file.save
current.phase in $L.file="Problem Investigation and Diagnosis"
category in $L.file="ITIL"
```

On ServiceCenter

- 1 Change the settings of these fields in **System Definition** → **Tables** → **rootcause** → **Fields and keys definitions for the rootcause table**.

No.	Field	Include in API	Field name in API	Field data type in API	Remarks
1	id	Y	ProblemID	StringType	
2	sysmodtime	Y	Modified	DateTimeType	
3	qcintegration.id	Y	QCEntityID	IntType	
4	qcintegration.project	Y	QCProject	StringType	
5	qcintegration.type	Y	QCIntegrationType	StringType	
6	qcintegration.created.from	Y	CreatedFrom	StringType	
7	current.phase	Y	CurrentPhase	StringType	
8	category	Y	WorkflowType	StringType	

 The caption value must be unique and alphanumeric (no spaces) with the first letter capitalized (AValidCaption123, AnotherValidCaption, and so on).

- 2 Fill in the Process Name **rca.save** in **Menu navigation** → **Utilities** → **Tools** → **Document Engine** → **Process**, and then click **Search**.
- 3 Change the name to **rca.qcupdate** and click **Add**, then append lines in **Initial Expressions** tab and click **Save**.

No.	Expression
1	cleanup(\$pm.activity);cleanup(\$rc.update);if same(update in \$L.file, update in \$L.file.save) then (\$L.need.to.update=true)
2	\$rc.update=update in \$L.file;if (dnull(\$rc.update)={}) then (\$rc.update={"QC update sent"})
3	if (\$L.need.to.update=true) then (\$rc.update={"QC update sent"})
4	update in \$L.file=update in \$L.file.save

Process Definition

Process Name:

Save Cursor Position? Run Standard Process when complete?

Run in Window? Window Title:

◆ Initial Expressions ◆ Initial Javascript ◆ RAD ◆ Final Expressions ◆ Final Javascript ◆ Next Process

```

if same(nullsub(full.name in $G.rc.environment, full.name in $G.rc.global.environment), true) then ($L.operator=nullsub($lo.ufname, nullsu...
$L.stamp=str(tod())+" ("+$L.operator+");"
if ($rc.update={} or $rc.update="") then ($rc.update=NULL);if ($kne.update={} or $kne.update="") then ($kne.update=NULL);if ($p...
if (update in $L.file=NULL) then (update in $L.file="")
if ($G.bg and not null($G.bg.activity.type)) then ($rc.update=nullsub($rc.update, $G.bg.activity.text);$rc.update=nullsub($rc.update, "E...
if (filename($L.file)~="rootcausetask") then if (status in $L.file="Past Due" and expected.resolution.time in $L.file>tod()) then (status in $L...

$L.save.status=status in $L.file
if (status in $L.file="Open" or status in $L.file="Reopened" and $reopen.flag=false) then (status in $L.file="Updated")
$reopen.flag=false

cleanup($pm.activity);cleanup($rc.update);if same(update in $L.file, update in $L.file.save) then ($L.need.to.update=true)
$rc.update=update in $L.file;if (denull($rc.update)={}) then ($rc.update={"QC update sent"})
if ($L.need.to.update=true) then ($rc.update={"QC update sent"})
update in $L.file=update in $L.file.save

```

- 4 Fill in the state name **rca.view** in **Menu navigation** → **Utilities** → **Tools** → **Document Engine** → **States**, click **Search**. Add one record in **Non-base methods** table, then click **Save**.

Display Action	Process Name	Condition	Save First
qcupdate	rca.qcupdate	\$L.mode~="close" and \$L.mode~#"add"	false

- 5 Search for the name **rootcause** in **Menu navigation** → **Toolkit** → **WSDL Configuration**, then update the External Access Definition as follows based on the table **rootcause**.

No.	Field	Value
1	Service Name	QCIntProblemService
2	Object Name	QCIntProblem
3	Allowed Actions	add/Create (Action Names)
4	Allowed Actions	qcupdate/Update (Action Names)

-  Delete all the Allowed Actions without Action Name.

This is shown in the following diagram.

External Access Definition

Service Name:

Name:

Object Name:

Allowed Actions
 Expressions
 Data Policy

Allowed Actions	Action Names
add	Create
qcupdate	Update

Create Subform

- 1 Create Global List.

Create a global list (**Menu Navigation** → **Tailoring** → **Tailoring Tools** → **Global Lists** on Service Manager 7.0x/7.10; **Menu navigation** → **Utilities** → **Tools** → **Global Lists** on ServiceCenter) with following parameters:

No.	Parameter	Value	Remarks
1	List Name	SMQC Integration PM Project List	
2	Regen Every	1 00:00:00	
3	Build List on Startup?	Yes	Check box
4	List Variable	\$G.qcintegration.problem.project	
5	User Defined List?	Yes	Check box
6	Value List	{"server1/domain1/project1", "server2/domain2/project2"}	Change to the values for your system Note: No spaces between slashes.

Save this global list and click **Rebuild Global List** from menu.

- 2 Create subform **pm.qcint.subform** without the Form Wizard (**Menu Navigation** → **Tailoring** → **Forms Designer** on Service Manager 7.0x/7.10; or **Menu Navigation** → **Toolkit** → **Forms Designer** on ServiceCenter) with the following components.

No.	Component	Properties
1	Label	Caption: "Synchronize with QC:"
2	Combo Box	Input: "qcintegration.type" Value List: "0;1;" Display List: "0 - Not Synchronize;1 - Synchronize with QC Defect" Select Only: "Yes" Read-Only Condition: "[!\$qcint.type.readonly]"
3	Label	Caption: "Defect ID:"
4	Text	Input: "qcintegration.id" Read-Only: "Yes"

No.	Component	Properties
5	Label	Caption: "Server/Domain/Project:"
6	Combo Box	Input: "qcintegration.project" Value List: "\$G.qcintegration.problem.project" Read-Only Condition: "[\$qcint.project.readonly]" Mandatory Condition: "[qcintegration.type]>0"
7	Label	Caption: "Created from:"
8	Text	Input: "qcintegration.project" Read-Only: "Yes"

This is shown in the following diagram.

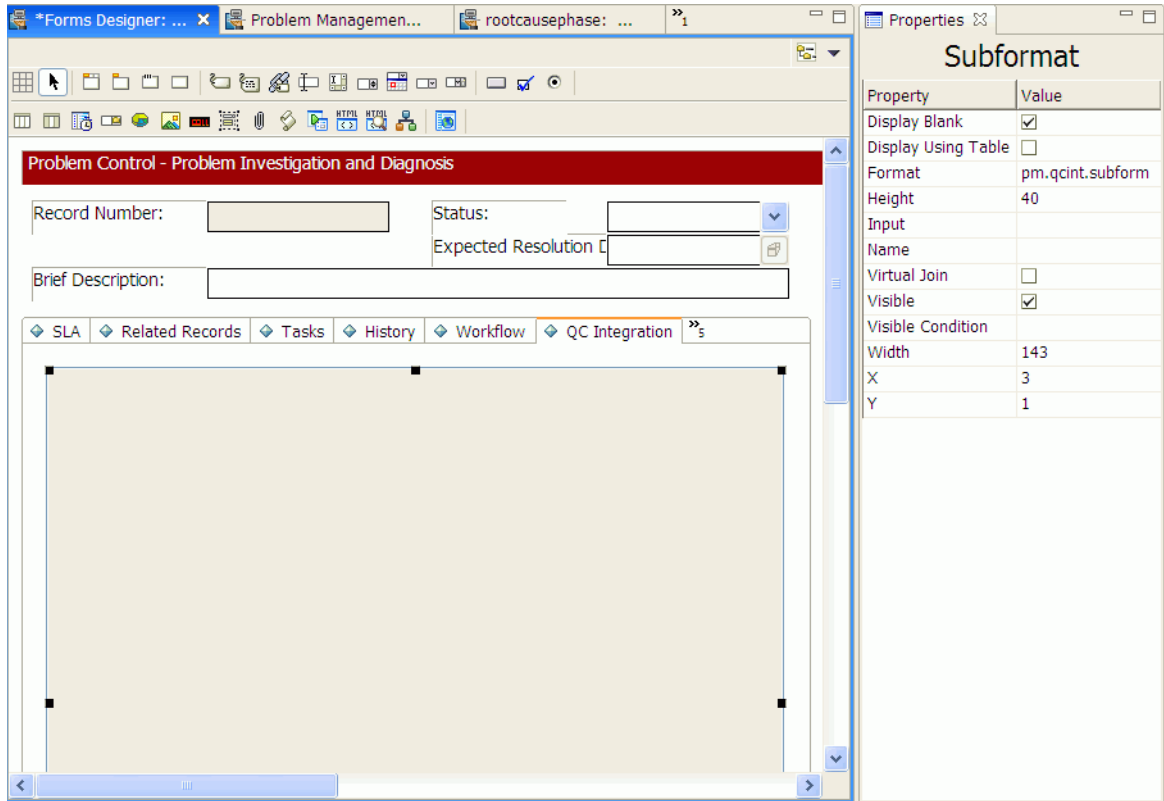
The screenshot shows a software form window with a title bar containing 'OK' and 'Cancel' buttons. Below the title bar is a toolbar with various icons. The form contains four fields:

- 'Synchronize with QC:' followed by a dropdown menu.
- 'Defect ID:' followed by a text input field.
- 'Server/Domain/Project' followed by a dropdown menu.
- 'Created from:' followed by a text input field.

Add Subform to Form

- 1 Open the default form of one phase of Problem Management via Forms Designer (PM.pc.ident.and.class is used as an example in ServiceCenter 6.2/Service Manager 7.0x).
- 2 Add a Notebook Tab with caption "QC Integration".

- 3 Add a Subform to the new tab with format "pm.qcint.subform".



- 4 Save the changes.

➤ If the error message "Format 'pm.qcint.subform' not found (display, show.rio)" appears, then restart the SM server to enable the subform.

Add Format Control Calculations/Validations

- 1 Open the corresponding format control of the form, such as PM.pc.ident.and.class.
- 2 Click **Calculations**.
- 3 Add two records with the following values:

Record	Parameter	Value
1	display	true
	initial	true
	calculation	\$qcint.type.readonly=2;if (qcintegration.type in \$file~=0) then (\$qcint.type.readonly=1)
2	display	true
	initial	true
	calculation	\$qcint.project.readonly=2;if (qcintegration.type in \$file~=0 and not null(qcintegration.project in \$file)) then (\$qcint.project.readonly=1)

- 4 Click **Validations**.
- 5 Add one record with the following values:

No.	Parameter	Value
1	Validation	not null(qcintegration.project in \$file)
2	Message	The Server/Domain/Project is required.
3	Add	qcintegration.type in \$file~=0
4	Update	qcintegration.type in \$file~=0
5	Set Focus to	qcintegration.project

- 6 Save your changes.
- 7 Verify:

◆ Classification ◆ Activities ◆ Attachments ◆ Related Records ◆ History ◆ Workflow ◆ QC Integration

Synchronize with QC: 1 - Synchronize with QC Defect ▼

Defect ID:

Server/Domain/Project localhost/QADEMO/Demo ▼

Created from:

Customizing Quality Center Defects Module

To customize Quality Center Defects module, follow these steps:

- 1 Add Fields
- 2 Add Tabs
- 3 Add Fields to Tabs
- 4 Create a View

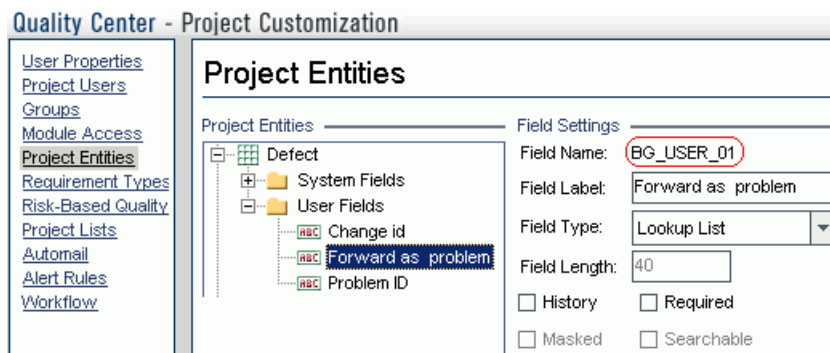
Add Fields

To add the required fields for defect customization, follow these steps.

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Add the following fields for the defect entity in project entities (XX XY XZ are sequential numbers auto-generated by QC).

Field Name	Field Label	Field Type	Remarks
BG_USER_XX	Synchronize with SM Problem	Lookup List/YesNo	Select "Verify Value" checkbox
BG_USER_XY	Problem ID	String	
BG_USER_XZ	Created from	String	

The following diagram shows an example project entity.



▶ The data type requirements for QC fields is described in [Matching Types](#) on page 30.

Add Tabs

To add tabs to the defect form and show fields on these tabs, follow these steps.

- 1 In "QC - Project Customization" click **Workflow**.
- 2 Click **Workflow** → **Script Editor**.

- 3 Choose **Defects module script**.

Quality Center - Project Customization

Workflow

[Script Generator - Add Defect Field Customization](#)
Enables you to customize the fields displayed for each user group in the Add Defects dialog box. You can also specify field order and whether a field is required.

[Script Generator - Defect Details Field Customization](#)
Enables you to customize the fields displayed for each user group in the Defect Details dialog box. You can also specify field order and whether a field is required.

[Script Editor](#)
Enables you to write VBScript code for all Quality Center modules. You can also use the Script Editor to modify the scripts generated by the above tools.

Script Editor

Script Editor Toolbar Button Editor

Workflow Scripts

- Common script
- Requirements module script
- Test Plan module script
- Test Lab module script
- Manual Runner script
- Defects module script

Event Procedures:

- GetNewBugPageName
- GetDetailsPageName
- Bug_New
- Bug_MoveTo
- Bug_FieldCanChange
- Bug_FieldChange
- Bug_CanPost
- Bug_CanDelete
- Bug_AfterPost
- SetFieldApp

- 4 Add the following code to the **GetNewBugPageName** event procedure (which is triggered before QC opens the Add Defect dialog box).

```
select case PageNum
  case "2"
    GetNewBugPageName = "SM Integration (New)"
end select
```

➤ 2 specifies tab 2 (the second tab). For a new bug, the tab name is **SM Integration (New)**.

- 5 Add the following code to the **GetDetailsPageName** event procedure (which is triggered before QC displays Defect Details dialog box).

```
select case PageNum
  case "2"
    GetDetailsPageName = "SM Integration (Details)"
end select
```

➤ 2 specifies tab 2 (the second tab). For an existing defect, the tab name is **SM Integration (Details)**.

Add Fields to Tabs

- 1 If WizardFieldCust_Details and WizardFieldCust_Add are not found in the list, then do the following to generate these two methods.
 - a **Script Generator - Add Defect Field Customization**
 - b **Script Generator - Defect Details Field Customization**



- 2 Add the following code to the WizardFieldCust_Details event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
SetFieldApp "BG_USER_XZ", True, False, 1, 2
```

The parameters are

- Field name (BG_USER_XX, where XX = two digits)
- Visible (True)
- Required (False)
- Page number (start from 0)
- View order (start from 0)

- 3 Add the following code to the WizardFieldCust_Add event procedure.

```
SetFieldApp "BG_USER_XX", True, False, 1, 0
SetFieldApp "BG_USER_XY", True, False, 1, 1
SetFieldApp "BG_USER_XZ", True, False, 1, 2
```

- 4 Set the **ReadOnly** fields by adding the following lines to Bug_New and Bug_Moveto subroutines:

```
if (Bug_Fields("BG_USER_XX").Value="Y") then
    Bug_Fields("BG_USER_XX").IsReadOnly=True
end if
Bug_Fields.Field("BG_USER_XY").IsReadOnly=True
Bug_Fields.Field("BG_USER_XZ").IsReadOnly=True
```

The if loop above marks the field "Synchronize with SM Problem" as read only after selected and saved.

- 5 Exit customization (save changes).

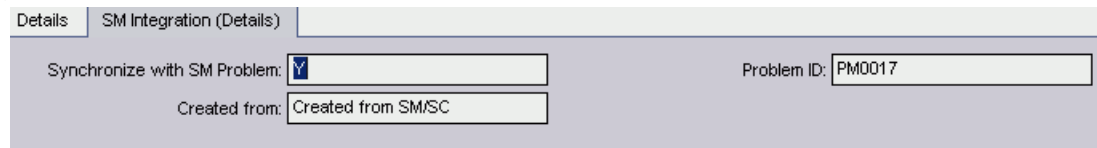
Create a View

- 1 Log on to QC as the integration account - SMQCIntUser.
- 2 In the Defects module, click **View / Filter/Sort / Set Filters/Sort**. The purpose of this view is to let QC Synchronizer correctly filter those defects to be synchronized to SM as problems.
- 3 Set **Synchronize with SM Problem** to **Y**.
- 4 Add a view to Favorites:
 - Name: SMIntegrationView
 - Location: private



In the QC Synchronizer this view will be selected as the QC data filter. QC defects can not be forwarded to SM without the filter.

Verify



Details | SM Integration (Details)

Synchronize with SM Problem: Problem ID:

Created from:

Configuring Links in QC Synchronizer

This section describes how to create and test a link.

- [Specify Endpoints / Type of Link](#)
- [Filters](#)
- [Field Mappings](#)
- [Events](#)
- [Test](#)

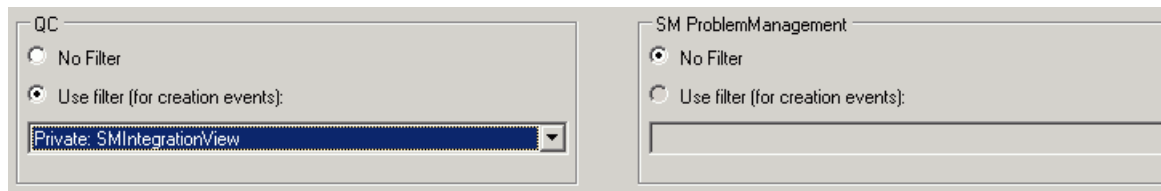
Specify Endpoints / Type of Link

Specify the connection properties as described in [Create a Link](#) on page 26 with the following settings specific for this type of link:

- 1 Step 1 endpoint 2 type = **SM ProblemManagement**.
- 2 Step 3 service URL =
**http://<service_manager_host>:<port>/sc62server/PWS/
QCIntProblemService.wsdl**
- 3 Step 4 select entity types = **Problem by Defect** (this is the only available selection).

Filters

In the Filters tab, select filter **SMIntegrationView** for QC endpoint. If the filter is not available, see [Create a View](#) on page 108.



QC

No Filter

Use filter (for creation events):

Private: SMIntegrationView

SM ProblemManagement

No Filter

Use filter (for creation events):

Field Mappings

Basic field mappings are summarized below:

QC	Directions	SM	Constant Value	Remarks
Problem ID	<-	ProblemID		Synchronize back on create: Yes
Defect ID	->	QCEntityID		Synchronize back on create: Yes
Synchronize with SM Problem			Y	
		QCIntegrationType	1	
Created from			Created from SM/SC	
		CreatedFrom	Created from QC	
	->	CurrentPhase	XXX	Note: Replace XXX with a valid phase name, such as "Problem Investigation and Diagnosis". This field mapping is optional for demo data of Service Manager 7.10.
	->	QCProject	(your setup)	This value should be same with "QC Project" parameter in Connectivity tab.
	->	WorkFlowType	YYY	Note: Replace YYY with a valid category name, such as "ITIL" for demo data of SM 7.0x/SC 6.2; "BPPM" is for demo data of Service Manager 7.10. This field mapping is optional for demo data of Service Manager 7.10.

Example field mappings are shown in the following figure:

Mapped Fields			
Type	QC Field	Direction	SM ProblemManagement Field
	Summary	<---->	Description
	Defect ID	---->	QCEntityID
	Severity	<---->	Severity
	Problem ID	<---->	ProblemID
	Synchronize with SM Problem	<---->	Value: Y
	Value: Created from Quality C...	---->	CreatedFrom
	Value: 1	---->	QCIntegrationType
	Value: AUTO	---->	AssignmentGroup
	Value: BOB_HELPDESK	---->	ProblemOwner
	Value: client system	---->	Category
	Value: software	---->	SubCategory
	Value: email client	---->	ProductType
	Value: outlook	---->	ProblemType
	Value: 4 - User	---->	Impact
	Value: Problem Identification ...	---->	CurrentPhase
	Value: localhost/DEFAULT/...	---->	QCProject
	Value: ITIL	---->	WorkFlowType
	Created from	<---->	Value: Created from SM/SC

Events

The following table lists the events.

Events Tab Settings	QC Action (Event)	SM Action (Event)
Creation	Create a corresponding record in the other endpoint.	Create a corresponding record in the other endpoint.
Update	Update its corresponding record in the other endpoint.	Update its corresponding record in the other endpoint.
Deletion	Do nothing.	Do nothing.

The following diagram shows the settings.

The screenshot shows a configuration window with two main sections: 'QC' and 'SM ProblemManagement'. Each section has three sub-sections: 'Creation', 'Update', and 'Deletion (Full Synchronization Only)'. In the 'QC' section, the 'Creation' sub-section has 'Create a corresponding record in the other endpoint' selected; the 'Update' sub-section has 'Update its corresponding record in the other endpoint' selected; and the 'Deletion' sub-section has 'Do nothing' selected. The same selections are made in the 'SM ProblemManagement' section. Red boxes highlight these selected options in both sections.

Test

To test the link, follow these steps.



The following is just an example. The exact steps required on your system may differ significantly. The phase in which the tab for QC Integration appears may be different on your system.

- 1 Save the configuration (an integrity check is automatically run).
- 2 Click **Enable Link**.
- 3 Create a problem in SM and select "1-Synchronize with QC Defect".

The screenshot shows the 'Problem Control - Problem Identification and Classification' interface. At the top, there is a red header bar. Below it, there are several input fields: 'Record Number' (PM0021), 'Status' (Updated), 'Expected Resolution Date', and 'Brief Description' (defect <-> problem new problem). A navigation bar contains tabs for 'Classification', 'Activities', 'Attachments', 'Related Records', 'History', 'Workflow', and 'QC Integration'. The 'QC Integration' tab is selected. Below the tabs, there are four rows of configuration options: 'Synchronize with QC' (dropdown menu set to '1 - Synchronize with QC Defect'), 'Defect ID' (text input field), 'Server/Domain/Project' (dropdown menu set to 'localhost/DEFAULT/Demo'), and 'Created from' (text input field). The HP logo is visible in the top right corner.

- 4 Create a defect in QC and set "Synchronize with SM Problem" to true.

The screenshot shows a 'New Defect' window. At the top, there's a title bar and a toolbar with icons for 'Clear', 'Attach', and other functions. Below the toolbar, the 'Summary' field contains the text 'defect <-> problem new defect'. The 'Details' tab is selected, showing 'SM Integration (New)'. In this section, the 'Synchronize with SM Problem' checkbox is checked, and the 'Problem ID' field is empty. The 'Created from' field is also empty. At the bottom of the form, there are 'Submit' and 'Close' buttons.

- 5 Synchronize.

The screenshot shows a task execution progress window. The title bar contains buttons for 'Cancel Current Task', 'View Report', 'Refresh Progress', and 'Auto Refresh'. The main area displays the following log:

```
Running: Task execution started.  
Running: Connecting to endpoint 1...  
Running: Querying non-filtered set...  
Running: Handling endpoint 2 - Processing entity #1 of #1 in the Create list. (Total: passed = 0, failed = 0)  
Running: Handling endpoint 1 - Processing entity #1 of #1 in the Create list. (Total: passed = 2, failed = 0)  
Passed: Disconnecting...  
Completed : Passed
```

6 View the problem in SM.

Problem Control - Problem Identification and Classification

Record Number: Status:
Expected Resolution Date:

Brief Description:

Classification Activities Attachments Related Records History Workflow **QC Integration**

Synchronize with QC:
Defect ID:
Server/Domain/Project:
Created from:



Problem Control - Problem Identification and Classification

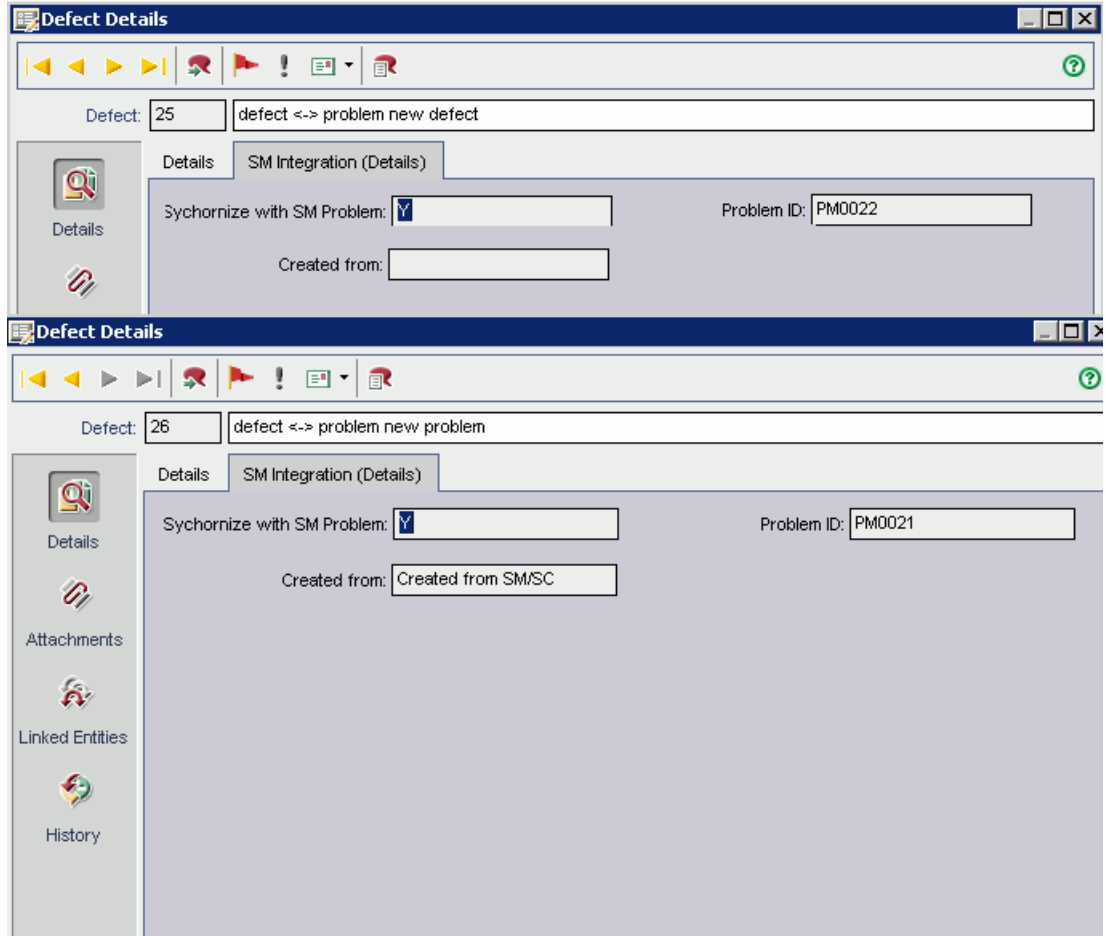
Record Number: Status:
Expected Resolution Date:

Brief Description:

Classification Activities Attachments Related Records History Workflow **QC Integration**

Synchronize with QC:
Defect ID:
Server/Domain/Project:
Created from:

7 View the defect in QC.



11 Upgrade

- ▶ If Service Manager/ServiceCenter, Quality Center, and Quality Center Synchronizer products need to be upgraded, see respective upgrade guide of these products.

Upgrading to the Latest Release

Upgrade consists of the following three tasks:

- Backup jar files and links
- Deploy the latest adapters
- Upgrade customization on SM/SC, QC and QCS environment

Backup Jar Files and Links

- 1 Backup all sm-*.jar files (including stub jar) under `<QCS_Install_Dir>\adapters\lib` directory.
- 2 Backup configuration of all links in QCS client by clicking **Link** → **Export** → **Link Configuration Into XML File....**
- 3 Backup data of all links in QCS client by clicking **Link** → **Export** → **Link Data Into Backup File....**

Deploy the Latest Adapters

- 1 Stop HP Synchronizer Server by clicking **All Programs** → **HP Quality Center Synchronizer** → **Stop Synchronizer**.
- 2 Remove all sm-*.jar files from `<QCS_Install_Dir>\adapters\lib` directory.
- 3 Install SMQC Patch 2 package.
Download installer for Patch 2 from HP Quality Center Add-ins website (<http://updates.merc-int.com/qualitycenter/qc90/sync/sm/index.html>) and run it.
- 4 Deploy all jars in `<release-package>\adapter` folder and stub jar to `<QCS_Install_Dir>\adapters\lib` directory. See [Deploying Adapters](#) on page 18 for more details.
- 5 Start HP Synchronizer server by clicking **All Programs** → **HP Quality Center Synchronizer** → **Start Synchronizer**.

Upgrade User Stories

SM Change -> QC Defect

Upgrade SM/SC Customization

No upgrade needed.

Upgrade QC Customization

Add a new field "Created from" in defect entity. Perform the following steps:

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Add the following fields for the defect entity in project entities (XY is a sequential number auto-generated by QC).

Field Name	Field Label	Field Type	Remarks
BG_USER_XY	Created from	String	

- 4 In "QC - Project Customization" click **Workflow**.
 - a Add the following code to the WizardFieldCust_Details and WizardFieldCust_Add event procedures.

```
SetFieldApp "BG_USER_XY", True, False, 1, 1
```
 - b Set the Readonly fields by adding the following lines to **Bug_New** and **Bug_Moveto** subroutines.

```
Bug_Fields.Field("BG_USER_XY").IsReadOnly=True
```
- 5 Exit customization (save changes) and log out.

Upgrade Quality Center Synchronizer Customization

- 1 In Quality Center Synchronizer client, edit the link and refresh Schemas.
- 2 Add a constant mapping to the link for this user story.

QC	Direction	SM	Constant Value
Created from	<-		Created from SM/SC

- 3 Save the link.

SM Change -> QC Requirement

Upgrade SM/SC Customization

No upgrade needed.

Upgrade QC Customization

Add a new field "Created from" in requirement entity. Perform the following steps:

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Add the following fields for the requirement entity in project entities (XY are sequential numbers auto-generated by QC).

Field Name	Field Label	Field Type	Remarks
RQ_USER_XY	Created from	String	

- 4 In Requirement Types add "Created from" field to the Business type requirement. Business type is the default requirement type for incoming requirements (other types can be used).
- 5 In "QC - Project Customization" click **Workflow**.
To set fields as read only and place the fields on the tabs, in the Script Editor for the Requirements module script, add the following code to Req_New and Req_Moveto (Req_New is called when a new Requirement is created; Req_Moveto is called when an existing Requirement is opened).

```
Req_Fields.Field("RQ_USER_XY").IsReadOnly=True  
SetReqField "RQ_USER_XY", True, False, 1, 1
```
- 6 Exit customization (save changes) and log out.

Upgrade Quality Center Synchronizer Customization

- 1 In Quality Center Synchronizer client, edit the link and refresh Schemas.
- 2 Add a constant mapping to the link for this user story.

QC	Direction	SM	Constant Value
Created from	<-		Created from SM/SC

- 3 Save the link.

QC Defect -> SM Problem

Upgrade SM/SC Customization

Perform the following steps:

- 1 Log on to SM/SC as project administrator.
- 2 Go to **System Definition** → **Tables** → **rootcause** → **Fields**.

- 3 Add two new fields:

Field	Type	
	Service Manager 7.0x/7.10	ServiceCenter 6.2
qcintegration.type	Character	Text
qcintegration.created.from	Character	Text

- 4 Customize forms.
Add a field "Created From" to this sub form "pm.qcint.subform". See section [Add Fields](#) on page 81, [Chapter 9, QC Defect -> SM Problem](#).
- 5 WSDL configuration.
In **WSDL Configuration**, open this WSDL configuration (service name: "QCIntProblemService") , and enable the following two fields:

Field	Caption	Type	Remarks
qcintegration.type	QCIntegrationType	StringType	
qcintegration.created.from	CreatedFrom	StringType	

See section [Specify Field External Access](#) on page 82, [Chapter 9, QC Defect -> SM Problem](#).

- 6 If you use ServiceCenter 6.2, restart the server.

Upgrade QC Customization

Add a new field "Created from" in defect entity. Perform the following steps:

- 1 Log on to QC as project administrator.
- 2 Click **Tools / Customize**. Module "QC - Project Customization" appears.
- 3 Rename "Forward as Problem" to "Synchronize with SM Problem".

Project Entities

Project Entities

- Cycle
- Defect
 - System Fields
 - User Fields
 - Change ID
 - Created from
 - Problem ID
 - Synchronize with SM Problem
- Release
- Release Folder
- Requirement
- Run
- Test
- Test Instance
- Test Set
- Test Step

Field Settings

Field Name: BG_USER_01

Field Label: Synchronize with SM Problem

Field Type: Lookup List

Field Length: 40

History Required

Masked Searchable

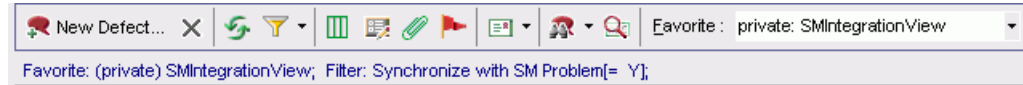
Lookup List

YesNo New List Goto List

Verify Value

Allow Multiple Values

- 4 Log off and log on again with the integration account.
- 5 Update filter "SMIntegrationView" as "Synchronize with SM Problem = Y".



- 6 Log off.

Upgrade Quality Center Synchronizer Customization

- 1 Deploy stub jar again. See [Generating/Deploying Stub](#) on page 18.
- 2 In Quality Center Synchronizer client, edit the link and refresh Schemas.
- 3 Add two constant mappings to the link for this user story.

QC	Direction	SM	Constant Value
	->	QCIntegrationType	1
	->	CreatedFrom	Created from Quality Center

- 4 Save the link.

Post-Upgrade

If you are using the integration solution release 1.00

A new parameter **Socket timeout (Minutes)** is introduced since Patch 1. To leverage this parameter, you only need to export/import links to make this new parameter available on the Quality Center Synchronizer client.

Perform the following steps:

- 1 Export all links as XML files in Quality Center Synchronizer client via **Link → Export → Link Configuration into XML File...**
- 2 Delete each link in HP Quality Center Synchronizer client by clicking **Link → Delete**.
- 3 Restore all links by importing XML files into HP Quality Center Synchronizer client by clicking **Link → Create From → Link configuration XML File...**

A Error Messages

This appendix describes the following categories of error messages:

- Required Fields
- Installation
- Configuration
- Runtime
- XML Validation

Required Fields

The following field names are fixed (can not be changed).

Field Name	Module	Action	Error Message/Symptom
qcintegration.type	SM Database	Synchronization	No errors in the log and the records failed to be synchronized.
qcintegration.project	SM Database	Synchronization	No errors in the log and the records failed to be synchronized.
QCIntChangeService	SM WSDL Configuration	Generate/ Deploy Stub	The stub for Service Manager Change generation failed.
QCIntChange	SM WSDL Configuration	Create a link	System.Web.Services.Protocols.SoapException:java.lang.reflect.InvocationTargetException
ChangeNumber	SM WSDL Configuration	Synchronization	Query: SM ChangeManagement: Can not getRecordIDs. Synchronize: Processing synchronization events failed. Error : java.lang.NoSuchMethodError

Field Name	Module	Action	Error Message/Symptom
Modified	SM WSDL Configuration	Synchronization	Query: SM ChangeManagement: Can not getRecordIDs. Synchronize: Processing synchronization events failed. Error : java.lang.NoSuchMethodError / Query: SM ProblemManagement: Can not getRecordIDs. Synchronize: Processing synchronization events failed. Error : java.lang.NoSuchMethodError
QCIntProblemService	SM WSDL Configuration	Generate/ Deploy Stub	Stub for Service Manager Problem generation failed.
QCIntProblem	SM WSDL Configuration	Create a link	System.Web.Services.Protocols.SoapException:java.lang.reflect.InvocationTargetException.
ProblemID	SM WSDL Configuration	Synchronization	Query: SM ProblemManagement: Can not getRecordIDs. Synchronize: Processing synchronization events failed. Error : java.lang.NoSuchMethodError.

Installation

MSG_ID	Message	Cause	Solution
INS_1	Stub for Service Manager Change generation failed.	The Change WSDL URL is not valid.	Provide the valid Change WSDL URL. (i.e. http://localhost:13080/sc62server/PWS/QCIntChangeService.wsdl).
INS_2	The stub for Service Manager Problem generated failed.	The Problem WSDL URL is not valid.	Please provide the valid Problem WSDL URL. (i.e. http://localhost:13080/sc62server/PWS/QCIntChangeService.wsdl).

Configuration

MSG_ID	Message	Cause	Solution
CFG_1	Can not select "Change Management" and "Problem Management" from the endpoint 2 type.	SM adapter cannot be loaded successfully.	Ensure the stub, adapter and dependency jars are in <QCS_Install_Dir>\adapters\lib.
CFG_2	No create/delete event on change entity is allowed.	For "Change->Defect" and "Change->Requirement" in the Events tab, 'Create a / Delete its corresponding record in the other endpoint' for QC Endpoint Events is selected.	Select Do nothing in the radio-box.
CFG_3	No delete event on the problem entity.	In the Events tab, 'Delete its corresponding record in the other endpoint' is selected.	Select Do nothing for all Deletion (Full Synchronization Only) in Events tab.
CFG_4	Missing connection parameter: UserName.	'User name' is empty.	Enter User name in Connectivity tab.
CFG_5	Missing connection parameter: Service URL.	'Service URL' is empty.	Enter Service URL in Connectivity tab.
CFG_6	Missing connection parameter: QC Project.	'QC Project' is empty.	Enter QC Project name in Connectivity tab.
CFG_7	Missing connection parameter: Service URL.	'Service URL' is empty.	Enter Service URL in Connectivity tab.
CFG_8	Connection parameter: Configuration File Path is not valid.	File path is invalid and specified file does not exist.	Enter valid configuration file path name in Connectivity tab (or leave empty).
CFG_9	To connect to endpoint of type SM ChangeManagement. Error: com.hp.qc.synchronizer.adapters.exceptions.AdapterException: Fail to connect to SM:Connection refused: connect. ERROR #2- Fail to connect to SM:Connection refused: connect.	SM Server is shutdown or not available.	Start the SM Server or make available.

MSG_ID	Message	Cause	Solution
CFG_10	ERROR #1- adapter.CONNECTION_FAILURE : Failed to connect to endpoint of type SM ChangeManagement. Error: com.hp.qc.synchronizer.adapters.exceptions.AdapterException: Fail to connect to SM:The web service of SM is not reachable! ERROR #2- Fail to connect to SM:The web service of SM is not reachable!	Web service is not available (for example, is not configured).	Make the Web service available.
CFG_11	ERROR #1- adapter.CONNECTION_FAILURE : Failed to connect to endpoint of type SM ChangeManagement. Error: com.hp.qc.synchronizer.adapters.exceptions.AdapterException: The URL of SM web service is not valid! ERROR #2- The URL of SM web service is not valid!	URL format is wrong.	Correct the URL. The format is: http://<sm server>:<port>/sc62server/PWS/[QCIntChangeService QCIntProblemService].wsdl
CFG_12	Retry times must be an integer between 0 and 3. (0 means disabled).	The value for the parameter Retries on Locked Record in Advanced tab is out of scope (0~3).	Input an integer (0 ~ 3) for this parameter Retries on Locked Record in Advanced tab.
CFG_13	Retry interval must be an integer between 1 and 10.	The value for the parameter Retry Interval (Seconds) in Advanced tab is out of scope (1~10).	Input an integer (1 ~ 10) for this parameter Retry Interval (Seconds) in Advanced tab.
CFG_14	Socket timeout must be an integer between 0 and 120. (0 means default timeout).	The value for the parameter: Socket Timeout (Minutes) in Advanced tab is out of scope (0~120).	Input an integer (0 ~ 120) for this parameter Socket Timeout (Minutes) in Advanced tab.

Runtime

MSG_ID	Message	Cause	Solution
RUN_1	Required field <Field Name> can not be empty or SPACE filled.	Synchronized null /space value to a required field from sponsor to receiver.	Ensure that required field values are not null or filled with spaces.
RUN_2	Error when reading web service response from SM: Not authorized	There are two possible causes: <ol style="list-style-type: none"> 1 Insufficient rights for SM Integration user when creating/updating defect/ requirement in QC or running synchronization to create/update the corresponding change/ problem. 2 Maximum active logins for integration account is exceeded. 	<ol style="list-style-type: none"> 1 Check the rights of integration account. 2 Check and make sure that Unlimited session in Security tab is selected.
RUN_3	Error 23scxmlapi(23) - XML DOM exception caught - code 5 msg An invalid or illegal XML character is specified	Synchronize with an illegal WSDL caption.	Correct WSDL configuration.
RUN_4	Update failed 1/2/3, retry in 10 seconds, error message=...	Record is locked in SM.	Close the locked record.
RUN_5	Update failed for 3 times, skip, error message=...	Record is locked in SM.	Synchronize record manually or run full synchronization to run all missing updates.
RUN_6	Cannot get field for <Field Name>.	SM adapter cannot get a field.	Ensure stub jar has been generated correctly.
RUN_7	value cannot be reached for <Field Name>.	SM adapter cannot find this field from the stub class.	Ensure stub jar has been generated correctly.
RUN_8	Mapping error, no such property <property name> defined in type <type name>.	SM adapter cannot find this property.	Ensure stub jar has been generated correctly.
RUN_9	Error during setting value for key <key name> with value <value>.	Dynamic model cannot find this key.	Ensure stub jar has been generated correctly.

MSG_ID	Message	Cause	Solution
RUN_10	<Module builder class name> cannot be created because of <message>.	SM adapter can't load a specified class.	Ensure stub jar has been generated correctly.
RUN_11	Can not convert to <target class name> from value <value>.	Value cannot be converted to target type.	Ensure WSDL does not expose unsupported data types.
RUN_12	Exception when getting SM response, return code: <return code>.	Problem with SM communication.	Refer to references for error messages.
RUN_13	Error when reading web service response from SM, record is locked [changeID=<recordID>], message=<Message>.	Record is locked.	Close the locked record.
RUN_14	The data in the '<field name>' field of record <record id> - of file <file name> contains data that does not conform to the SOA data type in datadict.	SOAP field data type in WSDL is not correct. If field is Number type, the value in the database is out of the scope of the specified SOAP type. For example, when choosing IntType (data range: (-2,147,483,648 to 2,147,483,647) for a Number field, if this field has a value of 2,147,483,648 (2 ³¹), it will cause this error when reading the record via web service interface.	If this field is Number type, choose DecimalType in WSDL. Otherwise select correct SOAP type.
RUN_15	Unable to create envelope from given source: ...	The name of structure type field in Service Manager may have non-English characters.	Use English characters as name of structure type field.
RUN_16	QC: findRequirementById: Failed getting requirement with id: <id> Failed to update, record was not found or deleted on target null	Deleting or removing a record may result in this problem, because incremental synchronization will fail to find the record.	Restore this record or just run "Full Synchronization" to remove this mapping relationship established for the record before.
RUN_17	Invalid byte 2 of 3-byte UTF-8 sequence.	There are some special non-English characters in the values of fields.	Update the WSDL definition for this field in Service Manager by leaving "Type" field blank and not entering "StringType" for this field.

XML Validation

MSG_ID	Error Message	Cause	Solution
XML_1	Failed to validate the configuration file: cvc-elt.1: Cannot find the declaration of element 'test'.	Root element is not mapping.	Add root element mapping.
XML_2	Failed to validate the configuration file: cvc-complex-type.2.4.b: The content of element 'mapping' is not complete. One of '{module}' is expected. cvc-complex-type.2.4.b: The content of element 'mapping' is not complete. One of '{module}' is expected.	No module element in the mapping element.	Add module element in the mapping root element.
XML_3	Failed to validate the configuration file: cvc-complex-type.2.4.d: Invalid content was found starting with element '{module}'. No child element is expected at this point.	More than two module elements in the mapping file.	mapping element has only have one or two module elements.
XML_4	Failed to validate the configuration file: cvc-enumeration-valid: Value 'others' is not facet-valid with respect to enumeration '[change, problem]'. It must be a value from the enumeration. cvc-attribute.3: The value 'others' of attribute 'name' on element 'module' is not valid with respect to its type, 'ModuleName'.	Name of module is not problem or change.	name attribute of module element should be change or problem.
XML_5	Failed to validate the configuration file: cvc-complex-type.2.4.b: The content of element 'module' is not complete. One of '{field}' is expected.	No field element in the module element.	Define field elements in each module element.
XML_6	Failed to validate the configuration file: cvc-enumeration-valid: Value 'Unknown' is not facet-valid with respect to enumeration '[String, Number, Date, Attachment, Single_Value_List, Multi_Value_List]'. It must be a value from the enumeration. cvc-attribute.3: The value 'Unknown' of attribute 'type' on element 'field' is not valid with respect to its type, 'FieldType'.	field element with the wrong type attribute.	Type attribute of field element must be enumeration '[String, Number, Date, Attachment, Single_Value_List, Multi_Value_List]'

MSG_ID	Error Message	Cause	Solution
XML_7	Failed to validate the configuration file: cvc-datatype-valid.1.2.1: 'wrong' is not a valid value for 'boolean'. cvc-attribute.3: The value 'wrong' of attribute 'readonly' on element 'field' is not valid with respect to its type, 'boolean'.	field element with the wrong readonly attribute.	readonly attribute of field elements should be true or false.
XML_8	Failed to validate the configuration file: cvc-enumeration-valid: Value 'wrong' is not facet-valid with respect to enumeration '[mandatory, optional, recommended]'. It must be a value from the enumeration. cvc-attribute.3: The value 'wrong' of attribute 'required' on element 'field' is not valid with respect to its type, 'FieldRequired'.	field element with the wrong required attribute.	required attribute of field elements should be mandatory, optional or recommended.
XML_9	Failed to validate the configuration file: cvc-complex-type.4: Attribute 'type' must appear on element 'field'.	field element without the type attribute.	type attribute must be defined in field element.
XML_10	Failed to validate the configuration file: cvc-complex-type.4: Attribute 'name' must appear on element 'field'.	field element without the name attribute.	name attribute must be defined in field element.
XML_11	Failed to validate the configuration file: cvc-complex-type.2.4.d: Invalid content was found starting with element 'items'. No child element is expected at this point.	field element with more than one child element items.	Only one items element can be defined in each field element.
XML_12	Failed to validate the configuration file: cvc-complex-type.2.4.b: The content of element 'items' is not complete. One of '{item}' is expected.	items element without the child element item.	Add item element in items element.
XML_13	Failed to validate the configuration file: cvc-complex-type.4: Attribute 'value' must appear on element 'item'.	item element without the attribute value.	Define value attribute for each item element.
XML_14	Failed to validate the configuration file: cvc-minLength-valid: Value " with length = '0' is not facet-valid with respect to minLength '1' for type 'Item'. cvc-complex-type.2.2: Element 'item' must have no element [children], and the value must be valid.	item element without text value.	Define text value for each item element.

MSG_ID	Error Message	Cause	Solution
XML_15	cvc-minLength-valid: Value " with length = '0' is not facet-valid with respect to minLength '1' for type 'NonEmptyString'. cvc-attribute.3: The value " of attribute 'value' on element 'item' is not valid with respect to its type, 'NonEmptyString'.	value attribute with empty value.	Define value of value attribute in item element.
XML_16	cvc-minLength-valid: Value " with length = '0' is not facet-valid with respect to minLength '1' for type 'NonEmptyString'. cvc-attribute.3: The value " of attribute 'name' on element 'field' is not valid with respect to its type, 'NonEmptyString'.	name attribute with empty value.	Define value of name attribute for item element.
XML_17	cvc-enumeration-valid: Value 'Attachment' is not facet-valid with respect to enumeration '[String, Number, Date, Single_Value_List, Multi_Value_List]'. It must be a value from the enumeration. cvc-attribute.3: The value 'Attachment' of attribute 'type' on element 'field' is not valid with respect to its type, 'FieldType'.	field element with type Attachment.	Remove Attachment type element.
XML_18	Fail to validate the configuration file: cvc-datatype-valid.1.2.1: 'xxx' is not a valid value for 'integer'. cvc-attribute.3: The value 'xxx' of attribute 'length' on element 'field' is not valid with respect to its type, 'positiveInteger'.	field element with incorrect length attribute value.	Correct value of field element.

