

## Master Guide

Remote Support Platform 3.2

Document Version: 1.2 – 2015-12-10

CUSTOMER

# Working with the Remote Support Platform Studio for SAP Business One

All Countries

# Typographic Conventions

Type Style	Description
<i>Example</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Textual cross-references to other documents.
<b>Example</b>	Emphasized words or expressions.
EXAMPLE	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
Example	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
<b>Example</b>	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE	Keys on the keyboard, for example, F2 or ENTER.

---

# Document History

Version	Date	Change
1.0	2015-10-22	First version
1.1	2015-11-25	Added B1 patch download tasks Added automatic creation of customers
1.2	2015-12-10	Document classification is changed to CUSTOMER

---

# Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>6</b>
1.1	Application Overview .....	7
1.2	Glossary .....	7
<b>2</b>	<b>Configuring the Remote Support Platform Studio.....</b>	<b>9</b>
<b>3</b>	<b>Developing Simple Tasks .....</b>	<b>11</b>
3.1	Creating Simple Tasks .....	11
<b>4</b>	<b>Developing Advanced Tasks .....</b>	<b>13</b>
4.1	Creating Advanced Tasks .....	13
4.2	Maintaining General Information .....	14
4.3	Defining Validity .....	15
4.4	Scheduling Automatic Runs .....	16
4.5	Defining Task Parameters .....	16
4.5.1	Pre-Defined Parameters .....	18
4.6	Configuring Task Steps .....	19
4.7	Designing Task Workflows .....	23
4.8	Defining Style Sheets .....	26
4.9	Testing Tasks .....	27
<b>5</b>	<b>Publishing Advanced Tasks and Customized Solutions .....</b>	<b>29</b>
5.1	Publishing Tasks .....	29
5.2	Publishing Customized Solutions .....	29
5.3	Delivering Solution Files .....	30
<b>6</b>	<b>Managing Customers and Assigning Tasks .....</b>	<b>31</b>
6.1	Adding and Maintaining Customers .....	31
6.2	Managing Task Assignments .....	31
<b>7</b>	<b>Managing SAP Tasks Remotely .....</b>	<b>34</b>
<b>8</b>	<b>Working with Reports .....</b>	<b>36</b>
<b>9</b>	<b>Task Workflow Language .....</b>	<b>38</b>
9.1	Workflow Statements .....	39
9.1.1	CALL Statements .....	40
9.1.2	LOOP Statements .....	41
9.1.3	BREAK/CONTINUE Statements .....	41
9.1.4	CHOOSE Statements .....	42
9.1.5	ASSIGN Statement .....	42
9.1.6	HALT Statements .....	43
9.1.7	RETURN Statements .....	44
9.2	Sub-Workflows .....	45

---

9.2.1	Defining Main Workflows .....	45
9.2.2	Defining Sub-Workflows .....	45
9.3	On Error Structure .....	46
9.3.1	Error on CALL Statements .....	46
9.3.2	Errors in Main or Sub-Workflows .....	47
9.4	Core Function Library .....	48
9.4.1	Boolean Functions .....	48
9.5	Conformance .....	48
9.6	References .....	49
9.6.1	EBNF .....	49
9.6.2	XML Schema for Task Workflow Language .....	49

# 1 Introduction

The remote support platform studio for SAP Business One enables partners to develop support tasks, deliver tasks to customers, and manage tasks for different customer installations. After configuring the partner channel in their remote support platform for SAP Business One (RSP) installations, customers can retrieve tasks and then send the results back to partners.

This guide provides instructions for using the remote support platform studio for SAP Business One to develop, deliver, and manage tasks.

## Note

For information about installing the remote support platform studio for SAP Business One, see *Administrator's Guide to the Remote Support Platform for SAP Business One*.

This guide consists of the following main sections:

- **Configuring the Remote Support Platform Studio**

This section provides instructions for configuring the basic settings of the remote support platform studio.

- **Developing Simple Tasks**

This section assists you with using templates to quickly create tasks based on common scenarios.

- **Developing Advanced Tasks**

This section assists you with using the graphical programming environment of the remote support platform studio to configure more complex, advanced tasks.

- **Publishing Advanced Tasks and Customized Solutions**

This section provides instructions for publishing advanced tasks and customized solutions, to enable your customers to download and use them.

- **Managing Customers and Assigning Tasks**

This section provides instructions for adding and maintaining customers in the remote support platform studio, and then assigning tasks to specific customers.

- **Working with Reports**

This section assists you with using the RSP Reporting System to view collated information about customers, including systems overviews, system usage, and assigned tasks and their results.

- **Task Workflow Language**

This section provides an overview of the language used to design the process flow of steps in remote support platform tasks.

## See Also

For more information about how to work with the remote support platform for SAP Business One, see the online help documentation for the remote support platform for SAP Business One. You can access the online help documentation by pressing F1 in the application window after you finished installing remote support platform for SAP Business One.

See also SAP Note [2215378](#): Central note for the remote support platform 3.2 for SAP Business One.

SAP Business One Partners: For more documents and sessions on the remote support platform for SAP Business One, see the landing page for the remote support platform in SAPPartnerEdge.com at <https://partneredge.sap.com/en/products/business-one/support/rsp.html>.

SAP Business One Customers: For the most up-to-date information about the remote support platform, see the landing page for the remote support platform in Customer Portal at <http://service.sap.com/smb/sbocustomer/rsp>.

## 1.1 Application Overview

The main window of the remote support platform studio consists of the following tabs:

- **Customer Management** – Use this tab to add and manage your customers, and view the tasks that you have assigned.
- **Simple Tasks** – Use this tab to create and manage simple tasks, which include executing simple SQL (MSSQL and/or SAP HANA) or Microsoft Windows PowerShell commands on customers' machines. You can use templates to quickly create simple tasks based on common scenarios, and can configure them to execute immediately.
- **Advanced Tasks** – Use this tab to develop, manage, and publish advanced tasks, which include the following types:
  - o Content Upload – Request that customers upload content, including databases, files, logs, and so on.
  - o Health checks – Evaluate the status of an SAP Business One landscape based on hardware, software, and database information.
  - o PowerShell – Perform remote administrative tasks on customers' Microsoft Windows operating systems using Microsoft Windows PowerShell scripting.
- **SAP Tasks** - Use this tab to manage customers' RSP SAP tasks, such as approve, reject or configure their schedules.

## 1.2 Glossary

Term	Description
Agent Service	Server side of remote support platform for SAP Business One. After configuration, the agent service communicates with the SAP backend and partner environment to exchange data, task, and status information.
Agent Console	Client side of the remote support platform for SAP Business One where users can configure and work with the remote support platform. The agent console is either installed on the same computer with the agent service, or remotely connected to the agent service if installed on different computers within the same network.
Task	Automated support or maintenance solution that you can execute either automatically according to a configured schedule, or manually upon user request. You can configure the

Term	Description
	support platform to receive tasks from SAP and SAP partners.
Customized Solution	A tailored solution developed for the incident you raised. When you encounter problems with SAP Business One, you can ask an SAP partner to create a message for the incident. After SAP delivers the solution, your SAP partner helps to test and import the customized solution to your installation so you can detect and solve the issue.
SAP Backend	Management software installed at SAP that is used to manage the remote support platform for SAP Business One support tasks, evaluate task output, and provide early notification to SAP consultants. The backend server receives information from remote support platform for SAP Business One and provides new and updated support tasks for download.
PowerShell	Microsoft Windows PowerShell scripting enables you to perform remote administrative tasks on customers' Microsoft Windows operating systems, by providing full access to Component Object Model (COM) and Windows Management Instrumentation (WMI).



## 2 Configuring the Remote Support Platform Studio

Before you can use the remote support platform studio to create and assign tasks, you must configure some basic settings.

### Procedure

To configure the remote support platform studio, do the following:

1. To open the [Configuration](#) window, from the menu bar, choose [Configuration](#) → [General](#).  
The [Configuration](#) window appears.
2. On the [General](#) tab, specify the following:
  - o [Language](#) – From the dropdown list, select the language of the remote support platform studio user interface.
  - o [Recipients' E-Mail Addresses](#) – Enter the e-mail addresses that will receive system notifications when tasks are delivered from the remote support platform studio to the remote support platform for SAP Business One. E-mail notifications are sent when new tasks are received, task results are generated, and content upload requests are received. You can enter multiple e-mail addresses, separated by semi-colons.
  - o [Enable HTTP Proxy](#) – Select this checkbox to use an HTTP proxy server when connecting to the Internet, and then specify the following:
    - o [Address](#) – Enter the address of the proxy server.
    - o [Port](#) – Enter the port number for connections to the proxy server.
    - o [User Name](#) – Enter the user name of an account used for proxy server authentication.
    - o [Password](#) – Enter the password for the account used for proxy server authentication.
3. On the [Databases](#) tab, specify the following:
  - o [Server](#) – Enter the address of the RSPSrv database, which is created during the installation of the remote support platform studio. If the database is on the same machine that runs the remote support platform studio, enter **localhost** in this field.
  - o [User Name](#) – Enter the user name of a database administrator account.
  - o [Password](#) – Enter the password of the database administrator account.
  - o [Use Trusted Connection](#) – Select this checkbox to use Microsoft Windows Authentication when connecting to the database server. If you select this option, you do not have to enter a user name and password for the database.

To test your connection to the database, choose the [Test Connection](#) button. If the connection fails, check your database settings and your Internet connection.
4. On the [WebDAV](#) tab, specify the following:
  - o [WebDAV Server URL](#) – Enter the address of your WebDAV server.
  - o [User Name](#) – Enter the user name of an account used for server authentication.

- o [Password](#) – Enter the password for the account used for server authentication.

To test your connection to the WebDAV server, choose the [Test Connection](#) button. If the connection fails, check your WebDAV settings and your Internet connection.

5. On the [Reporting](#) tab, specify the following:

- o [Sync Results from WebDAV Server Every <> Minutes](#) – Specify how often the reporting service synchronizes results from the WebDAV server.
- o [Port](#) – Enter the port that the reporting service uses to communicate with the WebDAV server.
- o [Run RSP Reporting when System Startup](#) – Select this checkbox to automatically start the Microsoft Windows service that manages reporting when your system starts.

To start the Microsoft Windows service that manages reporting, choose the [Launch RSP Reporting](#) button.

For more information, see *Working with Report*.

6. To save your settings, choose the [Save](#) button.

## 3 Developing Simple Tasks

The remote support platform studio enables you to use templates to quickly create tasks based on common scenarios. You can configure simple tasks to immediately execute on customers' machines, and then view the results in the remote support platform studio.

Simple tasks include the following types:

- Simple SQL tasks – Run simple SQL (MSSQL and/or SAP HANA) scripts to perform basic database operations.
- Simple PowerShell tasks – Run simple Microsoft Windows PowerShell scripts to perform remote administrative tasks on customers' Microsoft Windows operating systems.

### Note

Immediate execution of simple tasks on customers' machines depends on their remote support platform settings. For more information, see the *Online Help* for the remote support platform for SAP Business One.

### 3.1 Creating Simple Tasks

#### Procedure

To create a new simple task, do the following:

1. On the *Simple Tasks* tab, choose the *Create* button.  
The *Simple Tasks* window appears.
2. In the *General Settings* area, specify the following:
  - o *Name* – Enter a name for the new task.
  - o *Category* – From the dropdown list, select whether you want to execute SQL or PowerShell commands.
  - o *Version* – Assign a version number to the task.

### Note

The application automatically assigns a GUID to the task.

3. In the *Script* area, depending on the value you selected from the *Category* dropdown list, enter the MSSQL and/or SAP HANA script or PowerShell script you want to execute.

### Note

Task validation is determined according to the specified scripts, as follows:

When MSSQL script is specified, the task is valid for MSSQL databases.

When SAP HANA script is specified, the task is valid for SAP HANA databases.

When both MSSQL and SAP HANA scripts are specified, the task is valid both for MSSQL databases and

---

for SAP HANA databases.

For a simple PowerShell task, the task is valid for both MSSQL databases and SAP HANA databases.

4. In the *Selected Customers* area, do either of the following:
  - o To create a generic task that is assigned to all customers, select the *Generic Task* checkbox.
  - o To assign the new task to specific customers, select the corresponding checkboxes from the table.
5. To save the task, choose the *Save* button,

## Result

The remote support platform studio releases the task to the specified customers. Customers' remote support platform installations then download the task and, depending on their settings, immediately execute it.

When the task execution is complete, the remote support platform uses the partner channel to upload the task results to your WebDAV server, from which your remote support platform studio installation then downloads the task results.

You can view task results on the *Simple Tasks* tab of the *Remote Support Platform Studio for SAP Business One* window, or the RSP Reporting System.

## 4 Developing Advanced Tasks

The remote support platform studio provides a graphical programming environment that you can use to configure more complex, advanced tasks.

Advanced tasks include the following types:

- Content Upload – Request that customers upload content, including databases, files, logs, and so on.
- Health checks – Evaluate the status of an SAP Business One landscape based on hardware, software, and database information.
- PowerShell – Perform remote administrative tasks on customers' Microsoft Windows operating systems using Microsoft Windows PowerShell scripting.

### 4.1 Creating Advanced Tasks

#### Procedure

To create a new advanced task, do the following:

1. On the [Advanced Tasks](#) tab, choose the [Create](#) button.  
The [New Task](#) window appears.
2. In the [New Task](#) window, select a template for the new task and enter a name for the task. The following table provides an overview of the available templates.

Template	Description
Content Upload	Generate content upload requests, including upload requests for database, files, logs, and so on.
Health Check [Generic]	Creates a generic health check task to collect necessary information from the users' SAP Business One installation. The template contains a predefined parameter hc_result used to indicate whether the system is healthy or not.
Specific Health Check [Specific]	Create a specific health check task to collect necessary information from the users' SAP Business One installation.
PowerShell	Perform remote administrative tasks on customers' Microsoft Windows operating systems using Microsoft Windows PowerShell scripting.



3. Choose the [OK](#) button to save the data and create the new task.  
The task configuration window appears, in which you can configure the settings outlined in the remaining sections of this chapter.

## 4.2 Maintaining General Information

### Procedure

To maintain the general information belonging to a task, do the following:

1. Create a new task or open an existing task.
2. From the left menu tree, choose the name of the task.
3. Specify the information in the following table.

Field	Description
<i>Name</i>	<p>Displays the task name you defined when creating this task. You can change the name if required. The maximum allowed length is 200 characters.</p> <p> <b>Note</b></p> <p>Ensure the task name does not include any of the following special characters: / \ : " ? * &lt; &gt;  </p>
<i>Category</i>	<p>Displays the template category of the task. Note that you cannot change the category after you have created the task.</p>
<i>Execution Mode</i>	<p>From the dropdown list, select one of the following execution modes:</p> <ul style="list-style-type: none"><li>• <i>ManualOnly</i> – Allows customers to run the task only manually.</li><li>• <i>AutoManual</i> – Enable customers to run the task either automatically or manually. For more information about scheduling automatic runs, see <i>Scheduling Automatic Runs</i>.</li></ul>
<i>GUID</i>	<p>Displays the GUID of the task.</p> <p>The application automatically assigns a GUID when you create a task. You cannot edit the task GUID.</p>
<i>Version</i>	<p>Specify the version of the task.</p> <p>To update tasks after publication, you must first open the task, make the necessary updates, and then specify a later version number for the task. If you do not update the version, you cannot publish the updated task.</p>
<i>Priority</i>	<p>From the dropdown list, select the task priority. By default, the priority is <i>Medium</i>.</p> <p>Priority defines the schedule mechanism in task execution. When several tasks are available for execution, those with higher priorities are executed first.</p>
<i>Exclusive</i>	<p>Select this checkbox to force the remote support platform to execute the task only when other tasks are not running.</p>
<i>Note</i>	<p>Enter the numbers of any relevant SAP notes. You can enter multiple numbers, separated by semi-colons. This field is optional.</p>
<i>Execute Immediately</i>	<p>Select this checkbox to enable immediate execution of the task after it has been downloaded to customers' machines.</p> <p> <b>Note</b></p>

Field	Description
	Immediate execution requires customers to enable this functionality in the remote support platform.
<i>Description</i>	Enter a short description for the task, for example, the purpose of the task purpose. The maximum allowed length is 255 characters. This field is optional.
<i>Document</i>	Enter detailed information about the task and instructions for users. This field is optional.

## 4.3 Defining Validity

Determine the valid working environments for your task by specifying the validity information. In this way, only qualified customer's installations can retrieve and run tasks.

You can restrict your target group by specifying the database types, SAP Business One versions, SAP Business One company versions, and the remote support platform versions that your tasks support.

### Procedure

To define the validity of an advanced task, do the following:

1. From the left menu tree, choose *Validity*.
2. Specify the following:

Field	User Actions and Values
<i>Database Types</i>	Select one or more checkboxes to define the supported database types.
<i>Installation Type</i>	<p>Select one or more checkboxes to define the supported installation types.</p> <p><b>i</b> <b>Note</b></p> <p>This setting is relevant only when the task is retrieved in remote support platform version 3.1 patch 06 and higher.</p> <p>If the task was retrieved in previous RSP versions, after upgrading to patch 06 or higher you must re-download the task by creating a new version of the task or by reinstalling the RSP.</p>
<i>SAP Business One Version</i>	<p>Specify a range of supported versions of SAP Business One.</p> <p>To support all SAP Business One versions, enter an asterisk in both the <i>From</i> and <i>To</i> fields.</p>
<i>SAP Business One Company Version</i>	<p>Specify a range of supported company versions.</p> <p>To support all company versions, enter an asterisk in both the <i>From</i> and <i>To</i> fields.</p>
<i>Remote Support Platform Version</i>	<p>Specify a range of supported versions of the remote support platform.</p> <p>To support all versions of the remote support platform, enter an asterisk in both the <i>From</i> and <i>To</i> fields.</p>

Field	User Actions and Values
<a href="#">SAP HANA Version</a>	Specify a range of supported versions of SAP HANA server. To support all versions of SAP HANA server, enter an asterisk in both the <a href="#">From</a> and <a href="#">To</a> fields.

## 4.4 Scheduling Automatic Runs

### Note

The Schedule tab is disabled when you define the task Execution Mode as ManualOnly. To enable tasks to run in scheduled mode, specify the execution mode as AutoManual. For more information, see *Maintaining General Information*.

### Procedure

To define an automatic run for a task, do the following:

1. From the left menu tree, choose [Schedule](#).
2. To enable the schedule function, select the [Enable Schedule](#) checkbox.
3. Specify the start time and recurrence pattern.

## 4.5 Defining Task Parameters

Use the [Parameters](#) tab to define task parameters used in the configuration of task steps. For more information about configuring task steps, see *Configuring Task Steps*.

### Caution

The studio pre-defines one or more parameters for some task types.

Do not delete any of the parameters or you may not be able to run the task correctly in the remote support platform for SAP Business One.

### Procedure

To define parameters for an advanced task, do the following:

1. From the left menu tree, choose [Parameters](#).
2. To add a new parameter, right-click anywhere in the table and select the type of parameter you want to add from the context menu.

The following table provides an overview of the available parameter types.



Parameter Type	Description
Boolean	True or false
Integer	An integer number
String	A string combination of numbers and characters
Double	A floating-point number with double precision
Date Time	A date and time
File Path	The path to a file
Folder Path	The path to a folder
List	A list

3. The *<Parameter Type> Parameter* window appears.

Depending on the type of parameter you are defining, the window contains several possible fields. The following table provides an overview of all the possible fields that the window may contain. Specify values for the fields displayed in the window.

Field	User Actions
<i>Name</i>	Enter a name for the parameter. For the file path and folder path parameters, enter the full UNC path.
<i>Description</i>	Enter a short description of the parameter.
<i>Visibility</i>	Select either of the following radio buttons: <ul style="list-style-type: none"> <li><i>Visible</i> – The parameter is visible to users in the agent console of the remote support platform.</li> <li><i>Invisible</i> – The parameter is not visible to users in the agent console of the remote support platform.</li> </ul>
<i>Mandatory</i>	If you set the parameter as visible, you must specify whether it as a mandatory or optional field. In the agent console of the remote support platform for SAP Business One, a mandatory field requires configuration before customers can approve or run a task. <ul style="list-style-type: none"> <li>To set a parameter as a mandatory field in the task user interface, select the <i>Yes</i> radio button.</li> <li>To set the parameter as an optional field, select the <i>No</i> radio button.</li> </ul>
<i>Constraints Type</i>	You can define whether to apply constraints to this parameter, and what kinds of constraints are applied. From the dropdown list, select either of the following: <ul style="list-style-type: none"> <li><i>None</i> – Select this option so that the parameter is not constrained, and users can specify any value.</li> <li><i>Set</i> – Select this option to constrain the value of the parameter to a collection of discrete options.</li> <li><i>Range</i> – Select this option to constrain the value of the parameter to a continuous range.</li> </ul>
<i>Constraints</i>	Note that you can only edit this field when you select <i>Set</i> or <i>Range</i> for

Field	User Actions
	<p><i>Constraint Type.</i></p> <ul style="list-style-type: none"> <li>If you select <i>Set</i> for <i>Constraint Type</i>, in the <i>Constraints</i> field, specify a collection of values allowed for this parameter.</li> <li>If you select <i>Range</i> for <i>Constraint Type</i>, specify the range for values allowed for this parameter.</li> </ul>
<i>State</i>	<p>From the dropdown list, select one of the following values:</p> <ul style="list-style-type: none"> <li><i>None</i> – When customers make changes to the parameter value before executing a task, the changed value is permanently stored until customers make another change. When customers make changes to the parameter value during the task execution process, the steps and workflow may change the value of the parameter in memory, but the change is not stored. This means that a change during the execution process does not affect the parameter value after the execution is finished. When developers use the step and workflow definition to change the value, the changed value is used during execution but is not stored.</li> <li><i>Session</i> – When customers make changes to the parameter value during the task execution process, the change is stored for this particular execution. This means that, for the next execution, the original value is used if customers make no further changes. When developers use the step and workflow definition to change the value, the changed value is used and stored for the first execution.</li> <li><i>Task</i> – When customers make changes to the parameter value during the task execution process, the change is stored for this task, which means for the next execution the changed value is used if customers make no further changes. When developers use the step and workflow definition to change the value, the changed value is used and stored for the task.</li> </ul>

4. Choose the *OK* button.

The application displays the parameter along with its attributes in the table.

### Note

To edit an existing parameter, double click the corresponding row in the table.

## 4.5.1 Pre-Defined Parameters

If required, you can set parameter values in tasks; otherwise, default values are assigned by the remote support platform for SAP Business One. At runtime, the remote support platform for SAP Business One uses these pre-defined parameters to control certain logical operations.

The following table provides an overview of the pre-defined parameters.

Parameter Name	Type	Applicable Task	Description
<a href="#">EmailOnlyIfResultChanged</a>	Boolean	<ul style="list-style-type: none"> <li>Health Check Task (Generic)</li> <li>Health Check Task (Special)</li> </ul>	<p>The remote support platform sends the result and notification via e-mail only when the health check result is different from the previous execution.</p> <p>The default value is <code>True</code>.</p>
<a href="#">BackendOnlyIfResultChanged</a>	Boolean	<ul style="list-style-type: none"> <li>Health Check Task (Generic)</li> <li>Health Check Task (Special)</li> </ul>	<p>The remote support platform uploads the task result to the SAP backend only when the health check result is different from the previous execution.</p> <p>The default value is <code>True</code>.</p>
<a href="#">Hc_result</a>	Boolean	Health Check Task (Generic)	<p>This is an internal parameter. The task diagnoses whether the company database is healthy or not.</p> <p>The default value is <code>True</code>.</p>

## 4.6 Configuring Task Steps

### Procedure

To configure the steps of an advanced task, do the following:

- From the left menu tree, choose [Steps](#).  
To add a new step, right-click anywhere in the table and select the type of step you want to add from the context menu.
- Right-click anywhere in the table, select the type of step you want to add from the context menu, and then, In the [Step Details](#) window, configure the required settings, as follows:
  - Content Upload Request Tasks



#### Caution

The studio predefines steps for this task type.

Do not delete any of the steps otherwise you may not be able to run the task correctly in the remote support platform for SAP Business One.


Step Type	Description	Content/Parameters
CurlInformation Step	Defines the target customer's installation information.	Specify the customer's installation and system information. You can add the responsible consultant's

Step Type	Description	Content/Parameters
		information as required. After the customer uploads the files, your consultants receive a notification via e-mail.
CurDBUpload Step	Uploads one or more databases to the SAP backend.	To add the databases to be uploaded, specify the database names.
CurFileUpload Step	Uploads one or more files to the SAP backend.	To add files to be uploaded, specify the file extensions.
CurB1 LogUpload Step	Uploads the SAP Business One log to the SAP backend.	To add SAP Business One logs to be uploaded, specify the relevant PC names.

- o Health Check Tasks (Specific and Generic) and PowerShell Tasks

### Note

There are no default steps for the health check tasks; you can add relevant steps as required. The most commonly used steps for a health check task are SQL steps, batch steps, and quick store procedure steps.

Step Type	Description	Content/Parameters
Batch Step	Runs a Microsoft Windows batch script.	Specify the batch script.
Quick Store Procedure Step	<p>Runs a stored procedure according to the defined MSSQL script and/or SAP HANA script, and removes this script from the database automatically.</p> <p>This is a more efficient way to run MSSQL and/or SAP HANA scripts in tasks.</p> <p>You can use Print 'error message' to print error messages to task results.</p> <p>For result sets, you do not need to use print statements; the "Select" query result is included in the task results automatically.</p> <p> Note</p> <p>When the respective script (MSSQL or SAP HANA) is</p>	Specify the MSSQL script and/or SAP HANA script.

Step Type	Description	Content/Parameters
	<p>left empty, a step is skipped from execution.</p> <p>When this step is executed on an MSSQL DB type server, only the MSSQL script is executed.</p> <p>When this step is executed on an SAP HANA DB type server, only the SAP HANA script is executed.</p>	
SQL Step	<p>Executes an MSSQL script and/or SAP HANA script.</p> <p><b>i</b> Note</p> <p>When the respective script (MSSQL or SAP HANA) is left empty, a step is skipped from execution.</p> <p>When this step is executed on an MSSQL DB type server, only the MSSQL script is executed.</p> <p>When this step is executed on an SAP HANA DB type server, only the SAP HANA script is executed.</p>	Specify the MSSQL script and/or SAP HANA script.
User Confirmation Step	A built-in step which checks the customers' confirmation before fixing reported issues.	N/A.
Xsl Transformation Step	Transforms the style of a step result file (XML file) for further processing.	<p>Add the following parameters:</p> <ul style="list-style-type: none"> <li>The name of the step for which you want to transform the result file.</li> <li>The name of the style sheet to use for performing the transformation.</li> </ul>

3. Specify the following additional information, if required:

Field	User Actions
Description	Enter a short description for the task.
Group	<p><b>i</b> Note</p> <p>This field affects only data displayed in the task results, for example,</p>

Field	User Actions
	<p>a health check report.</p> <p>Enter a group name.</p> <p>In a task report, data belonging to the same group is displayed in a section titled with the group name you specify here.</p>

4. To set parameters for the step, in the [Step Details](#) window, choose the [Parameters](#) button.

The [Step Parameters](#) window appears, which displays all the parameters you have defined for the task in the left area of the list. Do the following:

1. Select the parameters for this step and choose the add button to add them to the right area of the list.
2. From the dropdown list, select one of the following direction types for the parameters:
  - [Input](#) – The parameter is used for input only.
  - [Output](#) – The parameter is used for output only.
  - [InputOutput](#) – The parameter is used for both input and output.
  - [ReturnValue](#) – The parameter is used to return a value.
3. Use the up and down buttons to adjust the parameters' position in the list.
5. To enable the rating function for the task, choose the [Ratings](#) button.

The [Step Ratings](#) window appears, in which you can add, edit, and remove rating rules.

To add a rating rule, choose [Add Rating Rules](#), and on the [Script MSSQL](#) tab or on the [Script HANA](#) tab, enter the script for rating the task result. You can define the following thresholds for status indicators: Excellent, Acceptable, and Poor.

You can use the following parameters to define the rating rules:

- Parameter `this_column`. For example, `if ${this_column}>0`
- Parameter `this_row[index]` (index from 1). For example, `this_row[1]`, or `this_row[column_name]`



### Example

Rating example for MSSQL:

```
DECLARE @FIELD_VALUE NUMERIC(10);
set @FIELD_VALUE = ${this_column};

if (@FIELD_VALUE <= 200) return 3;
if (@FIELD_VALUE >= 1000) return 1;
```

Rating example for SAP HANA:

```
DECLARE FIELD_VALUE INTEGER;

FIELD_VALUE := ${this_column};

RATING := 2;
IF :FIELD_VALUE <= 200
THEN RATING := 1; END IF;
```

```
IF :FIELD_VALUE >= 1000  
THEN RATING := 3; END IF;
```

### Example

When designing a health check task, you can add rating rules to evaluate a customers' databases status. After customers run the task for their particular databases, they will be able to view detailed status data in the resulting report, for example, whether the database performance is excellent, acceptable, or poor.

## 4.7 Designing Task Workflows

The studio predefines a workflow for each type of task. You can edit the workflow as required.

### Procedure

To design the workflow of an advanced task, do the following:

1. From the left menu tree, choose [Workflow](#).
2. To add a sub-workflow, right-click [Workflow](#) in the left menu tree and choose [New Sub-Workflow](#).
3. To add a statement, in the left menu structure, choose the main or sub-workflow you are going to edit. In the central panel, right-click the main or sub-structure, then specify a statement type to add.
4. To edit a node, click on the appropriate node and in the Misc area change the attribute values as required.

For more information about designing workflows, see *Working with Reports*

The remote support platform studio enables you to use the information gathered from System Status Report tasks to generate reports, which provide the following collated information about all your customers:

- Systems overview for each company database
- Usage of customers' systems
- Assigned tasks
- Results of executed tasks

### Prerequisites

- You have configured the required settings on the [Reporting](#) tab of the [Configuration](#) window in the remote support platform studio, and started the Microsoft Windows service that manages reporting.  
For more information, see *Configuring the Remote Support Platform Studio*.
- One or more of your customers have executed the System Status Report task.  
For more information, see the online help of the remote support platform for SAP Business One.
- You have installed either of the following Web browsers on the same machine on which the remote support platform studio is installed:

- o Microsoft Internet Explorer 8 or later
- o Mozilla Firefox 12 or later

## Procedure

1. To access the reporting system, in a supported Web browser on the machine on which you have installed the remote support platform studio, enter the following URL:

**`http://localhost:<port>/Default.aspx`**

### Note

The port number that you enter must be the same as the specified port number on the [Reporting](#) tab of the [Configuration](#) window in the remote support platform studio, which the reporting service uses to communicate with the WebDAV server. The default port number is 48900.

For example, if you specified the port as 48900 in the [Configuration](#) window, then enter the URL

**`http://localhost:48900/Default.aspx`** to access the reporting system.

2. To view the status of all customers' databases, a list of the most common issues, and the most recent system status reports you have received, select the [Overview](#) tab.
  - o To view a detailed system status report, select the corresponding link from either of the [Report Date](#) columns.
  - o To view a list of customers affected by a particular issue, select the corresponding link in the [Affected](#) column.
3. To view charts of various customer demographics, including the number of customers by version or location, number of users, and database size, select the [Installed Base](#) tab.
4. To view a time-stamped list of task results uploaded by customers, select the [Results List](#) tab. You can use the controls to filter the list by customer information, task information, and time.
 

To view a detailed report for a task result, select the corresponding link in the [Update At](#) column.
5. To view a list of corrections that are available for databases monitored by the remote support platform, select the [Fixing List](#) tab. You can use the control to filter the list customer information and time.

Task Workflow Language.

The following table provides an overview of the available nodes..

Node	Description	User Action
Call Step	Runs a step. You can only add this node after defining one or more steps for the task.	Specify the following attributes: <ul style="list-style-type: none"> <li>• <a href="#">ErrorHalt</a> <ul style="list-style-type: none"> <li>o To stop the step execution when encountering errors and exceptions, select True.</li> <li>o To ignore errors and exceptions during the execution process, select False.</li> </ul> </li> <li>• <a href="#">ErrorHandlerType</a> <ul style="list-style-type: none"> <li>o To invoke a step to handle the possible errors, select Call Step.</li> <li>o To invoke a Sub to handle the possible errors,</li> </ul> </li> </ul>



Node	Description	User Action
		<p>select Call Sub.</p> <ul style="list-style-type: none"> <li>• <b>HandlerName</b> – Specify the step or sub workflow's name that you use to handle possible errors and exceptions.</li> <li>• <b>Name</b> – Displays the name of the step concerned.</li> <li>• <b>Persistence</b> – To keep the result value of this step, select True. Otherwise, choose False.</li> </ul>
Call Sub-Workflow	<p>Runs a sub-workflow.</p> <p>You can only add this node after defining one or more sub-workflows for the task.</p>	<p>Specify the following attributes:</p> <ul style="list-style-type: none"> <li>• <b>ErrorHandlerType</b> <ul style="list-style-type: none"> <li>○ To invoke a step to handle the possible errors, select Call Step.</li> <li>○ To invoke a Sub to handle the possible errors, select Call Sub.</li> </ul> </li> <li>• <b>HandlerName</b> – Specify the step or sub workflow's name that you use to handle possible errors and exceptions.</li> <li>• <b>ReturnParameter</b> – Specify a parameter. The return value of this Sub will be assigned to this parameter.</li> </ul>
Choose	<p>Executes a sequence of statements depending on the value of a condition.</p>	<p>Add When and Otherwise node to specify the execution conditions.</p>
When	<p>Defines a condition, which, when satisfied, runs a specified step or sub-workflow.</p> <p>You can add one or more When nodes inside one Choose node.</p> <p>You can only add this node inside a Choose node.</p>	<p>Specify the following attributes:</p> <ul style="list-style-type: none"> <li>• <b>Function</b> – Specify one of the following functions to compare parameters: <ul style="list-style-type: none"> <li>○ less-than</li> <li>○ great-than</li> <li>○ equals</li> <li>○ unequals</li> </ul> </li> <li>• <b>LeftPara</b> – Specify the parameter on the left side of the expression.</li> <li>• <b>RightPara</b> – Specify the parameter on the right side of the expression.</li> </ul>
Otherwise	<p>If none of the When conditions are satisfied, triggers a specified step or sub-workflow.</p> <p>You can only add this node inside a Choose node.</p>	<p>Specify the following attributes:</p> <ul style="list-style-type: none"> <li>• <b>Function</b> – Specify one of the following functions to compare parameters: <ul style="list-style-type: none"> <li>○ less-than</li> <li>○ great-than</li> <li>○ equals</li> <li>○ unequals</li> </ul> </li> <li>• <b>LeftPara</b> – Specify the parameter on the left side of the expression.</li> </ul>

Node	Description	User Action
		<ul style="list-style-type: none"> <li><a href="#">RightPara</a> – Specify the parameter on the right side of the expression.</li> </ul>
Loop	Executes a sequence of statements multiple times as long as the specified conditions are satisfied.	For the <a href="#">Condition</a> attribute, enter a supported conditional expression, for example, less-than, equals, and so on.
Break	Exits a loop. You can only add this node inside a <code>Loop</code> node.	N/A
Continue	Breaks the current loop round and proceeds to the next round. You can only add this node inside a <code>Loop</code> node.	N/A
Assign	Assigns a value or an expression to a parameter.	Specify the following attributes: <ul style="list-style-type: none"> <li><a href="#">Expression</a> – Enter an expression or a value for the parameter.</li> <li><a href="#">Parameter</a> – Specify the target parameter.</li> </ul>
Halt	Terminates the task execution.	Specify the following attributes: <ul style="list-style-type: none"> <li><a href="#">ErrorMessage</a> – Enter the error message you want to display when the program stops.</li> <li><a href="#">WithError</a> – To define the halt as caused by errors, select True. Otherwise, select False.</li> </ul>
Return	Terminates a sub-workflow and returns the specified value.	Define the value you want to return in the <a href="#">ReturnValue</a> attribute. Supported value types include: <ul style="list-style-type: none"> <li>Literal string</li> <li>Boolean</li> <li>Decimal</li> <li>Date/Time</li> </ul>

## 4.8 Defining Style Sheets

You can use the style sheets tab to transfer task results in XML format to a report in HTML format. You first import Extensible Stylesheet Language Transformations (XSLT) files (.xsl files), and then specify the applicants of the HTML reports.

To enhance the usability and design of HTML reports, you can include CSS markup in the imported XSLT files.

## Procedure

To define style sheets for an advanced task, do the following:

1. From the left menu tree, choose *Stylesheets*.
  2. Right-click anywhere in the *Style Sheets* area, and then select *New* from the context menu.  
The *New Style Sheet* window appears.
  3. Specify the Name, Local Path, and URI of the XSLT file that you want to import, and enter an optional description for the file.
  4. Choose *OK* to save data and import the XSLT file into the current task.
  5. To apply the imported style sheet to the task result, do the following:
    1. Right-click anywhere in the *Applicants* area, and then choose *New* from the context menu.  
The *Applicant* window appears.
    2. In the *Applicant* window, specify the following:
      - *Name* – Enter a name for the applicant..
      - *Default* – Select this checkbox to set this as the default style sheet.
      - *Style sheet* – Right-click in this field, and then select the style sheet you want to assign from the context menu.
      - *Description* – Enter a description for the applicant.
- Note**
- When you have defined multiple applicants, the applicants are applied to the task execution results sequentially.
6. Choose the *Save* button.

## 4.9 Testing Tasks

Before you release a task to your customers, we recommend that you test the task in your installation of the remote support platform for SAP Business One.

### Prerequisites

- You have successfully configured the WebDAV connection and RSPSrv database connection. For more information, see *Configuring the Remote Support Platform Studio*.
- You have configured at least one installation of the remote support platform as the testing environment for your tasks.
- In the testing environment, ensure you have successfully enabled the partner channel and connected to the WebDAV server you use in the studio.

## Procedure


To test an advanced task, do the following:

1. On the *Advanced Tasks* tab, select the task that you want to test, and then choose the *Publish* button.  
The studio uploads this task both to your RSPSrv database and WebDAV server.  
The *Task Assignment* window appears.
2. Select the corresponding checkbox for the customer system you use as a testing environment, and then choose the *Assign* button.



### Caution

Ensure you do not assign tasks that are still in the testing phase to customer systems (other than your testing environments), or set them as generic tasks. For more information about tasks and customer management, see *Managing Customers and Assigning Tasks*.

3. In your testing environment, log on to the remote support platform for SAP Business One.  
For more information about working with the remote support platform, see the *Online Help* for the remote support platform for SAP Business One. You can access the online help documentation by pressing **F1** in the application window.
4. To retrieve tasks, from the tool bar, choose the  (Retrieve Tasks) icon.  
Follow the steps in the system task wizard to retrieve new tasks.



### Note

If retrieving tasks from your WebDAV server fails, check your connection and configuration settings for the WebDAV server, and ensure your customers can successfully configure and use the partner channel.

5. After downloading the task you want to test, you can execute the task to see whether the task works in the remote support platform.

For more information about how to work with tasks, see the *Online Help* for the remote support platform for SAP Business One.

---

## 5 Publishing Advanced Tasks and Customized Solutions

After you develop a task, you can publish the task to enable your customers download and use it. The process also applies to sending a customized solution to a specific customer.

When you publish the task, the studio uploads the task to both the RSPSrv database and your WebDAV server. Should your WebDAV server data become corrupt, you can always recover the data by restoring tasks from your RSPSrv database.

To publish tasks and customized solutions, you first configure the connections to your RSPSrv database and your WebDAV server.

### 5.1 Publishing Tasks

After you finish developing a task, you can deliver it to your customer by uploading the task to your WebDAV server. Customers using the remote support platform for SAP Business One are able to retrieve partner-delivered tasks that are assigned to their systems by connecting to the partner's WebDAV servers.

#### Prerequisite

You have configured a WebDAV server and the RSPSrv database. For more information, see *Configuring the Remote Support Platform Studio*.

#### Procedure

To publish an advanced task, do the following:

1. On the [Advanced Tasks](#) tab, select the task that you want to test, and then choose the [Publish](#) button.  
The studio uploads this task both to your RSPSrv database and WebDAV server.  
The [Task Assignment](#) window appears.
2. Select the corresponding checkbox for the customer system you use as a testing environment, and then choose the [Assign](#) button.

### 5.2 Publishing Customized Solutions

A customized solution is a tailored solution package developed by SAP in response to an incident raised by customers. After SAP delivers the solution, you can help to test and import the customized solution package to detect and solve the issue on behalf of your customers.

---

The process flow for customized solutions is as follows:

1. The customer reports an issue to an SAP partner.
2. The partner creates a message in SAP Service Marketplace to report the issue.
3. An SAP consultant receives the message and develops a customized solution package (a zipped file with .rsp extension).
4. The SAP consultant attaches the file in the CRM system and sends a message to the partner.
5. The partner opens the message and downloads the file.
6. The partner tests the solution, and then sends the solution file to the customer.
7. The customer imports the solution file and runs it to solve the issue.

## 5.3 Delivering Solution Files

### Prerequisite

You have configured a WebDAV server and the RSPSrv database. For more information, see *Configuring the Remote Support Platform Studio*.

### Procedure

To use the remote support platform studio to deliver the solution file to your customer, do the following:

1. From the menu bar, choose *File* → *Release Customized Solution*.
2. In the *Open* window, locate the customized solution (.rsp) file that you want to release, and then choose *OK*.  
The studio uploads this solution file to both your RSPSrv database and WebDAV server, and the solution is visible on the *Advanced Tasks* tab.
3. To assign this customized solution to the customer who reported the issue, assign the solution using the same method for assigning advanced tasks. For more information, see *Managing Task Assignments*.

---

## 6 Managing Customers and Assigning Tasks

After publishing a task, you can assign the task to your customers. Only customers with the assignment are able to retrieve the task that you deliver.

### 6.1 Adding and Maintaining Customers

To assign a task to a customer, the customer must first be added.

A customer is automatically added if one of the following exists:

1. A System Status Report is received at the WebDAV server and is processed by the reporting system.
2. Remote Management for SAP tasks is enabled at the customer's RSP and remote management data are received at the WebDAV server.

To add a customer manually and maintain customer data, do the following:

1. On the *Customer Management* tab, choose the *Add* button.
2. In the *Add Customer* window, enter the following:
  - o *Customer Name* – The name of the customer.
  - o *System Number* – The customer's system number, which is a unique 18-digit number that must be specified when the license is installed.
  - o *Installation Number* – The customer's installation number, which is a 10-digit number supplied by SAP as part of the installation package.
3. Choose the *Save* button.

#### Note

To change a customer's name, on the *Customer Management* tab, select the customer, and then choose the *Edit* button. Once set, you cannot change a customer's system number or installation number.

### 6.2 Managing Task Assignments

You can assign published tasks and also SAP tasks to customers, so that the customers can retrieve the corresponding tasks via the partner channel in the remote support platform.

#### Procedure

To assign an advanced task, do the following:

1. On the *Advanced Tasks* tab, select the task that you want to test, and then choose the *Publish* button.  
The studio uploads this task both to your RSPSrv database and WebDAV server.

2. In the [Task Assignment](#) window, click the [In Process](#) link.
3. Select the corresponding checkbox for the customer system you use as a testing environment, and then choose the [Assign](#) button.
4. To set this task as a generic task, select the [Generic Task](#) checkbox.

A generic task is a task that automatically becomes available to all of your existing and future customers.



#### Example

You have three customers: A, B, and C. You set task 01 as a generic task. Therefore the studio automatically assigns the task to all of your existing customers (A, B, and C).

You add a new customer D. Since task 01 is a generic task, the studio automatically assigns this task to the newly added customer D.

5. To set this task only to specific customers, select the corresponding checkboxes for one or more customers, and then choose the [Assign](#) button.



#### Example

You have three customers: A, B, C. You assign task 02 to all of your existing customers. Therefore task 02 becomes available for customers A, B, and C.

You add a new customer D. Since task 02 is not a generic task, to assign this task to the newly added customer, you have to manually apply the assignment in the [Task Assignment](#) window.

6. To cancel the assignment for a customer, in the [Task Assignment](#) window, click the [Published](#) link.



#### Note

For generic tasks, you cannot cancel the assignment for specific customers.

Select the task for which you want to cancel the assignment, and then choose either the [Set to In Process](#) button or the [Remove](#) button.



#### Note

If your customer has retrieved this task before you cancel the assignment, the remote support platform removes the downloaded task the next time the customer retrieves tasks.

To assign an SAP task to a specific customer, do the following:

1. On the [Customer Management](#) tab, choose the customer's row.
2. Choose the [View SAP Tasks](#) button.
3. To approve or reject the task for the specific customer, select the relevant checkbox in the relevant task's row. You can also approve or reject all SAP tasks for this customer simultaneously by selecting the [Select/Deselect All Tasks as Approve/Reject](#) checkboxes.
4. To set up the task's configuration as task scheduling, select the link in the [Configuration](#) column. You can use the default configuration defined for the task itself, or you can configure specific configurations for the specific customer.

The available configurations are enable/disable task result upload and task scheduling.



#### Note

Setting up the task configuration does not change the actual configuration at the RSP customer end, unless the task is approved for the customer,



---

### Note

Task configuration is not available for SAP Business One path download tasks. For more information, see *Managing SAP Tasks Remotely*.

5. You can configure the task also by task level on the [SAP Tasks](#) tab. For more information, see *Managing SAP Tasks Remotely*.
6. To save the configurations, choose the [Publish](#) button.

## 7 Managing SAP Tasks Remotely

The remote support platform studio enables you to remotely manage your customers' RSP SAP tasks. You can approve or reject SAP tasks, and you can set up task configuration as task scheduling. SAP Business One patch download tasks are available for approving/rejecting remotely but cannot be configured via the remote support platform studio. Those tasks are categorized separately as "SAP Business One Update" and can be searched by this category. RSP patch download tasks are not available in the remote support platform studio to be managed remotely but only in the RSP console, at the customer end.

### Prerequisites

You have configured the required settings on the Channels tab of the Configuration window in the remote support platform of the customer, in SAP Channel and also in Partner Channel.

For more information about channel configuration, see the Online Help for the remote support platform for SAP Business One.

### Procedure

To remotely manage SAP tasks, do the following:

1. Select the [SAP Tasks](#) tab.
2. To approve or reject a task, select the relevant task in the grid and choose the [Manage Task](#) button. Alternatively you can double-click the row. To approve or reject the task for each customer, select the relevant checkbox in the relevant row. You can also approve or reject the task for all customers simultaneously by selecting the [Select/Deselect All Tasks as Approve/Reject](#) checkbox. This setting would apply also for new customers that are added later.

#### Note

Tasks assigned by SAP for specific customers cannot be approved/rejected generically.

3. To set up a task's configuration as task scheduling, select the relevant task in the grid and choose the [Configuration](#) button. You can enable/disable task result upload and you can overwrite the task schedule.

#### Note

Setting up the task configuration does not change the actual configuration at the RSP customer end, unless the task is already approved for the customer and the customer uses the default configuration.

You can configure the task also at the customer-specific level in the [Customer Management](#) tab. For more information, see *Managing Customers and Assigning Tasks*.

#### Note

When the task configuration is defined by task level, [Default](#) is displayed in the [Configuration](#) column and when it is defined by customer level, [Specific](#) is displayed in the [Configuration](#) column.

---

## Note

SAP Business One patch download tasks cannot be configured via the remote support platform studio. Once the task is approved, the corresponding operation is done depending on the settings in the customer's RSP console, in the [Configuration](#) menu → [Software Update](#) tab → [SAP Business One Updates](#) field. It can be either do nothing or a combination of running a pre upgrade test, getting notification, and downloading the package.

A customer can always download the patch package manually, even if the patch task was rejected by the partner in the remote support platform studio. Eventually, the installation of the SAP Business One patch is done only manually.

For more information about software update configuration, see the Online Help for the remote support platform for SAP Business One.

4. To save the configurations, choose the [Publish](#) button.

## 8 Working with Reports

The remote support platform studio enables you to use the information gathered from System Status Report tasks to generate reports, which provide the following collated information about all your customers:

- Systems overview for each company database
- Usage of customers' systems
- Assigned tasks
- Results of executed tasks

### Prerequisites

- You have configured the required settings on the [Reporting](#) tab of the [Configuration](#) window in the remote support platform studio, and started the Microsoft Windows service that manages reporting.  
For more information, see *Configuring the Remote Support Platform Studio*.
- One or more of your customers have executed the System Status Report task.  
For more information, see the online help of the remote support platform for SAP Business One.
- You have installed either of the following Web browsers on the same machine on which the remote support platform studio is installed:
  - o Microsoft Internet Explorer 8 or later
  - o Mozilla Firefox 12 or later

### Procedure

5. To access the reporting system, in a supported Web browser on the machine on which you have installed the remote support platform studio, enter the following URL:

**`http://localhost:<port>/Default.aspx`**

#### Note

The port number that you enter must be the same as the specified port number on the [Reporting](#) tab of the [Configuration](#) window in the remote support platform studio, which the reporting service uses to communicate with the WebDAV server. The default port number is 48900.

For example, if you specified the port as 48900 in the [Configuration](#) window, then enter the URL

**`http://localhost:48900/Default.aspx`** to access the reporting system.

6. To view the status of all customers' databases, a list of the most common issues, and the most recent system status reports you have received, select the [Overview](#) tab.
  - o To view a detailed system status report, select the corresponding link from either of the [Report Date](#) columns.
  - o To view a list of customers affected by a particular issue, select the corresponding link in the [Affected](#) column.

- 
7. To view charts of various customer demographics, including the number of customers by version or location, number of users, and database size, select the [Installed Base](#) tab.
  8. To view a time-stamped list of task results uploaded by customers, select the [Results List](#) tab. You can use the controls to filter the list by customer information, task information, and time.  
To view a detailed report for a task result, select the corresponding link in the [Update At](#) column.
  9. To view a list of corrections that are available for databases monitored by the remote support platform, select the [Fixing List](#) tab. You can use the control to filter the list customer information and time.

## 9 Task Workflow Language

Task workflow language is used to design the process flow of steps in remote support platform tasks. It provides a common syntax and semantics for ordering the precedence of steps in a task.

Task workflow language enables you to design the workflow of steps in tasks and define error handling for task workflow.

Task workflow statements are expressed in XML. The XML namespace derives from the task schema definition. Defining a workflow during the development of tasks is optional. When there is no workflow element in a task, the steps in the steps element execute in the order in which they appear. Otherwise, the execution order follows the workflow rules.



### Example

The following example shows the source code of the workflow defined for a task:

```
<?xml version="1.0" encoding="UTF-8"?>
<Tasks xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" schema-
version="2.0">
  <Task>
    <General>
      ...
      <Name>CheckAndNotify</Name>
      ...
    </General>
    <Validity/>
    <Schedule/>
    <Parameters/>
    <Channels/>
    <Steps>
      <Step Name="CheckCount" Type="Sql">...</Step>
      <Step Name="SendSMS" Type="other">...</Step>
      <Step Name="MakeCall" Type="other">...</Step>
      <Step Name="SendEmail" Type="other">...</Step>
    </Steps>
    <Workflow>
      <Main>
        <call-step name="CheckCount"/>
        <choose>
          <when condition="equals(${count}, 10)">
            <call-step name="MakeCall"/>
          </when>
          <when condition="less-than(${count}, 10)">
            <loop occurrence="2">
              <call-step name="SendSMS"/>
            </loop>
          </when>
        </choose>
      </Main>
    </Workflow>
  </Task>
</Tasks>
```

```

        <otherwise>
            <call-step name="SendEMail"/>
        </otherwise>
    </choose>
</Main>
</Workflow>
</Task>
</Tasks>

```

In the previous code sample, the `CheckAndNotify` task defines the following steps:

- `CheckCount` – collect count through SQL statement
- `SendSMS` – send short messages
- `MakeCall` – make phone calls
- `SendEMail` – send emails

The `Main` workflow constructs the process order for steps.

The following list explains the order in which steps are processed:

1. Process the `CheckCount` step because it appears in the first workflow statement.
2. The second workflow statement contains a `CHOOSE` statement, which has the following execution branches:
  - o `MakeCall` step in a `WHEN` clause
  - o `SendSMS` step in a `WHEN` clause
  - o `SendEMail` step in an `Otherwise` clause

The first two `WHEN` clauses execute corresponding steps when the Boolean functions are matched. The `OTHERWISE` clause is executed by default when neither `WHEN` clause is matched. If the second `WHEN` clause matches the condition that the parameter count less than 10, the `SendSMS` step executes twice because it is wrapped by the `LOOP` statement with an occurrence equal to 2.

## 9.1 Workflow Statements

Task workflow language introduces the following types of workflow statements. Workflow statements are the primary elements to construct workflows for tasks.

- `CALL` statement – Represents invocation to Step or Sub workflow.
- `LOOP` statement – Repeats or loops work statements within it over a given occurrence.
- `BREAK/CONTINUE` statement – Exits a loop, or breaks the current loop round and proceeds to the next round.
- `CHOOSE` statement – Represents a multi-way conditional branching of workflow.
- `ASSIGN` statement – Assigns a value or an expression to a parameter.
- `HALT` statement – Terminates the task execution.
- `RETURN` statement – Terminates the workflow and returns the value.

### Definition (EBNF):

```
Workflow_Statement ::= Call_Statement | Loop_Statement | Choose_Statement
```

## 9.1.1 CALL Statements

Use call-step to call a step. Use call-sub to call a sub workflow.

The required `name` attribute specifies which step is executed when the process meets a `call-step` statement; likewise, `name` also specifies the sub-workflow for a `call-sub` statement.

For more information about sub workflows, see *Sub-Workflows*.

For more information about error handling, see *On Error Structure*.

### Definition (EBNF):

```
Call_Statement      ::=    Call_Step | Call_Sub
Call_Step           ::=    '<call-step' S 'name=' StepName S (On_Error)? ( '/'>' | '>'
S '</call-step>' )
Call_Sub            ::=    '<call-sub' S 'name=' SubFlowName S (On_Error)? ( '/'>' | '>'
S '</call-sub>' )
```



The following statements demonstrate the use of the `call-step` and `call-sub` statements:

Statement	Behavior
<code>&lt;call-step name="CheckCount"/&gt;</code>	Calls <code>CheckCount</code> step.
<code>&lt;call-step name="CheckCount" onerror-step="WriteLog"/&gt;</code>	Calls <code>CheckCount</code> step. When encountering system errors, call another step set in the <code>onerror-step</code> attribute. End the task.
<code>&lt;call-step name="CheckCount" onerror-sub="WriteLogAndRollback"/&gt;</code>	Calls <code>CheckCount</code> step. When encountering system errors, call a sub-workflow set in the <code>onerror-sub</code> attribute. End the task.
<code>&lt;call-sub name="CheckHealth"/&gt;</code>	Calls <code>CheckHealth</code> sub-workflow.
<code>&lt;call-sub name="CheckHealth" onerror-step="WriteLog"/&gt;</code>	Calls <code>CheckHealth</code> sub-workflow. When encountering system errors, call another step set in the <code>onerror-step</code> attribute. End the task.
<code>&lt;call-sub name=" CheckHealth" onerror-sub="WriteLogAndRollback"/&gt;</code>	Calls <code>CheckHealth</code> . sub-workflow When encountering system errors, call a sub-workflow set in the <code>onerror-sub</code> attribute. End the task.



## 9.1.2 LOOP Statements

LOOP statements execute a sequence of statements multiple times as long as it satisfies a specified condition. For the `Condition` attribute, enter a supported conditional expression, for example, less-than, greater-than, equals, and so on.

### Definition (EBNF):

```
[5] Loop_Statement ::= ' <loop' S condition=' ' Argument ' >' (Workflow_Statement)+ ' </loop> '
```



#### Example

```
<Loop condition="$HasNext(SessionListForBackup)">
  <Assign Parameter="MySession">$GetNext(SessionListForBackup)</Assign>
  <Assign Parameter="TestValue">$GetValueOf(${MySession}, SomePara)</Assign>
  <Choose>
    <When condition="equals (${TestValue}, 'OK')">
      <Continue/> //See Continue statement in next section
    </When>
  </Choose>
  <call-step name="huge"/>
</Loop>
```



#### Example

```
<Loop>
  <Choose>
    <When condition="equals (${TestValue}, 'OK')">
      <Break/> //See Break statement in next section
    </When>
  </Choose>
</Loop>
```

## 9.1.3 BREAK/CONTINUE Statements

BREAK and CONTINUE statements are subsidiary statements for loop. They only affect the nearest LOOP statement where they embrace.

To break the loop, use `<Break/>`.

To stop the current round of loop and go to the next round, use `<Continue/>`.

## 9.1.4 CHOOSE Statements

CHOOSE statements execute a sequence of statements depending on the value of a condition, which is a fundamental statement used to navigate the process in different directions.

CHOOSE statements consist of one or more WHEN clauses and an optional OTHERWISE clause.

Starting from the first WHEN clause, once the condition is satisfied, the system execute this WHEN clause, and ignores the following WHEN and OTHERWISE clauses.

For more information about Boolean conditions, see *Core Function Library*.

### Definition (EBNF):

```
Choose_Statement ::= '<choose>' S (When_Clause)+ S (Otherwise_Clause)? S '</choose>'
```

```
When_Clause ::= '<when' S 'condition=' Boolean_Condition '>' (Workflow_Statement)+ '</when>'
```

```
Otherwise_Clause ::= '<otherwise>' (Workflow_Statement)+ '</otherwise>'
```



### Example

```
<choose>
  <when condition="equals(${count}, 10)">
    <call-step name="MakeCall"/>
  </when>
  <when condition="less-than(${count}, 10)">
    <loop occurrence="2">
      <call-step name="SendSMS"/>
    </loop>
  </when>
  <otherwise>
    <call-step name="SendEmail"/>
  </otherwise>
</choose>
```

- If the variable count has an integral value of 10, the system executes the first WHEN clause and ignores the following WHEN clause and OTHERWISE clause.
- If the variable count value is less than 10, the system ignores the first WHEN clause, executes the second WHEN clause, and ignores the OTHERWISE clause.
- If the variable count does not match either of the above two conditions, the system ignores the two WHEN clauses, and executes the OTHERWISE clause.

## 9.1.5 ASSIGN Statement

Use the ASSIGN statement to assign a value or an expression to a parameter.

You can define the following attributes:

- **Parameter** – Specify the target parameter name. Supported parameter states include **None** and **Task** parameters.

For more information about the parameter states, see [Defining Parameters for a Task](#).

- **Expression** – A single parameter reference or function.

If the expression returns a list of values, the parameter of the `ASSIGN` statement must have the type of List.

#### Definition (EBNF):

`Assign_Statement ::=`

`<assign 'S' Parameter='Parameter_Name'><expression>'Expression'</expression></assign>`



#### Example

The following code sample represents an operation to retrieve the first 3 days' sessions, and assign the value to the `SessionListForBackup` parameter.

```
<Assign Parameter="SessionListForBackup">$GetFirstDaysSessions(3,
0001234567)</Assign>
```

## 9.1.6 HALT Statements

`HALT` statements provide an immediate exit point within the task workflow.

`HALT` statements must:

- Appear as the last statement within main or sub-workflows
- Appear as the last statement within `WHEN` or `OTHERWISE` clauses in a `CHOOSE` statement.

To stop the task execution when an error occurs, specify `True` for the `WithError` attribute.

#### Definition (EBNF):

`Halt_Statement ::= ' <halt (with-error='true|false')?>Error_Message</halt> '`

`Error_Message ::= tokens | S`



#### Example

```
<choose>
  <when condition="equals(${count}, 10)">
    <call-step name="MakeCall"/>
    <halt with-error="true">intended error here</halt>
  </when>
  <otherwise>
    <call-step name="SendEmail"/>
  </otherwise>
</choose>
```

## 9.1.7 RETURN Statements

RETURN statements can appear anywhere under main or sub workflows to stop the execution process.

When the program encounters a return statement, it terminates the workflow immediately with the returned value. The returned value can be anything, but the recommended types include:

- Literal string
- Boolean
- Decimal
- Date/Time

### Definition (EBNF):

Return\_Statement ::= '<return>Return\_Value</return>'

Return\_Value ::= Token



### Example

```
<Main>
  <call-sub name="CheckStatus" return="IsHealthy"/>
  <choose>
    <when condition="equals(${IsHealthy},true)">
      <call-step name="DoRight"/>
    </when>
    <otherwise>
      <call-step name="FixingSomething"/>
    </otherwise>
  </choose>
</Main>
<Sub name="CheckStatus">
  <call-step name="YepDoit"/>
  <choose>
    <when condition="less-than(${certainParameter},100)">
      <call-step name="IamFine"/>
      <return>true</return>
    </when>
  </choose>
  <return>>false</return>
</Sub>
```

## 9.2 Sub-Workflows

Use a sub-workflow to group a set of statements.

Some workflow statements in the same order as a group may appear in the whole task workflow more than once. To avoid write same code snippets many times in different places in the same task workflow, group them and give a name as an identity.

Another important reason that introducing sub workflow will mention in section [4 On Error Structure](#).

### 9.2.1 Defining Main Workflows

A main workflow is a mandatory workflow in a task. It appears only once in a task and is the only entry point when the task execution is triggered.

It supports all the workflow statement types: `CALL` statements, `LOOP` statements, and `CHOOSE` statements.

#### Definition (EBNF):

```
Main_Workflow      ::=    '<Main' S (On_Error)? '>' (Workflow_Statement)+ '</Main>'
Workflow           ::=    Main_Workflow (Sub_Workflow)*
```

### 9.2.2 Defining Sub-Workflows

A sub-workflow is an optional workflow in a task. You can define one or more sub-workflows in a task and use different names to identify them.

Sub-workflows support all the workflow statement types: `CALL` statements, `LOOP` statements, and `CHOOSE` statements.

#### Definition (EBNF):

```
Sub_Workflow       ::=    '<Sub' S 'name='SubFlowName S (On_Error)? '>'
                        (Workflow_Statement)+ '</Sub>'
```

#### Note

To avoid deadlock and endless call stacks, `CALL` statements or `On_Error` attributes must not refer to the sub-workflow itself. The following example demonstrates a simple sub-workflow, and a sub-workflow referred to by a `CALL` statement, which causes an endless call stack.

#### Example

Simple sub-workflow:

```
<Sub name="Hello">
    <call-step name="Collect"/>
</Sub>
```

Endless call stack:

```
<Sub name="Hello">
    <call-step name="Collect" onerror-sub="Hello"/>    /* endless call */
<call-sub name="Hello"/>                            /* endless call */
</Sub>
```

## 9.3 On Error Structure

`On Error` structure is a mechanism that enables you to handle errors for or make supplements to your task when a step within workflow encounters environmental changes or system exceptions.

To support error handling with step granularity, define the `onerror-step` attribute for a `CALL` statement. By doing this, you are able to:

- Invoke the `onerror-step` attribute to cover possible exceptions or errors encountered during execution.
- Use the `onerror-sub` attribute to define a sub-workflow to cover the exception that occurred.

### Note

Do not use the same name for the `On_Error` attribute as the `CALL` statement.

Task workflow language also provides a similar try-catch mechanism to cover exceptions or errors derived from statements with no defined `On_Error` attribute. To do so, set `On_Error` in the main or sub-workflow.

For example, when a workflow with `onerror-step` setting detects an error on a `CALL` statement at runtime (when there is no `onerror-step` attribute defined for this statement), the step sets `onerror-step` to cover the error. Same as the `onerror-sub` attribute on Sub or Main workflow, but call another sub workflow to handle the error. This is another reason why the sub workflow concept was introduced.

### Definition (EBNF):

```
On_Error      ::=      'onerror-step='StepName | 'onerror-sub='SubFlowName
```

### 9.3.1 Error on CALL Statements

The `On_Error` setting in a `CALL` statement always has precedence over those in sub- or main workflows.

#### Example

The following code samples demonstrate how to define `onerror-step` attributes and `onerror-sub` attributes to handle possible errors in a `CALL` statement:

```
<call-step name="Collect" onerror-step="Clear"/>
```

When an error or exception occurs in the `Collect` step, the `Clear` step in `onerror-step` runs to cover the problem.

```
<call-step name="Collect" onerror-sub="ProcessError"/>
  <Sub name="ProcessError">
    <call-step name="WriteLog"/>
    <call-step name="Clear"/>
```

```
</Sub>
```

When an error or exception occurs in the `Collect` step, the `ProcessError` sub-workflow runs.

```
<call-sub name="Hello" onerror-step="Bye"/>
  <Sub name="Hello">
    <call-step name="Capture"/>
    <call-step name="Say"/>
  </Sub>
```

When an error or exception occurs in the `Hello` sub-workflow, the `Bye` step runs.

```
<call-sub name="Hello" onerror-sub="Again"/>
  <Sub name="Again">
    <call-step name="PoliteWord"/>
  </Sub>
  <Sub name="Hello" onerror-step="Bye">
    <call-step name="Capture"/>
    <call-step name="Say"/>
  </Sub>
```

When an error or exception occurs in the `Hello` sub-workflow, the `Again` sub-workflow runs.

Since the `On_Error` setting in a `CALL` statement always has precedence over those in sub- or main workflows, in this example, the `onerror-step` step `Bye` defined in `Hello` sub-workflow is ignored.

## 9.3.2 Errors in Main or Sub-Workflows

### Example

```
<Main onerror-step="ProcessError">
  <call-step name="Collect"/>
  <call-step name="Send"/>
</Main>

<call-sub name="Hello" onerror-sub="Bye"/>
  <Sub name="Hello">
    <call-step name="Capture"/>
    <call-step name="Say"/>
  </Sub>
  <Sub name="Bye" onerror-sub="Again">
    <call-step name="Leave"/>
  </Sub>
  <Sub name="Again" onerror-step="Nothing">
    <call-step name="PoliteWord"/>
  </Sub>
```

The previous code sample shows an error handling chain as follows:

1. When an error or exception occurs in the `Hello` sub-workflow, the `Bye` sub-workflow runs.
2. If `Bye` has an error, the `Again` sub-workflow runs.

3. If there is an error again, the `Nothing` step runs.

## 9.4 Core Function Library

To define conditions and variables, the task workflow language provides the following functions:

- Boolean
- Number
- String

### 9.4.1 Boolean Functions

Task workflow defines three Boolean functions, and only integer values are supported.

- `equals(operand-one, operand-two)` – Evaluates whether two operands are equal, that is, whether they have the same integer values. If they are equal, the function returns `True`; otherwise, it returns `False`.
- `less-than(operand-one, operand-two)` – Evaluates whether the integer value of operand-one is less than that of operand-two. If it is less, the function returns `True`; otherwise, it returns `False`.
- `great-than(operand-one, operand-two)` – Evaluates whether the integer value of operand-one is greater than that of operand-two. If it is greater, the function returns `True`; otherwise, it returns `False`.

#### Definition (EBNF):

```
Boolean_Condition      ::= ' " ' Equals_Expression | Less_than_Expression |  
Great_than_Expression ' " '  
Equals_Expression      ::= 'equals(' Argument ', ' Argument ') '  
Less_than_Expression   ::= 'less-than(' Argument ', ' Argument ') '  
Great_than_Expression  ::= 'great-than(' Argument ', ' Argument ') '  
Argument               ::= ' " ' Constant_Integer | Referred_Parameter ' " '  
Referred_Parameter     ::= '${' Parameter_Name '}'
```

## 9.5 Conformance

Task workflow is intended primarily as a component that can be used by task specifications. Therefore, it relies on task specification.



## 9.6 References

### 9.6.1 EBNF

The Extended Backus–Naur Form (EBNF) syntax used in this guide is derived from a simple EBNF notation. For more information, see <http://www.w3.org/TR/2004/REC-xml-20040204/#sec-notation>.

This guide refers to the following EBNF syntax:

```
StepName           ::=      NameAttribute
SubFlowName        ::=      NameAttribute
NameAttribute      ::=      ' ' BasicName ' '
Parameter_Name     ::=      BasicName
Constant_Interger  ::=      ( '0' | [1-9] ([0-9])* )
Digit              ::=      [0-9]
BasicName          ::=      [A-Za-z] ([A-Za-z0-9._] | '-')*      /* Encoding name
contains only Latin characters */
S                  ::=      (#x20|#x9|#xD|#xA)+                /* White Space, consists
of one or more space characters, carriage returns, line feeds, or tabs */
```

### 9.6.2 XML Schema for Task Workflow Language

```
<xs:element name="Workflow" minOccurs="0">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Main">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="call-step" type="callWF"/>
            <xs:element name="choose" type="chooseWF"/>
            <xs:element name="loop" type="loopWF"/>
            <xs:element name="call-sub" type="callWF"/>
          </xs:sequence>
          <xs:attributeGroup ref="onerror"/>
        </xs:complexType>
      </xs:element>
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
        <xs:element name="Sub">
          <xs:complexType>
```

```

        <xs:sequence>
          <xs:element name="call-step" type="callWF"/>
          <xs:element name="choose" type="chooseWF"/>
          <xs:element name="loop" type="loopWF"/>
          <xs:element name="call-sub" type="callWF"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:string" use="required"/>
        <xs:attributeGroup ref="onerror"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:sequence>
</xs:complexType>
</xs:element>

<xs:complexType name="callWF">
  <xs:annotation>
    <xs:documentation> call a real step</xs:documentation>
  </xs:annotation>
  <xs:attribute name="name" use="required"/>
  <xs:attributeGroup ref="onerror"/>
</xs:complexType>

<xs:complexType name="loopWF">
  <xs:annotation>
    <xs:documentation>loop one or more steps </xs:documentation>
  </xs:annotation>
  <xs:sequence maxOccurs="unbounded">
    <xs:choice>
      <xs:element name="call-step" type="callWF"/>
      <xs:element name="loop" type="loopWF"/>
      <xs:element name="choose" type="chooseWF"/>
      <xs:element name="call-sub" type="callWF"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="ocurrence" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="chooseWF">
  <xs:sequence>
    <xs:sequence maxOccurs="unbounded">

```

```

<xs:element name="when">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element name="call-step" type="callWF"/>
        <xs:element name="loop" type="loopWF"/>
        <xs:element name="choose" type="chooseWF"/>
        <xs:element name="call-sub" type="callWF"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="condition" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>
</xs:sequence>
<xs:element name="otherwise" minOccurs="0">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element name="call-step" type="callWF"/>
        <xs:element name="loop" type="loopWF"/>
        <xs:element name="choose" type="chooseWF"/>
        <xs:element name="call-sub" type="callWF"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

<xs:attributeGroup name="onerror">
  <xs:attribute name="onerror-step" type="xs:string" use="optional"/>
  <xs:attribute name="onerror-sub" type="xs:string" use="optional"/>
</xs:attributeGroup>

```



[www.sap.com/contactsap](http://www.sap.com/contactsap)

© 2015 SAP SE. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System ads, System i5, System p, System p5, System x, System z, System z10, System z9, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

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

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, xApps, xApp, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.