

# **HP OpenView Service Desk 4.0**

## **Data Exchange Administrator's Guide**

**First Edition**



**i n v e n t**

**Manufacturing Part Number: N/A**

**August 2001**

---

## Legal Notices

*Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.*

**Restricted Rights Legend.** Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company  
3000 Hanover Street  
Palo Alto, CA 94304 U.S.A.

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19 (c)(1,2).

**Copyright Notice.** © Copyright 2000, 2001 Hewlett-Packard Company

The nomenclature of each version of this software (and manuals therefore) has been devised for commercially convenient reasons, and is not intended to denote the degree of originality of any version of the software with respect to any other version. The extent of protection afforded by, and duration of copyright is to be determined entirely independently of this nomenclature.

### Trademark Notices

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Microsoft® is a U.S. registered trademark of Microsoft Corporation.

MS-DOS™ is a U.S. registered trademark of Microsoft Corporation.

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

Oracle Reports™, Oracle7™, and Oracle7 Server™ are trademarks of Oracle Corporation, Redwood City, California.

SQL\*Net® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

**SQL\*Plus® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.**

**UNIX® is a registered trademark of the Open Group.**

**Windows NT® is a U.S. registered trademark of Microsoft Corporation.**

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



**Preface****1. The Data Exchange Concept**

Data Exchange Overview .....	33
Overview of the Export Process .....	36
Configuring the Import Mapping .....	37
Overview of the Importing Process .....	38

**2. Exporting Data**

Deciding What to Export .....	43
Checking the Database Structure .....	43
Configuring the Extractor .....	45
Configuration File Keywords .....	47
Extraction Configuration Wizard .....	51
Defining the ODBC Connection .....	53
Defining the System Section .....	55
Classes .....	56
Select Source Tables .....	57
Select Source Columns .....	59
Enter the Parent Relation .....	62
Join Tables .....	64
Filter Data .....	66
Data Sort Order .....	69
Enter Settings for Loading Classes .....	71
Name and Block a Class .....	71
Defining Relations in the Configuration File .....	73
Search Key; N:1 Relations .....	73
Parent-Child; N:N Relations .....	74
Relations with Relation Type; 1:Rel:1 Relations .....	76
Defining the ODBC link .....	79
The Extraction Process .....	81
Exporting Data .....	83
Using a Task to Export Data From a Storage Device .....	83
Creating a Task to Export Data .....	85
Relating Data Exchange Tasks .....	85
Exporting From the Command Line .....	85
The Data Exchange Files .....	86
The XML Format .....	86

---

# Contents

Viewing Data Exchange Files .....	90
<b>3. Import Mapping</b>	
About Item Mapping .....	95
Attribute Mapping .....	95
Key Binding .....	95
Value Mapping .....	96
Defining Relations .....	96
Search Key; N:1 Relations .....	97
Parent-Child; N:N Relations .....	100
Relations with Relation Type; 1:Rel:1 Relations .....	102
Mapping Service Events .....	107
Mapping Classes and Attributes .....	108
Creating a New Import Mapping .....	108
Modifying an Import Mapping .....	110
<b>4. Importing Data in Batches</b>	
Using a Task to Import Data .....	119
Creating Data Exchange Tasks .....	124
Executing a Task .....	124
Scheduling a Data Exchange Task .....	125
Creating a Data Exchange Task Group .....	126
Reconciling your Data .....	130
Delta Processing .....	131
Removing Redundant Items and Relations .....	131
Memory Usage .....	134
Scalable Importing .....	135
Requirements .....	136
Preparing the Environment .....	137
Exporting the XML Files .....	139
Performing the Scalable Import .....	139
Starting the Servant Machines .....	139
Starting the Master Machine .....	140
Performing Scalable Import as a Task Group .....	141
Reviewing the Log Files .....	141
Troubleshooting Scalable Importing Errors .....	142

Command Line Parameters . . . . .	144
Exporting Parameters . . . . .	144
Importing Parameters . . . . .	144
Importing and Exporting Parameters . . . . .	145
Integration Accounts . . . . .	147
<b>5. Importing Service Events</b>	
The Service Event Import Process . . . . .	150
Service Desk Event Command Line . . . . .	151
Special Characters . . . . .	153
Using the Service Event Configuration File . . . . .	153
Service Event Examples . . . . .	154
Resending Service Events . . . . .	156
Preventing Event Storms . . . . .	158
Queuing Events. . . . .	158
Creating a Queue . . . . .	158
Enqueue. . . . .	159
Dequeue. . . . .	159
Queuctl. . . . .	159
Addentry . . . . .	160
Service Desk Event Communicator and Database Rules . . . . .	161
Resending Failed Service Events. . . . .	161
<b>6. Auditing and Troubleshooting</b>	
The Viewer, Log Files, and the Debug Mode . . . . .	164
Troubleshooting . . . . .	165
Error Messages. . . . .	170
Possible Errors While Reading the .ini File . . . . .	170
Possible Errors While Writing the .ini File. . . . .	171
<b>A. Integration With Network Node Manager</b>	
Integration Possibilities. . . . .	174
Creating Incidents in Service Desk . . . . .	174
Importing NNM Nodes into Service Desk. . . . .	174
Installation. . . . .	176
Service Desk Application Server . . . . .	176
Configuration . . . . .	177
Configuring Network Node Manager. . . . .	177

---

## Contents

Select an NNM Event . . . . .	177
Modify an Event . . . . .	177
Exporting NNM Data to a Data Source . . . . .	179
Configuring Service Desk . . . . .	181
Configuring the ODBC Connection . . . . .	181
Configure the Sd_event File . . . . .	181
Modify the Import Mapping . . . . .	182
NNM Variables . . . . .	182
Special Characters . . . . .	182
Sequential Attribute Variables . . . . .	183
Special Information Variables . . . . .	184
SNMP-Specific Variables . . . . .	185
Example of NNM on UNIX Server . . . . .	187
Example of NNM on UNIX . . . . .	187
Managing Event Storms on UNIX . . . . .	187

### **B. Integration With ManageX**

Integration Possibilities . . . . .	190
Installation . . . . .	191
Service Desk Application Server . . . . .	191
ManageX Application Server . . . . .	192
ServiceDeskSamplePassThru . . . . .	192
ManageX on a Different Server . . . . .	193
Configuration . . . . .	194
Configuring Service Desk . . . . .	194
Configure the .ini File . . . . .	194
Database Rules . . . . .	194
ManageX Server Account . . . . .	195
Configuring ManageX . . . . .	195
Configure the Acceptance Filter . . . . .	195
ManageX Lights Out Policy . . . . .	197
The Event Queue . . . . .	202

### **C. Integrating with LDAP**

Integration Possibilities . . . . .	204
A Technical Description . . . . .	205



Installation . . . . .	206
LDAP Server Account . . . . .	206
Configuration . . . . .	207
SD_Export . . . . .	207
Configuring the LDAP.ini File . . . . .	207
Explaining the LDAP .ini File . . . . .	208
XML File . . . . .	211
Import Mapping . . . . .	215
Importing your LDAP Directory Information . . . . .	215

## **D. Integrating with Radia**

Installation . . . . .	218
Configuration . . . . .	219
Configuring an ODBC Data Source . . . . .	219
Creating the SQL Server ODBC Data Source . . . . .	219
Configuring the Radia.ini File . . . . .	226
Import Mapping . . . . .	229
Exporting and Importing Radia Data . . . . .	230
Opening Radia Browser from Service Desk . . . . .	231

## **E. Examples**

Using an Example Configuration File . . . . .	234
Importing From an MS Excel Spreadsheet . . . . .	237
Importing Data With Relations . . . . .	245
Data Source . . . . .	245
Preparing the Excel Sheet for Data Exchange . . . . .	247
Defining the ODBC Link . . . . .	248
Configuring the Extractor and the Import Mapping . . . . .	252
Configuring the DSN Section . . . . .	257
Configuring the SYSTEM Section . . . . .	257
Configuring the CLASSES Section . . . . .	257
Configuring the SOFTWARE Class . . . . .	258
Import Mapping for the SOFTWARE Class . . . . .	258
Configuring the ORG Class . . . . .	260
Import Mapping for the ORG Class . . . . .	260
Configuring the EMP Class . . . . .	262
Import Mapping for the EMP Class . . . . .	262
Configuring the PHONE Class . . . . .	264

---

## Contents

Import Mapping for the PHONE Class .....	264
Configuring the HARDWARE Class .....	266
Import Mapping for the HARDWARE Class .....	266
Configuring the HWUSERS Class .....	268
Import Mapping for the HWUSERS Class .....	268
Configuring the REL Class .....	270
Import Mapping for the REL Class .....	270
Configuring the WORKGROUP Class .....	272
Import Mapping for the WORKGROUP Class .....	272
Configuring the WORKGROUPEMP Class .....	273
Import Mapping for the WORKGROUPEMP Class .....	274
Importing the Data .....	276
Importing Multiple Telephone Numbers .....	277
Importing Data From an ASCII Text File .....	281

## Glossary



---

# Contents

Figure 1-1. Overview of the Data Exchange Process .....	34
Figure 2-1. Extraction Configuration Wizard. ....	52
Figure 2-2. ODBC Tab .....	54
Figure 2-3. System Tab .....	55
Figure 2-4. Classes Tab .....	57
Figure 2-5. Currently Selected Source Tables .....	58
Figure 2-6. Selecting Source Tables .....	59
Figure 2-7. Currently Selected Table Columns .....	60
Figure 2-8. Select Table Columns .....	61
Figure 2-9. Enter Parent Relation. ....	63
Figure 2-10. Joined Tables. ....	64
Figure 2-11. Add a Joining Condition .....	65
Figure 2-12. Filter Data. ....	67
Figure 2-13. Add a Filter Condition .....	68
Figure 2-14. Current Sort Order .....	69
Figure 2-15. Sorting Data .....	70
Figure 2-16. Enter Settings for Loading Classes .....	71
Figure 2-17. Name and Block a Class. ....	72
Figure 2-18. Data Exchange Task .....	83
Figure 2-19. Data Exchange dialog box - Exporting Data .....	84
Figure 2-20. Example of XML file .....	91
Figure 3-1. Import Mapping for PERSON Class .....	98
Figure 3-2. Import Mapping for CI_ADMIN_RELATION Class. ....	99
Figure 3-3. Field Mapping for ADMIN_ID .....	99
Figure 3-4. Import Mapping for CI_USER_RELATION_PARENT Class. ....	100
Figure 3-5. Import Mapping for CI_USER_RELATION_CHILD .....	101
Figure 3-6. Field Mapping for Parent .....	102
Figure 3-7. Import Mapping for CI_ID_NAME Class .....	103
Figure 3-8. Import Mapping for SW_ID_NAME Class .....	104
Figure 3-9. Import Mapping for SW_REL_RELATION. ....	105
Figure 3-10. Field Mapping for CI_ID. ....	105
Figure 3-11. Field Mapping for SW_ID. ....	106
Figure 3-12. Import Mapping dialog box for sd_event .....	107
Figure 3-13. Import Mapping Window .....	109
Figure 3-14. Import Mapping dialog box .....	111
Figure 3-15. External Class dialog box. ....	112

---

## Figures

Figure 3-16. Field Mapping dialog box . . . . .	113
Figure 3-17. Expanded Field Mapping dialog box . . . . .	114
Figure 4-1. Data Exchange Task Window . . . . .	119
Figure 4-2. Data Exchange dialog box - Importing Data . . . . .	120
Figure 4-3. Import Progress Monitor . . . . .	122
Figure 4-4. Import Progress Monitor with Delta Processing. . . . .	123
Figure 4-5. Data Exchange Task Group . . . . .	126
Figure 4-6. Relate Tasks . . . . .	127
Figure 4-7. Search for Tasks . . . . .	128
Figure 4-8. Example Group Task . . . . .	129
Figure 4-9. Scalable Importing . . . . .	136
Figure 5-1. Inbound Service Events . . . . .	150
Figure A-1. Modify an Event in NNM. . . . .	178
Figure A-2. Service Event Command Line in NNM. . . . .	179
Figure B-1. Set ManageX Acceptance Filters. . . . .	196
Figure B-2. ManageX ActiveX Script Action . . . . .	198
Figure C-1. The LDAP Integration . . . . .	205
Figure D-1. Create New Data Source for SQL Server dialog box . . . . .	220
Figure D-2. SQL Server DSN Configuration wizard . . . . .	221
Figure D-3. SQL Server DSN Configuration wizard . . . . .	222
Figure D-4. SQL Server DSN Configuration wizard . . . . .	223
Figure D-5. SQL Server DSN Configuration wizard . . . . .	224
Figure D-6. ODBC SQL Server Setup dialog box . . . . .	225
Figure D-7. SQL Server ODBC Data Source Test dialog box . . . . .	226
Figure D-8. Radia Default Import Mapping. . . . .	230
Figure D-9. Web Application with Radia . . . . .	231
Figure E-1. Adjust the Excel Spreadsheet . . . . .	237
Figure E-2. ODBC Microsoft Excel Setup dialog box. . . . .	238
Figure E-3. New Import Mapping dialog box . . . . .	240
Figure E-4. New External Class dialog box . . . . .	240
Figure E-5. Mapping Attributes . . . . .	241
Figure E-6. Complete Import Mapping. . . . .	242
Figure E-7. Example Export Data Task . . . . .	243
Figure E-8. Example Import Data Task . . . . .	244
Figure E-9. EMPLOYEE SOURCE. . . . .	246
Figure E-10. EMPLOYEE SOURCE - Continued . . . . .	246

Figure E-11. ORGANIZATION, SOFTWARE & WORKGROUP SOURCES . . . . .	247
Figure E-12. HARDWARE SOURCE . . . . .	247
Figure E-13. Excel File Name Definition . . . . .	248
Figure E-14. Add the Excel ODBC Driver . . . . .	249
Figure E-15. New Data Source . . . . .	250
Figure E-16. Select the New ODBC Connection. . . . .	251
Figure E-17. Excel ODBC_DSN . . . . .	252
Figure E-18. Org_Excel Data Exchange Task . . . . .	254
Figure E-19. Import Mapping - SOFTWARE . . . . .	259
Figure E-20. Mapping External Class - SOFTWARE. . . . .	260
Figure E-21. Import Mapping - ORG . . . . .	261
Figure E-22. Mapping External Class - ORG. . . . .	262
Figure E-23. Import Mapping - EMP . . . . .	263
Figure E-24. Mapping External Class - EMP. . . . .	264
Figure E-25. Import Mapping - PHONE. . . . .	265
Figure E-26. Mapping External Class - PHONE . . . . .	265
Figure E-27. Import Mapping - HARDWARE . . . . .	267
Figure E-28. Mapping External Class - HARDWARE. . . . .	268
Figure E-29. Import Mapping - HWUSERS . . . . .	269
Figure E-30. Mapping External Class - HWUSERS . . . . .	269
Figure E-31. Import Mapping - REL. . . . .	271
Figure E-32. Mapping External Class - REL . . . . .	271
Figure E-33. Import Mapping - WORKGROUP . . . . .	273
Figure E-34. Mapping External Class - WORKGROUP . . . . .	273
Figure E-35. Import Mapping - WORKGROUPEMP. . . . .	275
Figure E-36. Mapping External Class - WORKGROUPEMP . . . . .	276
Figure E-37. Import Mapping for CL_CONTACT . . . . .	277
Figure E-38. Import Mapping CL_TEL_CONTACT1. . . . .	278
Figure E-39. Field Mapping - Person Field . . . . .	279
Figure E-40. Field Mapping - Number Field . . . . .	279
Figure E-41. Field Mapping - Telephone Type . . . . .	280
Figure E-42. ODBC Text Setup . . . . .	282
Figure E-43. ODBC Text file - Import Mapping . . . . .	284

---

# Figures



Table 1. Revision History . . . . .	21
Table 2-1. . . . .	47
Table 4-1. Decision Matrix for Importing Items . . . . .	130
Table 4-2. Decision Matrix for Importing Relations. . . . .	131
Table 4-3. Example Environment for Scalable Importing . . . . .	138
Table 4-4. Complete Importing and Exporting Commands . . . . .	144
Table 4-5. Importing and Exporting Command Syntax Defined . . . . .	145
Table 5-1. Service Event Switches. . . . .	151
Table 5-2. Service Event Configuration File. . . . .	154
Table 6-1. File Directory . . . . .	165
Table 6-2. Problem Solving . . . . .	166
Table 6-3. Tips . . . . .	168
Table A-1. Service Desk Server . . . . .	176
Table A-2. Sequential Attribute Variables . . . . .	183
Table A-3. Special Information Variables . . . . .	184
Table A-4. SNMP Specific Variables . . . . .	185
Table A-5. NNM UNIX Server. . . . .	187
Table B-1. Service Desk Server . . . . .	191
Table B-2. ManageX Event String Substitutions . . . . .	199
Table B-3. ManageX Event COM Objects. . . . .	201
Table C-1. LDAP Configuration File . . . . .	208

---

# Tables

---

## Preface

Service Desk is an integral part of the HP OpenView portfolio of products. The Service Desk application has a generic data exchange interface that facilitates open integration with third party applications for the purpose of sharing data. Data exchange is the process of exporting data from one application database and importing it into another, in a format that can be used by the application. Human resource data or inventory data can be exported from one application, reformatted and imported into Service Desk. Service events such as a single incident, employee record, or configuration item can also be inserted and updated, allowing you to tie all of your system management applications together.

The *HP OpenView Service Desk: Data Exchange Administrator's Guide* provides you with the ability to better understand the capabilities that exist for exporting data from external sources and importing data into Service Desk. This guide is intended for system administrators who will configure the data exchange settings, or for others involved with data exchange. It is assumed that users of this manual have an understanding of databases.

This guide contains both conceptual and detailed how-to information for configuring and using the data exchange features of Service Desk. A brief outline of the information in this guide is below:

- Chapter 1, “The Data Exchange Concept,” on page 31, provides the reader with a conceptual overview of how the entire data exchange process works.
- Chapter 2, “Exporting Data,” on page 41, explains how to configure an extractor, how to export data, and how to review exported XML files with the Viewer.
- Chapter 3, “Import Mapping,” on page 93, provides detailed descriptions of the import process with information on mapping classes and properties to items and attributes in Service Desk.
- Chapter 4, “Importing Data in Batches,” on page 117, explains how to import batches of data from an external data source into Service Desk; for example, from a network management database into the Service Desk database.
- Chapter 5, “Importing Service Events,” on page 149, provides information about the service event command line and explains how

to import service events.

- Chapter 6, “Auditing and Troubleshooting,” on page 163 provides some helpful information about the auditing and troubleshooting tools available with Data Exchange.
- Appendix A , “Integration With Network Node Manager,” on page 173 contains useful information for importing service events from HP OpenView Network Node Manager into Service Desk.
- Appendix B , “Integration With ManageX,” on page 189 contains information useful in setting up a bi-directional service event integration with ManageX.
- Appendix C, “Integrating with LDAP,” on page 203 explains how you can set up an integration with an LDAP directory to import data into Service Desk.
- Appendix D, “Integrating with Radia,” on page 217 explains how you can import Radia data into Service Desk.
- Appendix E , “Examples,” on page 233 includes examples for modifying a configuration file to export data, importing data from a Microsoft® Excel spreadsheet, and from an ASCII text file.

## Revision History

When an edition of a manual is issued with a software release, it has been reviewed and tested and is therefore considered correct at the date of publication. However, errors in the software or documentation that were unknown at the time of release, or important new developments, may necessitate the release of a service pack that includes revised documentation. Revised documentation may also be published on the Internet, see “We Welcome Your Comments!” in this preface for the URL.

A revised edition will display change bars in the left-hand margin to indicate revised text. These change bars will only mark the text that has been edited or inserted since the previous edition or revised edition.

When a revised edition of this document is published, the latest revised edition nullifies all previous editions.

**Table 1**

### Revision History

<b>Edition and Revision Number</b>	<b>Issue Date</b>	<b>Product Release</b>
First Edition	August, 2001	Service Desk 4.0

## Related Publications

This section helps you find information that is related to the information in this guide. It gives an overview of the Service Desk documentation and lists other publications you may need to refer to when using this guide.

### The Service Desk Documentation

Service Desk provides a selection of books and online help to assist you in using Service Desk and improve your understanding of the underlying concepts. This section illustrates what information is available and where you can find it.

---

#### NOTE

This section lists the publications provided with Service Desk 4.0. Updates of publications and additional publications may be provided in later service packs. For an overview of the documentation provided in service packs, please refer to the readme file of the latest service pack. The service packs and the latest versions of publications are available on the Internet. See the section “We Welcome Your Comments!” in this preface for the URLs.

- The `Readme.htm` file on the Service Desk CD-ROM contains information that will help you get started with Service Desk. It also contains any last-minute information that became available after the other documentation went to manufacturing.
- The *HP OpenView Service Desk: Release Notes* give a description of the features that Service Desk provides. In addition, they give information that helps you:
  - compare the current software’s features with those available in previous versions of the software;
  - solve known problems.

The Release Notes are available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Release_Notes.pdf`.

- The *HP OpenView Service Desk: User’s Guide* introduces you to the key concepts behind Service Desk. It gives an overview of what you can do with Service Desk and explains typical tasks of different types of Service Desk users. Scenario descriptions are provided as examples of how the described features could be implemented.

The User's Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `User's_Guide.pdf`.

- The *HP OpenView Service Desk: Supported Platforms List* contains information that helps you determine software requirements. It lists the software versions supported by Hewlett-Packard for Service Desk 4.0.

The Supported Platforms List is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Supported_Platforms_List.pdf`.

- The *HP OpenView Service Desk: Installation Guide* covers all aspects of installing Service Desk.

The Installation Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Installation_Guide.pdf`.

- The *HP OpenView Service Desk: Administrator's Guide* provides information that helps application administrators to set up and maintain the Service Desk application server for client usability.

The Administrator's Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Administrator's_Guide.pdf`.

- The *HP OpenView Service Desk: Data Exchange Administrator's Guide* explains the underlying concepts of the data exchange process and gives instructions on exporting data from external applications and importing it into Service Desk. The data exchange process includes importing single service events and batches of data.

The Data Exchange Administrator's Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Data_Exchange.pdf`.

- The *HP OpenView Service Desk: VantagePoint Operation Integration Administrator's Guide* explains the integration between Service Desk and VantagePoint for Windows and UNIX®. This guide covers the installation and configuration of the integration and explains how to perform the various tasks available with the integration.

The VantagePoint Operation Integration Administrator's Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `VPO_Integration_AG.pdf`.

- The *HP OpenView Service Desk: Migration Guide* provides a detailed

overview of the migration from ITSM 5.7 to Service Desk 4.0, to include an analysis of the differences in the two applications. Detailed instructions in this guide lead through the installation, configuration and other tasks required for a successful migration.

The Migration Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Migration_Guide.pdf`.

- The *HP OpenView Service Desk: API Programmer's Guide* contains information that will help you create customized integrations with Service Desk. This guide depicts the API structure, and explains some of the basic functions with examples for using the Application Programming Interface (API) provided with Service Desk. The API extends the HP OpenView Service Desk environment by providing independent programmatic access to data-centered functionality in the Service Desk application server environment.

The API Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `API_pg.pdf`.

- The *HP OpenView Service Desk: Web API Programmer's Guide* contains information that will help you create customized integrations with Service Desk using the Service Desk Web API. This API is particularly suited for developing Web applications.

The Web API Programmer's Guide is available as a PDF file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Web_API_pg.pdf`.

- The *HP OpenView Service Desk: Data Dictionary* contains helpful information about the structure of the application.

The Data Dictionary is available as an HTML file on the HP OpenView Service Desk 4.0 CD-ROM. The file name is `Data_Dictionary.htm`.

- The *HP OpenView Service Desk 4.0 Computer Based Training (CBT)* CD-ROM is intended to assist you in learning about the functionality of HP OpenView Service Desk 4.0 from both a user and a system administrator perspective. The CD-ROM contains demonstration videos and accompanying texts that explain and show how to perform a wide variety of tasks within the application. The CBT also explains the basic concepts of the Service Desk application.

The *HP OpenView Service Desk 4.0 Computer Based Training (CBT)* CD-ROM will be shipped automatically with the regular Service Desk software. The CBT will be available for shipment shortly after the



release of the Service Desk software.

- The online help is an extensive information system providing:
  - procedural information to help you perform tasks, whether you are a novice or an experienced user;
  - background and overview information to help you improve your understanding of the underlying concepts and structure of Service Desk;
  - information about error messages that may appear when working with Service Desk, together with information on solving these errors;
  - help on help to learn more about the online help.

The online help is automatically installed as part of the Service Desk application and can be invoked from within Service Desk. See the following section entitled “Using the Online Help” for more information.



### **Reading PDF Files**

You can view and print the PDF files with Adobe® Acrobat® Reader. This software is included on the HP OpenView Service Desk 4.0 CD-ROM. For installation instructions, see the `readme.htm` file on the CD-ROM.

The latest version of Adobe Acrobat Reader is also freely available from Adobe’s Internet site at <http://www.adobe.com>.

### **Using the Online Help**

You can invoke help from within Service Desk in the following ways:

- To get help for the window or dialog box you are working in, do one of the following:
  - Press **F1**.
  - Click the help toolbar button .
  - Choose **Help** from the **Help** menu.
  - Click the help command button  in a dialog box.
- To search for help on a specific subject using the table of contents or the index of the help system: choose **Help Contents & Index** from the **Help** menu.



When you are in the help viewer, you can find help on how to use the help system itself by clicking the Help toolbar button:




Service Desk also provides *tooltips* and “*What’s This?*” *help* for screen items like buttons, boxes, and menus.

A *tooltip* is a short description of a screen item. To view a tooltip, rest the mouse pointer on the screen item. The tooltip will appear at the position of the mouse pointer.

“*What’s This?*” *help* is a brief explanation of how to use a screen item. “*What’s This?*” help generally gives more information than tooltips. To view “*What’s This?*” help:

1. First activate the “*What’s This?*” mouse pointer in one of the following ways:
  - Press **Shift+F1**.
  - Click the “*What’s This?*” toolbar button .
  - Choose *What’s This?* from the Help menu.
  - In dialog boxes, click the question mark button  in the title bar.

The mouse pointer changes to a “*What’s This?*” mouse pointer .

2. Then click the screen item for which you want information. The “*What’s This?*” help information appears in a pop-up window.

To close the pop-up window, click anywhere on the screen or press any key on your keyboard.

## Typographic Conventions

The table below illustrates the typographic conventions used in this guide.

Font	What the Font Represents	Example
<i>Italic</i>	References to book titles  Emphasized text	See also the <i>HP OpenView Service Desk: Installation Guide</i> .  <i>Do not delete</i> the System user.
<b>Bold</b>	First-time use of a term that is explained in the glossary	The <b>service call</b> is the basis for incident registration.
Courier	Menu names  Menu commands  Button names  File names  Computer-generated output, such as command lines and program listings	You can adjust the data view with the commands in the View menu.  Choose Save from the menu.  Click Add to open the Add Service Call dialog box.  To start the installation, double-click setup.htm.  If the system displays the text C:\>dir a: The device is not ready then check if the disk is placed in the disk drive.
<b>Courier bold</b>	User input: text that you must enter in a box or after a command line	If the service call must be solved within 30 minutes, enter 30.
<i>Courier italic</i>	Replaceable text: text that you must replace by the text that is appropriate for your situation	Go to the folder x:\Setup, where x is your CD-ROM drive.

<b>Font</b>	<b>What the Font Represents</b>	<b>Example</b>
Helvetica bold	Keyboard keys  A plus sign (+) means you must press the first key (Ctrl in the example), hold it, and then press the second key (F1 in the example).	Press Ctrl+F1.

## **We Welcome Your Comments!**

Your comments and suggestions help us understand your needs, and better meet them. We are interested in what you think of this manual and invite you to alert us to problems or suggest improvements. You can submit your comments through the Internet, using the HP OpenView Documentation Comments Web site at the following URL:

[http://ovweb.external.hp.com/lpe/comm\\_serv](http://ovweb.external.hp.com/lpe/comm_serv)

If you encounter errors that impair your ability to use the product, please contact the HP Response Center or your support representative.

The latest versions of OpenView product manuals, including Service Desk manuals, are available on the HP OpenView Manuals Web site at the following URL:

[http://ovweb.external.hp.com/lpe/doc\\_serv](http://ovweb.external.hp.com/lpe/doc_serv)

Software patches and documentation updates that occur after a product release, will be available on the HP OpenView Software Patches Web site at the following URL:

<http://support.openview.hp.com/cpe/patches>



---

# **1      The Data Exchange Concept**

Data Exchange is the process of exporting information from a data source, formatting it and then importing it into the Service Desk application. A number of example configurable extractors are provided, that can be used to gather data from multiple data sources. The extractor

exports a data exchange file in extensible markup language meeting the common information model, (CIM-XML) making it easy to import. Mapping the import settings in Service Desk ensures the data is imported correctly.

The extractor will work with any third-party database with ODBC, extending the capability of Service Desk even further. The extractor can also be configured to export data from an LDAP directory.

The following chapters provide more detailed information about the extractor and the data exchange process.

---

**NOTE**

For additional information on installing components necessary for data exchange, see the *HP OpenView Service Desk: Installation Guide*.

---



## Data Exchange Overview

The command to exchange data is sent from Service Desk. Data is exported from the external application's database by an extractor configured for the application. The key steps involved in the process are:

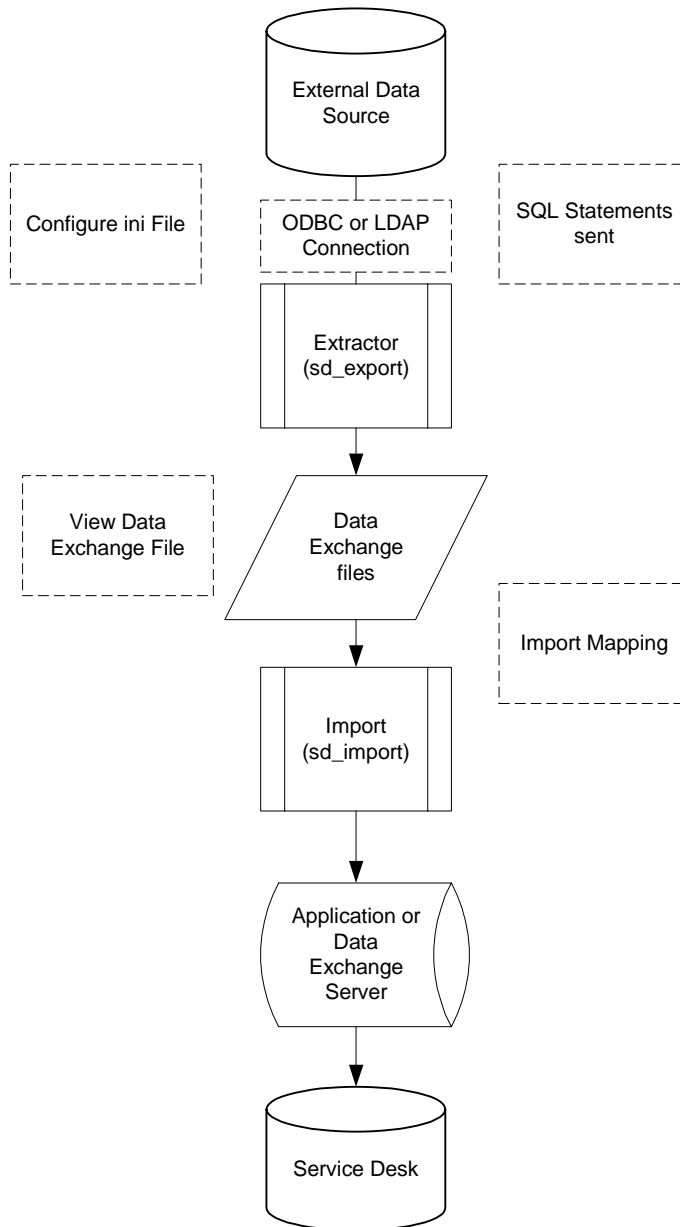
- Step 1.** Create or modify the configuration file using one of the examples provided.
- Step 2.** Establish a connection to your external data source.
- Step 3.** Export the data by running the extractor from the Service Desk application server or a dedicated data exchange server.

Examine the exported XML file with the viewer, if needed configure the extractor and export the data again

- Step 4.** Configure the data import settings by mapping the external application's classes, properties and values to Service Desk items, attributes, and values,
- Step 5.** Import the data from the XML file (data exchange file) into Service Desk. Batch importing is done from the Data Exchange dialog box, see Chapter 4, "Importing Data in Batches," on page 117. The data exchange process for service events is initiated from a command line in the external application, see Chapter 5, "Importing Service Events," on page 149.

Once the data exchange process is configured for an external application, you can save the settings as a task and use the task to export and import data again. It is also possible to use a scheduling program, for more information see "Scheduling a Data Exchange Task" on page 125.

**Figure 1-1 Overview of the Data Exchange Process**



---

**NOTE**

The data exchange process only works if Service Desk runs in three-tier mode. The data exchange process can only be run from an application server or a data exchange server running on a Windows NT® platform. Configuration, XML, and log files can only be accessed from the application server or the data exchange server.

---

## Overview of the Export Process

Most companies use a number of applications to monitor their networks, store human resource data, and inventory information. This information can be imported into Service Desk using Data Exchange. To export data from the external application's database, a special program called an extractor is needed. The extractor has two major tasks. First, to export the information that needs to be imported into Service Desk. Second, to configure the exported information into a format that can be imported properly by Service Desk.

System administrators can create new configuration files by copying one of the examples provided and modifying it. The configuration file makes it possible to establish what data to gather from the external application's database. Changing the configurable file changes the export mapping, giving system administrators the ability to vary the data that is exported into the data exchange files.

When the extractor receives the Export data from a storage device command, it connects to the database of the external application and reads the data it is configured to read.

The following types of information can be defined with the configuration file:

- Database information
- System information
- Element information

The exported data is formatted into an XML file, referred to as a data exchange file, that can be viewed prior to importing into Service Desk. The data exchange file is then imported into the Service Desk database so that the data can be displayed in the Service Desk application.

For more detailed information See "Exporting Data" on page 41.

## Configuring the Import Mapping

It is important to configure the import mapping so that the data imported from another application is interpreted correctly. This process is completed through the use of import mapping dialog boxes in the Service Desk application. To complete this task, it is necessary to know what information is in the external application database and where and how the data must be displayed in Service Desk. Classes will need to be mapped to a template which contains default values. Templates exist for every item available in Service Desk. Attributes, relations, and attribute values will also need to be mapped if you intend to import them into Service Desk.

For more detailed information, See “Import Mapping” on page 93.

## Overview of the Importing Process

Once the import mapping is configured and the configurable extractor is configured to export data from your particular external application, it is possible to start importing data into Service Desk. There are two options available for the purpose of importing data; importing batches of data with `sd_import.exe` or importing service events with `sd_event.exe`.

The batch import process makes it possible to import a number of different items. The data is organized into a preload file; data that is not mapped will be filtered out. The filtered data in the preload file is then imported.

The option to import a service event is recommended for use when there is only one class or configuration item to be imported, for example if you only want to import a service call, or an incident. A command is entered, in the external application, to send data to the Service Desk application server where it is imported. While the data exchange process can only be run on a Windows NT server, the command to import a service event can come from any machine that uses hypertext transfer protocol (HTTP) for Web servers, to include UNIX® machines.

The import mapping defined in Service Desk is used by both the batch and service event importing options. The import mapping used for a service event is selected from the command line, while the import mapping for batch imports is selected in the data exchange dialog box in the Service Desk application. Templates exist for both types of import mapping.

All Data Exchange processes take place through the application server; if you will be importing a large number of items it will be necessary to use a dedicated data exchange server.

For more detailed information See “Importing Data in Batches” on page 117.

---

### NOTE

Old objects and relations are not removed in this version of Service Desk. If you change the unique key(s) of a record Data Exchange will consider it a new item for import. If you do not change the unique key(s), Data Exchange will update the record and overwrite the existing value.

You must manually delete old objects and relations to remove them. See

“Removing Redundant Items and Relations” on page 131 for more information.

---

The Data Exchange Concept  
**Overview of the Importing Process**



---

## **2** **Exporting Data**

To import data into Service Desk it first needs to be exported from the external data source. A number of example extractors are provided with Service Desk that can be configured for your data source.

The extractor's purpose is to take data from where it is stored by one application and put it in a format that can be imported by another application.

The export mapping is set in the configuration file so that it will export only specific data. Any number of classes and properties can be exported. The information is exported into XML formatted files that are compliant with the common information model, also known as CIM-XML. This format is used because it can be read and imported by many different database applications without needing to change the extractor. A DTD file, located in the same folder as your XML file, is used to validate the file, check the tag hierarchy and tag spelling. The file will also be checked for misspelled class and property names via the import mapping, and whether relations reference objects in the exported XML file or not. A text file and a log file are also created by the extractor.

The entire data export process includes approximately 4 steps, which will be explained in the following sections. They are:

- Step 1.** Decide what to export.
- Step 2.** Configure the extractor, using one of the example configuration files provided.
- Step 3.** Define the connection to your external data source.
- Step 4.** Run the extractor to export CIM-XML files.
- Step 5.** View the XML files.

## Deciding What to Export

Before configuring the extractor and setting the export mapping to export data, you need to know what information you want to manage or view in Service Desk. Many organizations use Service Desk in conjunction with other management applications and want to avoid adding inventory or human resource information manually into Service Desk. The following list is a guide to help you determine what to configure the extractor to export:

1. Define the information you want to manage in Service Desk.
2. Evaluate the Service Desk database to see how information is stored. This should result in a list of table names, column names and attributes. You might want to use the Data Dictionary for this purpose.
3. Determine the items you want to export from your external data source.
4. Evaluate the external application data source: Can it provide the information you selected or will you need to use an additional management application? You should develop a list of class names, attributes, key values and relations. You will need to note how the data is organized in the database for use for configuring the extractor. The following section may be helpful in this task.

---

### TIP

Loading only selected columns instead of complete tables will result in a much faster data exchange process.

---

## Checking the Database Structure

One of the most important steps in creating or modifying a configurable extractor file is knowing the table structure. Some applications do not have an object browse tool and it can be difficult to view how information in the external application database is structured. It is crucial that table names and columns are accurate when configuring the extractor. For Oracle® database users it is possible to use SQL\*Plus® for this purpose. You may also use the technical documentation provided with the

## Exporting Data

### Deciding What to Export

external application. Another possible method is to use Microsoft Access as explained below to browse through the data and make simple queries:

1. Start MS Access.
2. Create New, Blank Database.
3. Enter a Name for testing purposes and click Create.
4. From the File menu click Get External Data, then click Link Tables.
5. Select File of Type ODBC databases.
6. Click the Machine Data Source tab.
7. Select the ODBC driver.
8. Select the tables you want to add and click OK.
9. Click Open to view the contents, or Design to view the columns of a table. Use Query building to browse data and links.

---

## Configuring the Extractor

The extractor comes with a number of example extractors. The examples contain files in Windows® initialization format that can be configured. Windows initialization files are used to define and transmit commands. System administrators can use the configuration file to specify precisely what data they want to export and how and where it needs to be exported from the database. The examples provided contain default information and can be used as is or modified. The parameters set in the configuration file are directions sent to the external application database through SQL statements. The SQL statements are executed and the information is exported in the format defined.

When configuring the import settings, keep in mind:

- The class name between [ ] brackets in the configuration file is the class you need to map in the import mapping.
- ATT and PARENT\_RELATION\_NAME determine the attributes that have to be mapped when configuring the import mapping.

To view the configuration files look in the `data_exchange\config` folder of the Service Desk application or open them from the Data Exchange dialog box from within the Administrator Console. A portion of a configuration file follows with the keywords defined in the following section:

### Example 2-1

#### Configuration File

```
[DSN]
NAME=hpdtadb_dta
USR=hpdt
PWD=hpdt

[SYSTEM]
LOG=TRUE
XML=TRUE
OUTPUT_FILE=dta5.xml
```

## Exporting Data

### Configuring the Extractor

```
LOG_FILE=C:\Temp\DTA5.log
XML_OUTPUT_FILE=C:\Temp\DTA5.xml
APPLICATION_NAME=DTA5

ENCODING=UTF-8

SQL_TO_UPPERCASE=TRUE

[CLASSES]
NAME=ADMIN,CPU

[ADMIN]
SOURCE=ADMINISTRATIVE
ID=[MACHID]
ATT=[MACHINENAME],[IPADDRESS]
CONDITION=[OSTYPE]='Windows NT' OR [OSTYPE]='Windows 98'
COLUMNS=[ADMINISTRATIVE].[MACHINEID] as
[MACHID],[ADMINISTRATIVE].[MACHINENAME] as
[MACHINENAME],[ADMINISTRATIVE].[IPADDRESS] as [IPADDRESS]

[CPU]
SOURCE=CPU
PARENT=ADMIN
PARENT_RELATION=[CPU].[MACHINEID]=MACHID
ID=[CLOCKSPEED],[CPUTYPE]
COLUMNS=[CPU].[MACHINEID] as [MACHID],[CPU].[CLOCKSPEED] as
[CLOCKSPEED],[CPU].[CPUTYPE] as [CPUTYPE],[CPU].[REPLICATE]
as [REPLICATE]
```

## Configuration File Keywords

Definitions of the syntax used in the configuration file follow:

**Table 2-1**

### Configuration File Keywords

#### **[CONNECTION] Define the connection type**

**TYPE** The default value is an ODBC connection. You can also select WIN32ODBC, and LDAP. ODBC use the JDBC-ODBC bridge to establish a connection(this driver cannot connect to a WBEM data source). WIN32ODBC uses the native win32 ODBC driver. Refer to “Configuring the LDAP.ini File” on page 207 for information concerning LDAP.

#### **[DSN] Define the data source to be used**

**NAME** Name of the ODBC data source.

**USR** The database user who owns the data source tables and views

**PWD** The database user’s password

#### **[SYSTEM] Define the system and data file settings**

**LOG** Enter TRUE to create a log file created.

**XML** Enter TRUE to create an XML file.

**LOG\_FILE** Enter the location where you want the log file placed. This field is only used when you run Data Exchange from a command line.

**XML\_OUTPUT\_FILE** Enter where you want the extracted XML file placed. This field is only used when you run Data Exchange from a command line.

**APPLICATION\_NAME** Enter the name of the external application you are extracting data from. For example, DTA.

**[SYSTEM]**

**Define the system and data file settings**

ENCODING

Enter the name of the language code you want to use when viewing the XML file. UTF-8 is used as the default.

SQL\_TO\_UPPERCASE

Default is TRUE. Enter TRUE to convert all SQL Queries to uppercase (required for ORACLE databases). Select FALSE to accept upper and lowercase SQL queries.

CLASS\_TO\_XML

Set to FALSE if no value is entered. If set to TRUE, each class that is not a child class mentioned in the [CLASSES] section will be extracted to a separate XML file.

The following example will result in two files(PERSON.xml, and WORKGROUP.xml) in

C:\TEMP\xml\Organization.xml:

```
[ SYSTEM ]
```

```
XML=TRUE
```

```
XML_OUTPUT=C:\TEMP\xml\Organization.xml
```

```
CLASS_TO XML=TRUE
```

```
[ CLASSES ]
```

```
NAME=PERSON, WORKGROUP
```

**[CLASSES]**

**Define what entities will be exported**

NAME

Name of the item you will be importing. You can choose the name yourself. This is the class name that is mapped from the XML file to a Service Desk item when you do the import mapping.

SOURCE

Specify the tables or views to select data from.

ID

Only used in text output file, for backwards compatibility with ITSM.

ATT

Defines the contents of the classes. Alias names of attributes you want to export to XML file. Not all columns specified in COLUMNS have to appear in the XML file.



<b>[CLASSES]</b>	<b>Define what entities will be exported</b>
COLUMNS	Specifies all columns to be selected with an alias [columnname1] AS [alias 1]. Columns that appear in the PARENT_RELATION have to be put into the column list when tables are cached [LOADTABLE=TRUE], because the extractor uses this information to find all children for a particular parent in memory. If an alias is not specified the column name will be used as the alias name.
MAXRECORDS	Use to specify the maximum number of records to be exported.
LOADTABLE	TRUE or FALSE. Specifies whether the records are cached in memory to process parent-child relations faster, or queries are run for each parent to find its children. Use TRUE if you have plenty of memory and want quicker performance. Use FALSE if you have little memory available, it will be slower.
PARENT	Specifies the parent class.
PARENT_RELATION	Specifies the relation between the child and its parent. Left of the equal sign is the column of the foreign key of the child. Right of the equal sign is the alias of the primary key of the parent.
PARENT_RELATION_NAME	Specifies the name of the relation. The default is PARENT.
CONDITION	Provide conditions or selection criteria for the records to be retrieved.
ORDERBY	Specifies the order of the records.

---

**NOTE**

Do not use ID as one of your field names in the configuration file, it will cause an error when exporting. When `sd_export` extracts data and puts it into XML files a field called ID is added by the program to give each record a unique ID in the XML file. As an alternative you can use field

Exporting Data  
Configuring the Extractor

names such as NNM\_ID, or Event\_ID, for example.

---

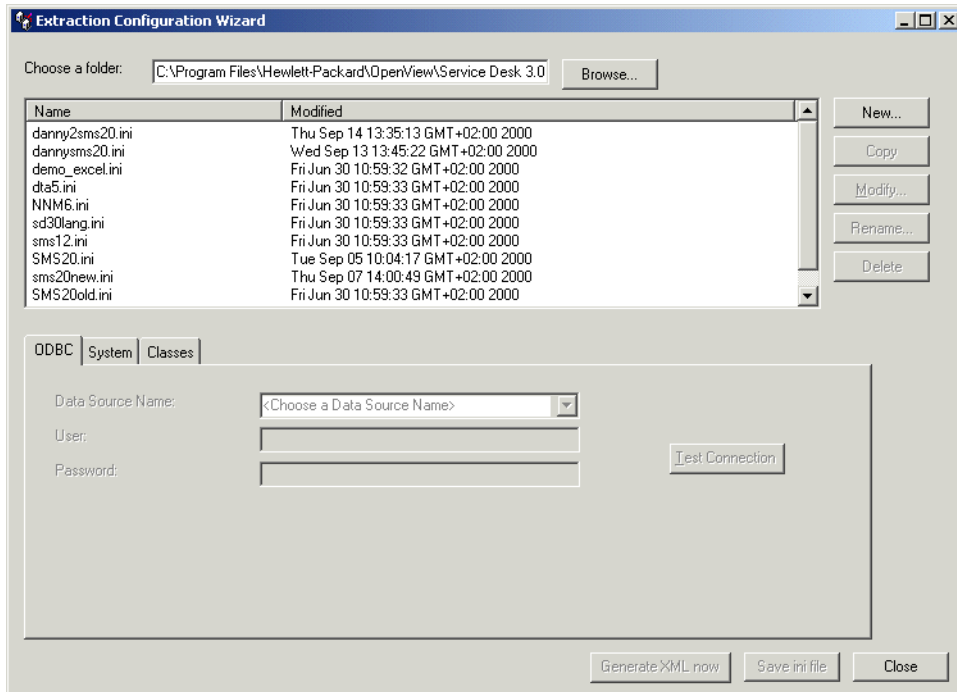
## Extraction Configuration Wizard

The extraction configuration wizard assists you in creating and maintaining an export configuration file without having a deep understanding of SQL, or possessing coding skills. You can select one from the list, copy and modify one from the list, or create a new configuration file. To start the wizard:

1. Select `Programs` from the Start menu.
2. In the `Programs` submenu choose `hp OpenView service desk 4.0`, then `Tools`.
3. Choose the `hp OpenView service desk Extraction Configuration Wizard` option.

The main dialog box shows all configuration files currently available and includes three tabbed pages that mirror the structure of the extractor file:

**Figure 2-1** Extraction Configuration Wizard



These tabs are available when you click New, or select an existing .ini file.

- **ODBC tab:** Use this tab to define the ODBC connection with the external database. This is the DSN section in the configuration file:
- **System tab:** Use this tab to define the system and data file settings.
- **Classes tab:** Use this tab to define the items that will be exported. You can perform the following tasks using the Extraction Configuration Wizard started from the Classes tab:
  - Select source tables
  - Select columns
  - Enter the parent relation
  - Join tables
  - Filter data

- Sort data
- Enter settings for loading classes
- Name and block a class

There are two ways to store the configured file:

- **Generate XML now button:** Use this button to save the .ini file and generate XML code if you need to check the XML code before processing. An XML file name must be entered in the configuration file to perform this step.
- **Save .ini file button:** Use this button to save your settings as an .ini file. This step can be performed at any time during the configuration of the .ini file.

---

**NOTE**

It is possible to maintain configuration files with a text editor, without using the wizard. A text editor allows more flexibility for users familiar with SQL, and the coding of configurable .ini files. Additional information on this subject is available in the *HP OpenView Service Desk: Data Exchange Administrator's Guide*.

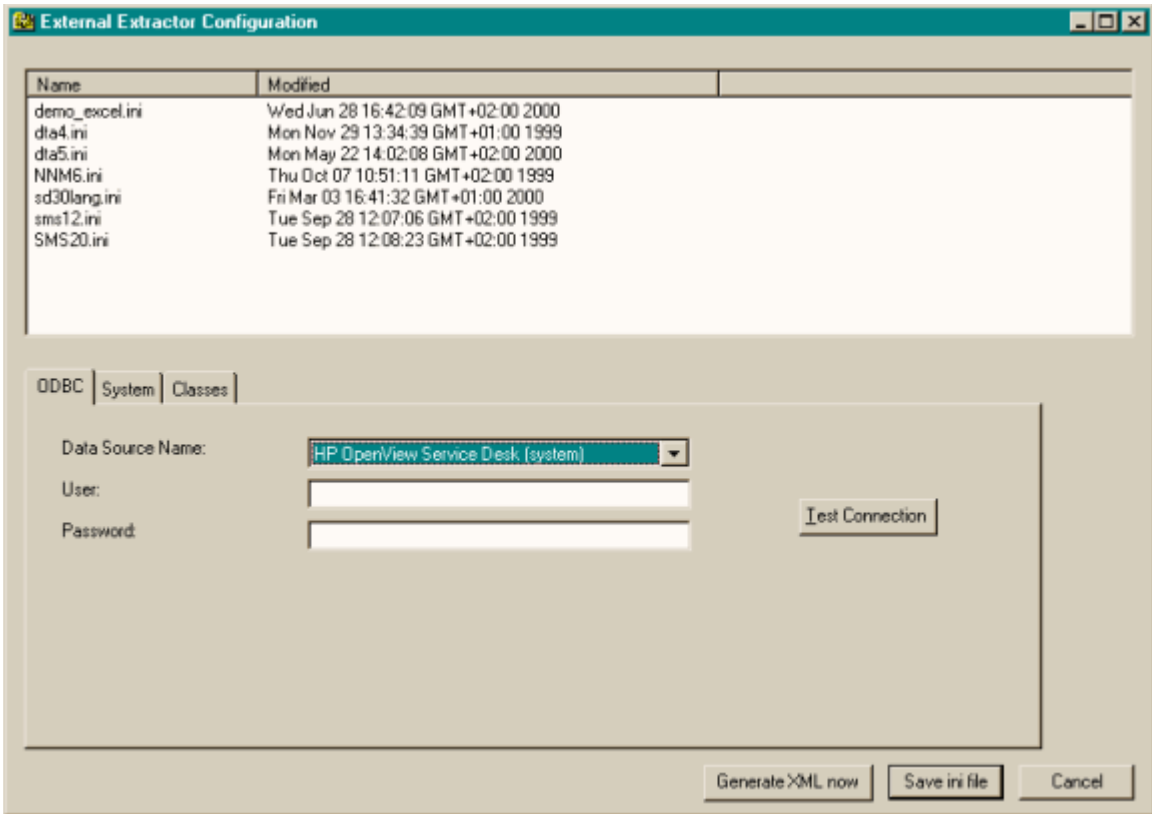
---

## Defining the ODBC Connection

The ODBC tab is used to define the DSN section of the export configuration file. Fill in the tabbed page as follows:

1. In the Data Source Name field, select the name of the external ODBC data Source from the drop-down list:

**Figure 2-2 ODBC Tab**



2. In the **User** field, enter the name of the database user who owns the data source tables and views you intend to select, or have selected from the **Classes** tab.
3. In the **Password** field, enter the password for the user.
4. If you are unsure, click **Test Connection** to verify that the ODBC connection is valid.

**NOTE**

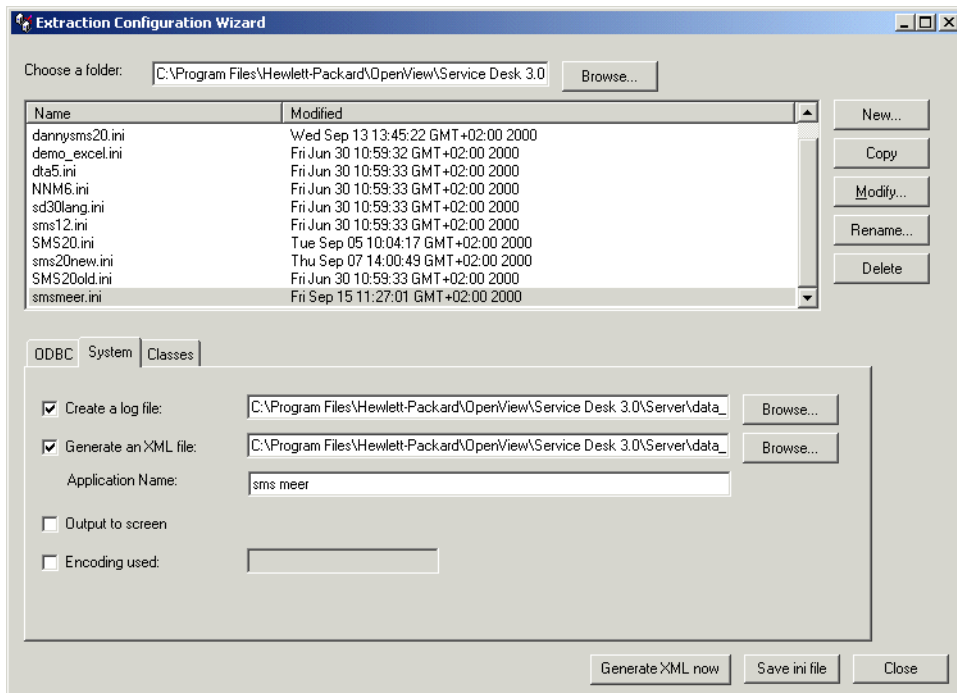
If you have not already done so, you will need to setup an ODBC link for this connection to work. You can enter the necessary information to create an ODBC link by going to the **Start** menu on your desktop, then click **Settings** and then the **ODBC** icon.

## Defining the System Section

The System tab is used to define the output files you want created during the export process:

1. Select the Create a log file check box and browse for the location where you want the log file created:

**Figure 2-3 System Tab**



2. Select the generate XML check box if you want to create an XML file. If you want to import the file into Service Desk, you must select this option. Use the Browse button to select a file location.

---

### NOTE

The `CIM_DTD_V20.dtd` file must be located in the same folder as the generated XML file. If you store your xml files in a location other than the default folder you must copy `CIM_DTD_V20.dtd` to that folder. If you do not, the View XML option in the Data Exchange Task dialog

box will not work.

---

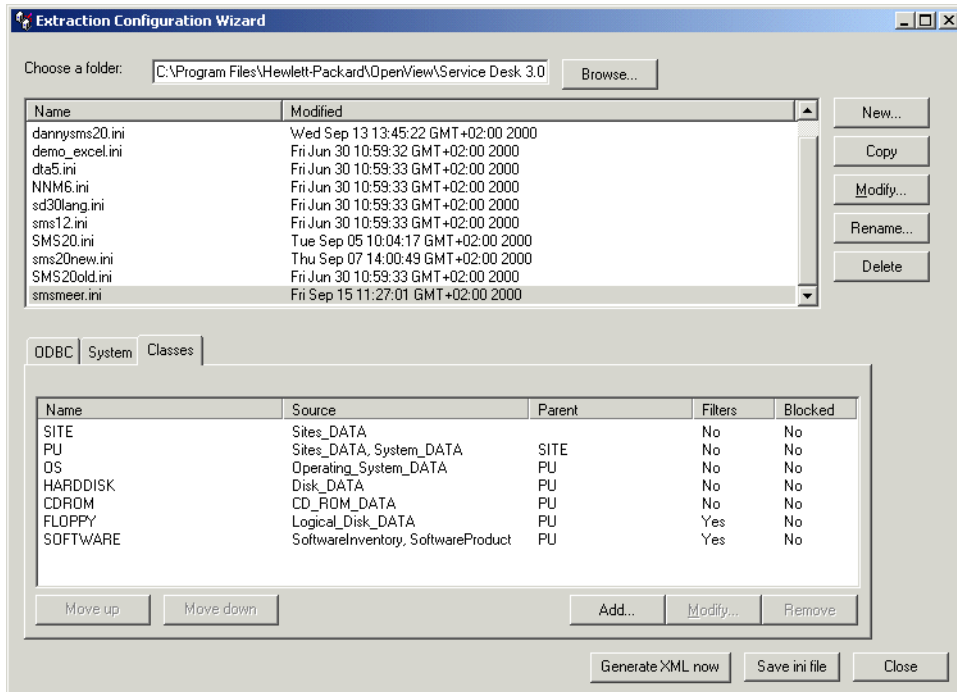
3. In the Application name field, enter the name of the external application you are extracting data from.
4. Select Output to screen to view any extraction errors on your screen as the extraction process runs.
5. The extraction process uses UTF-8 encoding as a default. If you want to use a different code, select the Encoding used check box and enter a different language code.

## **Classes**

The Classes tab is the starting point for defining the classes that you want to export. In the classes tab you will see the names of any classes currently in the configuration file, the table and alias source, the parent for the class, if a filter is applied or if the class is blocked:



**Figure 2-4**      **Classes Tab**



- Use the Move up and Move down button to rearrange the classes in the order that the data should be extracted:
- Click Add to add a class to the configuration file.
- To modify an existing class, select the class and click the Modify button on the classes tab.
- If you click Add or Modify, a wizard page will open for selecting source tables.

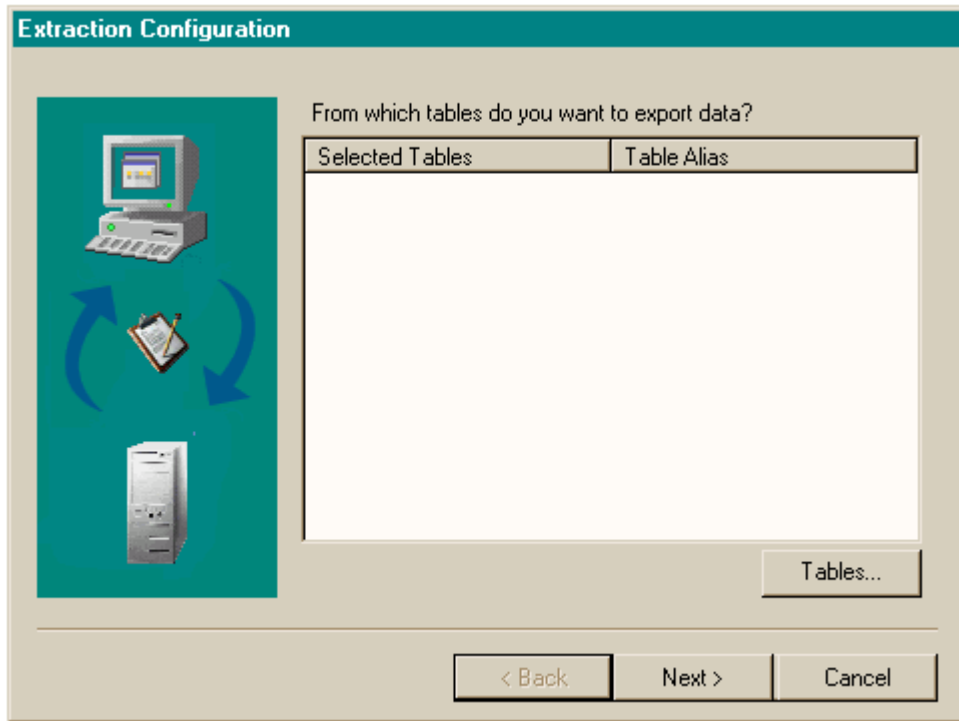
### Select Source Tables

If you are modifying a configuration file, a list of all currently selected tables and their aliases is displayed. When you create a new file this screen will be blank.

To add or remove tables:

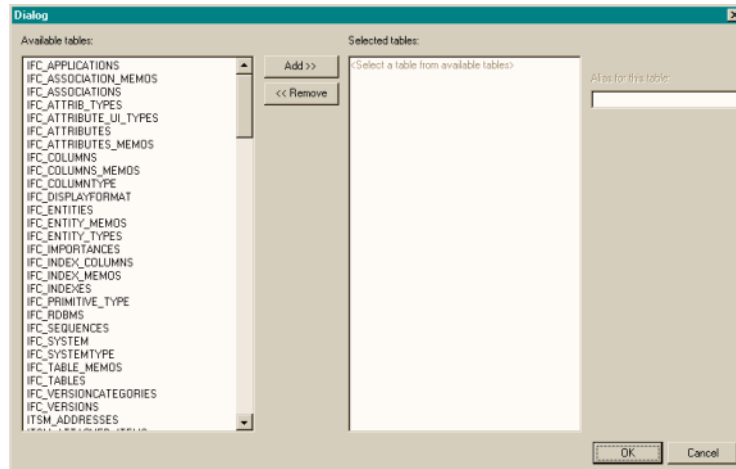
1. Click the Source button:

**Figure 2-5**      **Currently Selected Source Tables**



2. In the dialog box that appears, select one of the available tables from the list and click Add to select it. When it is moved to the Selected tables list you will be able to view the alias for the selected table. You can add, change, or delete the alias name in the Alias for this table field, but it must be unique:

**Figure 2-6**      **Selecting Source Tables**

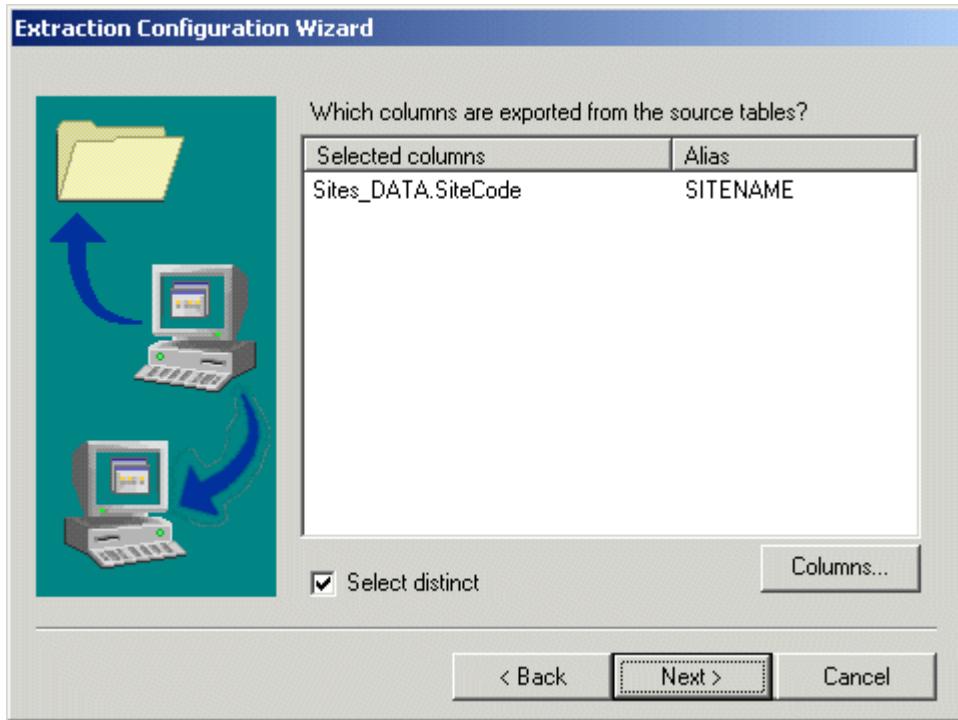


3. Select a table from the Selected tables list and click Remove to remove it. Once a table is removed the alias name is no longer displayed.
4. Use the Describe button to view the columns of a selected table, the Describe pop-up will show the type and values of the columns for the table.
5. Use the Show Value button to view the column values of a selected table.
6. Click OK to return to the Extraction Configuration Wizard, Currently Selected Source Tables wizard page, and Next to continue.

### Select Source Columns

A wizard page appears showing the columns currently selected to be exported from the source tables:

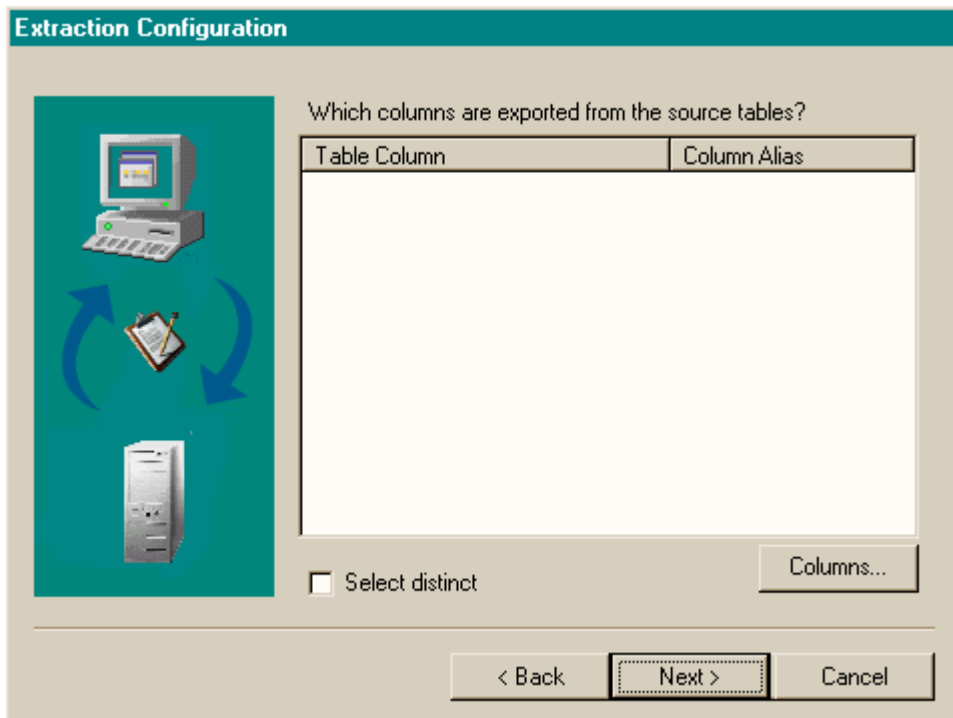
**Figure 2-7**      **Currently Selected Table Columns**



To select additional data source columns:

1. Click the `Columns` button to open the table column dialog box where you can add, or remove columns. All the columns of the previously selected tables are available:

**Figure 2-8**      **Select Table Columns**



2. Choose a table from the `Source` drop-down list. The columns of the selected table are listed on the left-hand side of the screen. Select a column from the sources list and click `Add` to move it into the selected list or `Remove` to remove items from the selected column list.
3. Each column can have an alias. To add an alias for a column, select the column and enter a name in the `Alias` field. The alias name you create must be unique.
4. The `Free Text` button enables you to enter SQL code to adjust column values. To use `Free Text` to enter SQL code:
  - a. Click `New Field`. This creates an entry in the `Selected columns` field.
  - b. Select the new entry, and click the `Free text` button. The `Free text` dialog box is displayed.
  - c. In the `Free text` dialog box, enter the required SQL code and field names in the `Enter free text for this column field`.

- d. The `Fields` button enables you to choose fields from the selected columns to be inserted at the position of the cursor in the `Free text` field.
  - e. The `Show Column Results` button lists the resulting values produced by the SQL statement in a pop-up dialog box.
  - f. Click `OK` to return to the `Select table columns` dialog box.
5. Use the `Show Value` button to view the real values of the selected columns.
  6. Use the `Block this column` check box to block the currently selected column. You may want to do this, for example, when a column is in a parent relation but you do not want to extract the data. When a column is blocked, it will not be listed behind `ATT=`.
  7. Click `OK` to return to the `Extraction Configuration Wizard, Select Source Columns` wizard page.
  8. Choose the `Select distinct` check box to ensure that duplicate columns are not extracted.
  9. Click `Next` to continue.

### **Enter the Parent Relation**

One parent class can be entered for each class. To enter a parent for the class:

1. Select the `Yes` option in answer to the question "Does this class have a parent?". The default is `No`, if you do not want to create a parent relation leave the setting as `No` and click `Next`:

**Figure 2-9 Enter Parent Relation**

Extraction Configuration Wizard

Does this class have a parent?

Yes  No

Relation name:

How is the parent related to this class?

Fields

Logical\_Disk\_DATA.MachinelD

Equals

Attribute

PU System\_DATA.IMACHID

< Back Next > Cancel

2. Select the class name of the parent in the field on the right from the drop-down list containing all of then previously entered class names.
3. The Relation name is optional.
4. Click **Fields** to open a list that includes all tables previously selected for this class. The list will include all columns available for the selected tables. Select the column or variable name that is related to the parent.
5. Click the **Attribute** button to open a list of previously selected tables for the parent class. The list will include alias names or columns available for the selected tables. Select the appropriate column. This information is entered after the ATT line of the parent in the configurable .ini file.
6. Click **Next** to Continue.

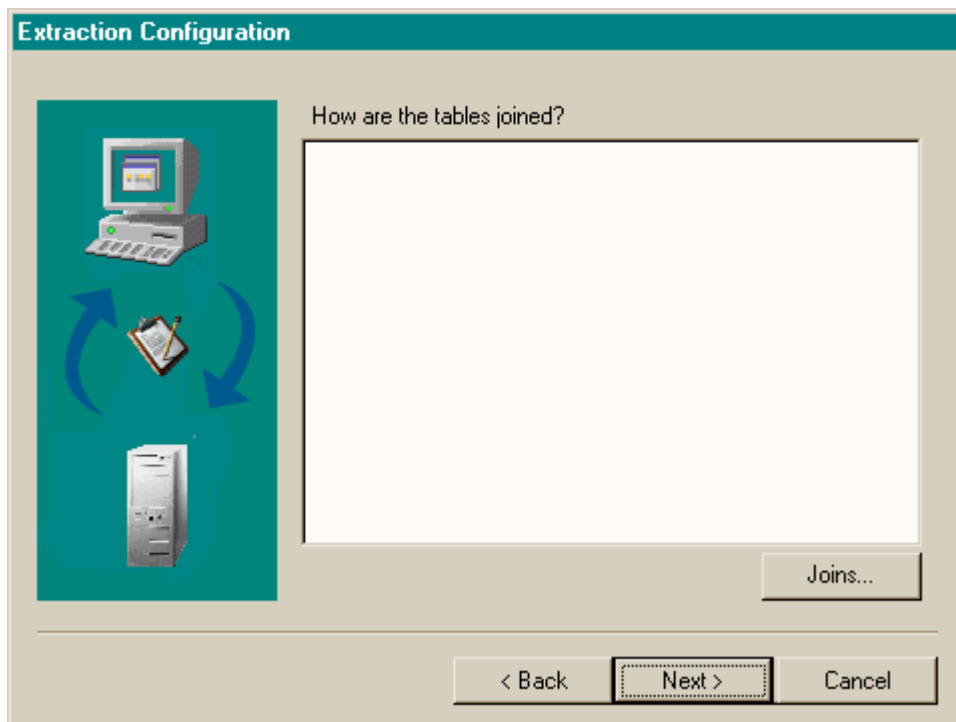
### Join Tables

Tables can be joined to help link information in columns. For example, you may want to link information from one column to a code table definition listed in another column.

In this dialog box you can see all joined tables for this class:

1. Click `Joins` to define the conditions for joining tables:

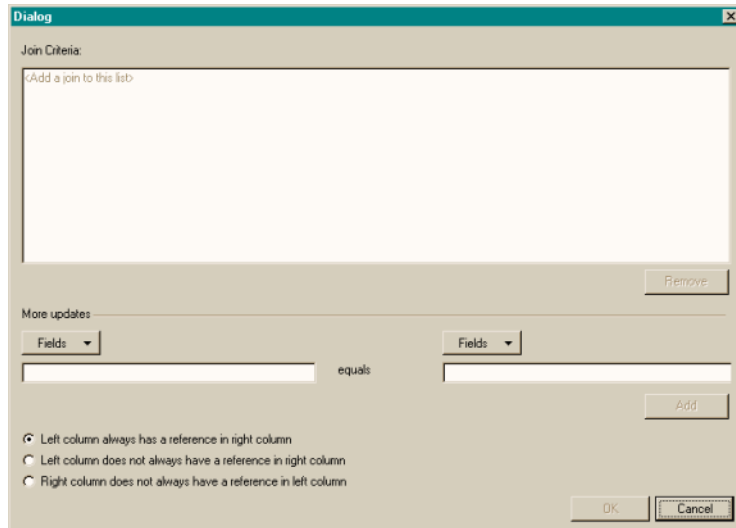
**Figure 2-10**     **Joined Tables**



2. To add a new joining condition, click the `Fields` button on the left portion of the dialog box. A list of all selected tables and all columns available for those tables will be displayed:



**Figure 2-11 Add a Joining Condition**



3. Select the table and column.
4. Click **Fields** on the right portion of the dialog box. A list of all selected tables and all columns available for those tables will be visible. Select the column you want to join with the one selected in step 2.
5. Click **Add** to add the joining condition to the list.

---

**NOTE**

The syntax used by the wizard for a join is a comma (.). When a join is created by the above method this syntax is automatically used, however if you are modifying an existing .ini file any existing joins that use LEFT JOIN, JOIN, or RIGHT JOIN will not be displayed in the wizard. You must replace these joins manually using a text editor, before they can be displayed.

6. The following options are only available for Oracle® database users. Select one of the following options:
  - Left column always has a reference in right column, this refers to an outer join.
  - Left column does not always have a reference in right column, this

## Exporting Data Extraction Configuration Wizard

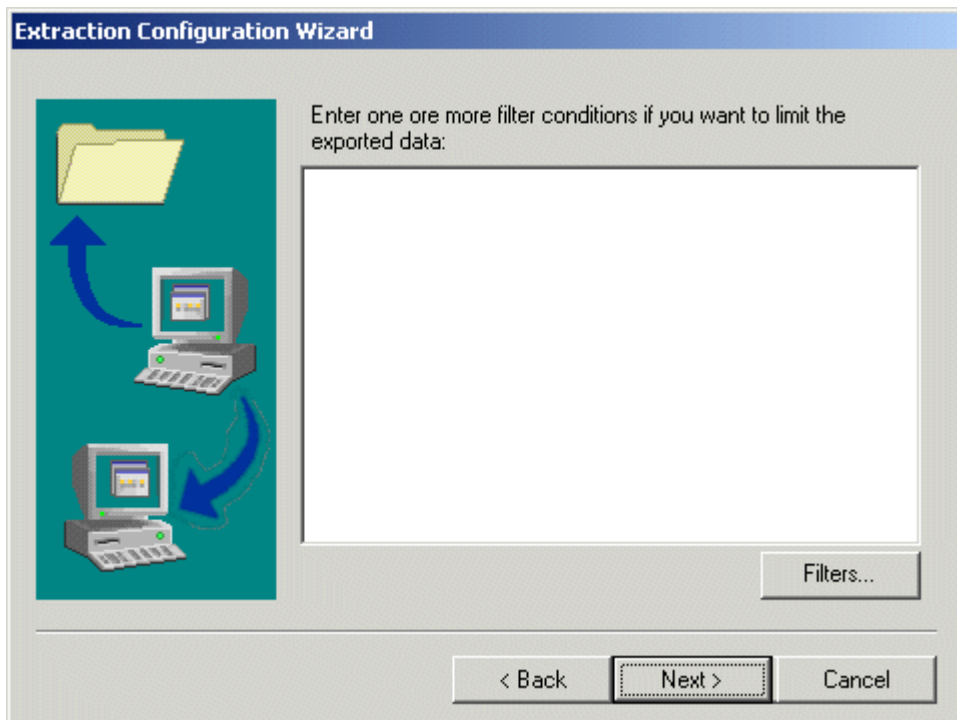
refers to a left outer join. The left column may be Null.

- Right column does not always have a reference in left column, this refers to a right outer join. The right column may be Null.
7. Click **OK** to return to the Extraction Configuration Wizard, Join Tables wizard page.
  8. Click **Next** to continue.

### Filter Data

Filters can be defined for each class being exported. Each filter can then be blocked or activated from a subsequent Filter conditions dialog box:

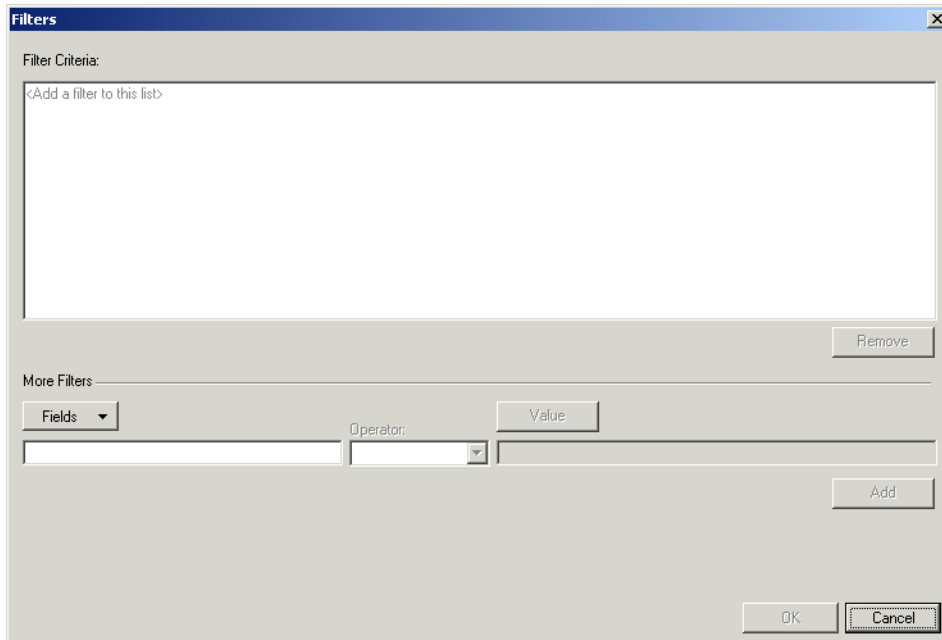
**Figure 2-12**      **Filter Data**



To create a new filter definition:

1. Click the **Filter** button to open the Filter conditions dialog box:

**Figure 2-13 Add a Filter Condition**

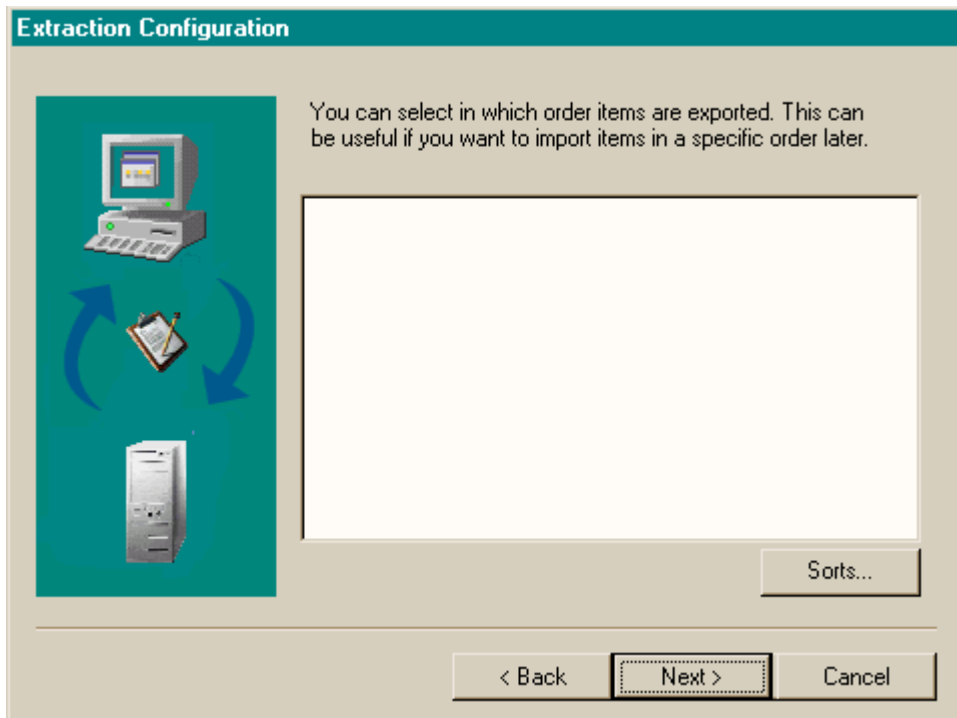


2. Click **Fields** to select the table and column that you want to apply the filter to.
3. In the **Operator** field, use the drop-down list to select an operator. The Operators available vary depending on the field type, as different operators are needed for text, numbers, or dates. The operator selected may also affect what you do with the **Value** field. For example, if the field type is numerical and the **Between** operator is chosen, two value fields are displayed. First place the cursor in one field, click **Value** and select a value from the list, place the cursor in the other field and repeat the action. For a different field type the **In** operator will show a list of check boxes you can select.
4. Click **Value** to select the value range.
5. Click **Add** when you are finished with the definition. You can add multiple filter definitions for each table column selected.
6. Click **OK** to return to the Extraction Configuration Wizard, Filter Data wizard page.
7. Click **Next** to continue.

### Data Sort Order

Use this wizard page to select the order in which data is exported. This can be an important step if the data being exported needs to be imported in a specific order. This screen initially shows sort order currently defined in the database, to change it:

**Figure 2-14** Current Sort Order



1. Click Sort by to open another dialog box where you can establish the sort order:

**Figure 2-15**     **Sorting Data**

The screenshot shows a 'Sort By' dialog box with a teal title bar. It contains four sections for sorting criteria. Each section has a 'Fields' dropdown menu, a text input field, and two radio buttons for 'Ascending' and 'Descending'. The first section has 'IFC\_COLUMNS\_MEMOS.COM\_OID' in the first field and 'Descending' selected. The other three sections are empty. At the bottom right are 'OK' and 'Cancel' buttons.

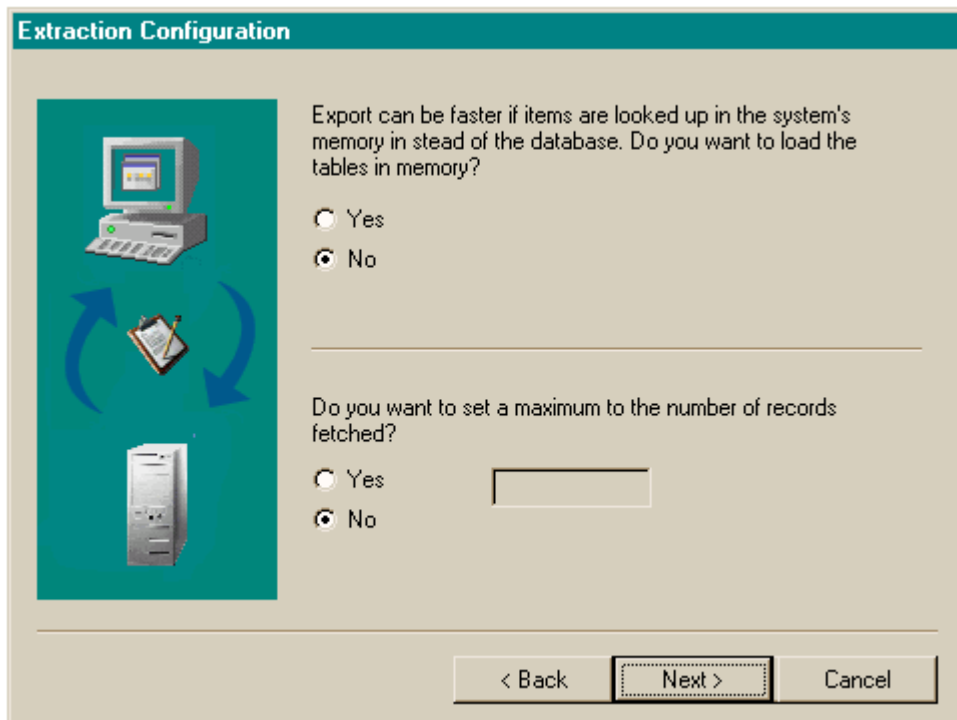
2. Click **Fields** to view a list of all tables and columns previously selected. You can enter a maximum of four table columns per class.
3. Click on the table column you want to sort by to enter it in the field.
4. Select the **Ascending** or **Descending** option for each table column entered.
5. Click **OK** to return to the **Extraction Configuration Wizard, Current Sort Order** wizard page.
6. Click **Next** to continue.

### Enter Settings for Loading Classes

Use this option to achieve a faster extraction rate.

- If you select the Yes option, tables will only be loaded in the system memory and not in the database when exported. This will speed up the processing time. You can select this option for each class.

**Figure 2-16** Enter Settings for Loading Classes



- To set a maximum number of records fetched, select the Yes option in the lower half of the dialog box and then type in the number of records. This will limit the number of records extracted from the database to the number entered.
- Click **Next** to continue.

### Name and Block a Class

Use this wizard page to give a name to the settings you have just defined. This name will be displayed in the classes of the wizards main screen.

1. Enter a Name for this class, do not use spaces in the name. This name will appear as the class name in the exported XML file. If you configure the import mapping in Data Exchange you will map this name from the XML file to the correct item in Service Desk:

**Figure 2-17 Name and Block a Class**

Extraction Configuration

Enter a name for this class:

You can block this class if you do not want to use it now.

Blocked

< Back Finish Cancel

2. Select the **Blocked** check box if you do not want to export the class. You can change this option for each class as often as necessary.
3. Click **Finish** to save all of your settings to a temporary table and return to the **Classes** tab page.

---

**NOTE**

The setting defined above will not be saved to the .ini file until you click **Save .ini file** on the main **Extraction Configuration Wizard** page, see “**Extraction Configuration Wizard**” on page 51.

---

## Defining Relations in the Configuration File

You can specify how relations should be imported in the export configuration .ini file. The .ini file is also where you specify how data is joined and sorted, not during the import mapping phase. There are three primary types of relations that can be imported; N:1 relations based on a search key, N:N or Parent-Child relations, and 1:Rel:1 that involve multiple classes. The following sections will explain the relation types in more detail and include information on how to configure the configurable .ini file to export them.

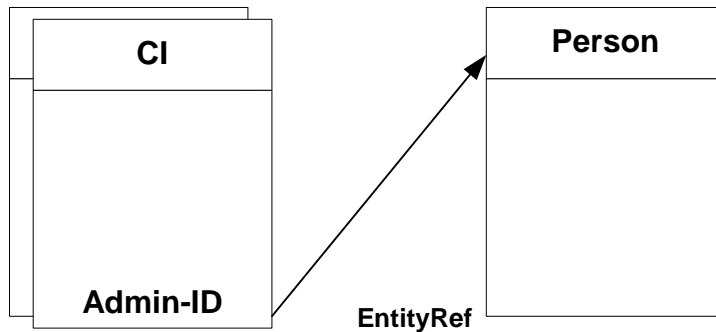
The Data Dictionary provided with Service Desk provides a description for all items and tables in Service Desk to include the relations between those items. For example, if the attribute for an item has a 1:1 relation, e.g. Service call caller to the Person item, the table where the other item is stored will be listed. If the attribute has a 1:N relation, e.g. Service call History lines to History line items, the table where the items are stored will be shown along with the table column that contains the foreign key.

### Search Key; N:1 Relations

In the N:1 relation the N refers to one or more items in Service Desk related to one (1) other item in Service Desk. This type of relation uses a search code as an entity reference to determine the relationship between the items. When modifying the configuration file you need to make sure that the item that is referenced using a search code is exported first. The order is determined by how the classes are listed in the [CLASSES] section. If the referenced entity is not exported to the XML source file first, an error will result when importing. The following diagram shows



this relation with the Admin CI referencing the Person item:



Following is a portion of a configuration file demonstrating how to export this type of relation, note that PERSON is configured to be exported before CI\_ADMIN\_RELATION:

```
[PERSON]
SOURCE=[EMPLOYEE]
ATT=[PERSON_ID], [PERSON_NAME]
COLUMNS=[PERSON_ID] , [PERSON_NAME]
LOADTABLE= TRUE
```

```
[CI_ADMIN_RELATION]
SOURCE=[EQUIPMENT]
ATT=[ADMIN_ID], [CI_ID]
COLUMNS=[ADMIN_ID],[EQUIPMENT_ID] AS [CI_ID]
LOADTABLE=TRUE
```

To complete the process for importing this relation you will need to create the import mapping to use a search code as a reference to the item. A complete import mapping for this example is available at; “Search Key; N:1 Relations” on page 97.

For another importing example see “Importing Data With Relations” on page 245.

## Parent-Child; N:N Relations

In this type of relation the N refers to one or more items in Service Desk related to N, one or more other items in Service Desk. This type of relation uses multiple item search codes or an entity set reference to

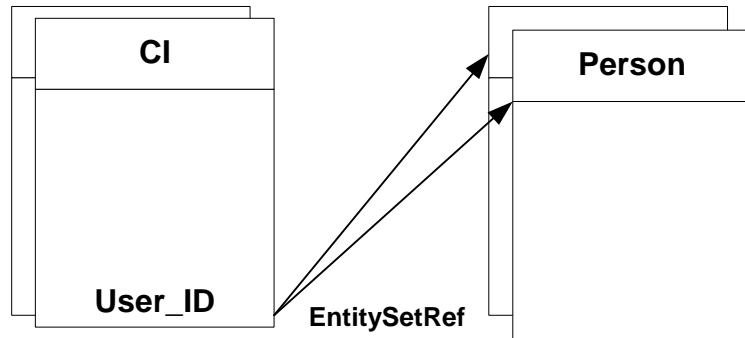
make the relationship. The configuration file is modified differently for a N:N relation than a N:1 relation. For the N:N relation you can use PARENT, PARENT\_RELATION, and PARENT\_RELATION\_NAME fields to specify the Parent-Child relation. The following diagram demonstrates this relation with multiple configuration items. Users are related to Person items using the EntitySetRef. The relationship can be made in either direction:

---

**NOTE**

When you configure the .ini file to import a parent relation, add the CONDITION: IS NOT NULL to the line following the ATT statement for the PARENT. This is an error trapping statement that will prevent an error from occurring when you try to import a CI that has no parent.

---



Following is a portion of an .ini file demonstrating how to export this type of relation:

```
[ CI_USER_RELATION_PARENT ]
SOURCE=[ EQUIPMENT ]
ATT=[ USER_ID ]
COLUMNS=[ USER_ID ], [ EQUIPMENT_ID ] AS [ CI_ID ]
LOADTABLE=TRUE

[ CI_USER_RELATION_CHILD ]
SOURCE=[ EQUIPMENT ]
ATT=[ CI_ID ]
COLUMNS=[ EQUIPMENT_ID ] AS [ CI_ID ]
```

```
LOADTABLE=TRUE  
PARENT=CI_USER_RELATION_PARENT  
PARENT_RELATION=[EQUIPMENT_ID]=[CI_ID]  
PARENT_RELATION_NAME=PARENT
```

To finish the process and import this relation you will need to create an entity reference in the import mapping.

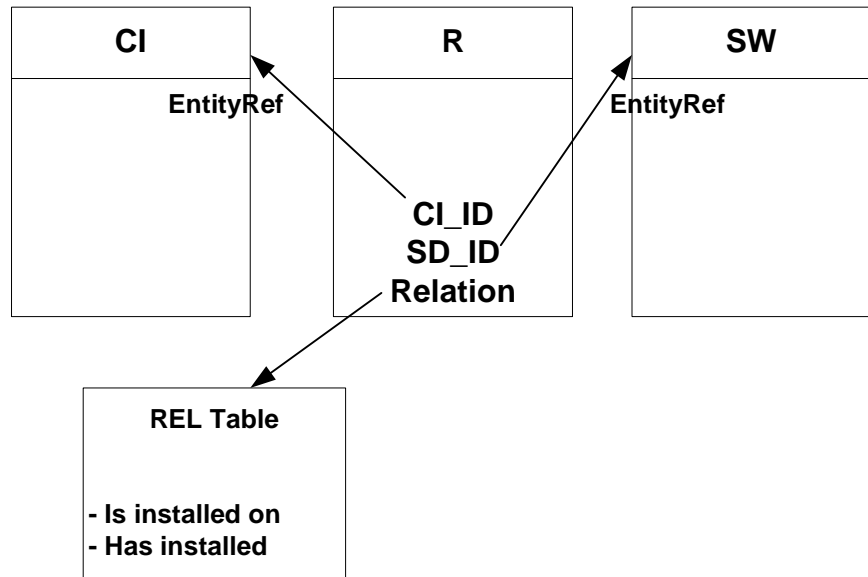
To complete the process for importing this relation you will need to create the import mapping to use a search code as a reference to the item. The complete import mapping for this example can be found at, “Parent-Child; N:N Relations” on page 100.

For another importing example see “Importing Data With Relations” on page 245.

## **Relations with Relation Type; 1:Rel:1 Relations**

The 1:Rel:1 type relation is a relation that is specified by creating another class for the purpose of describing the relation between two or more classes. You can use an entity reference with a search code and an entity reference to a relation table in Service Desk to describe the relation. When using a reference to a relation table, the relation information is maintained in a separate relation code table. You can map items to the values in the relation table to define the type of relation. You can also create value mapping for the relations.

Exporting Data  
Defining Relations in the Configuration File



Following is a portion of an .ini file demonstrating how to export this type of relation:

```
[ CI_ID_NAME ]
SOURCE=[ EQUIPMENT ]
ATT=[ CI_ID ]
COLUMNS=[ EQUIPMENT_ID ]AS [ CI_ID ]
LOADTABLE=TRUE

[ SW_ID_NAME ]
SOURCE=[ EQUIPMENT ]
ATT=[ SW_ID ]
COLUMNS=[ SW_ID ]
LOADTABLE=TRUE

[ SW_REL_RELATION ]
SOURCE=[ EQUIPMENT ]
ATT=[ CI_ID ], [ SW_ID ]
COLUMNS=[ EQUIPMENT_ID ]AS [ CI_ID ], [ SOFTWARE ]. [ SW_ID ],
LOADTABLE=TRUE
```

To complete the process for importing this relation you will need to create

the import mapping to use a search code as a reference to the item. Refer to the following section for the complete import mapping, “Relations with Relation Type; 1:Rel:1 Relations” on page 102.

For another importing example see “Importing Data With Relations” on page 245.

## Defining the ODBC link

An open database connectivity driver (ODBC) allows applications to communicate with a number of other ODBC compliant databases. To use ODBC to connect the extractor and the external application database follow these general steps:

- Step 1.** Click **Start** on your desktop, then **Settings**, then click **Control Panel** and then the **ODBC** icon (this may differ depending on your environment).
- Step 2.** Click the **System DSN** tab. Service Desk uses the **System DSN for Data Exchange**, and all examples are provided with that type of DSN in mind. System DSN is used when the data source is local to a computer, rather than dedicated to a user. The system, or any user with access privileges, can use a data source set up with a system DSN.

- Step 3.** Choose the appropriate driver for your application, examples are provided in the appendices of this guide, for example:

DTA 5.0: Solid ODBC driver 3.0

DTA 4.0: DTA 4.0 ODBC Driver (32-bit) or DTA 5.0 ODBC Driver.

- Step 4.** Select **Finish**.

- Step 5.** Fill in the following fields, dependent upon the application:

*DTA 5:*

Data Source Name: HPDTADB\_DTA

Description: DSN for HP DTA SOLID database

Network Name: TCPIP <server name> 1313

If your DTA 5 server is different from the Service Desk server you still need to have the SOLID ODBC driver installed on the Service Desk application server.

*DTA 4:*

Data Source Name, example: my\_dta\_source

Select Data File, example: F:\HPDTA\DATA\NADMIN.DBD

---

### NOTE

The location for configuring your ODBC connection and the drivers listed may vary depending on the environment (Windows 2000, NT or others)

and the programs that you have installed.

---

---

**NOTE**

If you are connecting to an LDAP directory, refer to Appendix C, “Integrating with LDAP,” on page 203.

---

---

**TIP**

If you are using *Microsoft System Management Server* the following may be useful:

Use the SQL server driver

Data Source Name: my\_sms\_source.

Description: Microsoft Systems Management Server

Server: Enter you server name

Click Next and select SQL server authentication, then enter the Login and Password for the server.

Click Next, and then Finish.

---

## The Extraction Process

All data exchange processes must be run from either the application server or a dedicated data exchange server. The option exists to export data in one step and then import it at a later time or export the data and import it immediately afterwards.

SQL statements are generated by class definitions established in the configuration file and are passed to the ODBC driver. Tables, columns, and classes are selected, and relationships made according to the definitions in the configuration file. The extractor processes the SQL statements in the following manner:

**Step 1.** Starts with the first class without parents.

**Step 2.** Creates an SQL statement.

**Step 3.** For each record:

- a. Extracts the first record
- b. Puts it in the XML file.

**Step 4.** For all child classes:

- a. If `LOADTABLE=TRUE` or if no records are loaded, creates an SQL statement to load all records in memory and uses the `PARENT_RELATION` to scan through the records in memory to find the children.

else

- b. Creates an SQL statement to get the records belonging to the parent using the `PARENT_RELATION`.

**Step 5.** Processes all child records recursively.

**Step 6.** The following records are created from the class section:

```
table loaded LOADTABLE=TRUE or no child class:  
SELECT <COLUMNS>  
FROM <SOURCE>  
WHERE <CONDITON>  
ORDER BY <ORDERBY>
```



Table not loaded *LOADTABLE=FALSE* and it is a child class *PARENT=*:  
SELECT <COLUMNS>  
FROM <SOURCE>  
WHERE <CONDITON>AND<PARENT\_RELATION (with the right side  
filled in.)  
ORDER BY <ORDERBY>;

joins (LEFT JOIN, etc.)  
SELECT <COLUMNS>  
FROM <SOURCE>ON<CONDITION>  
ORDER BY<ORDERBY>

## Exporting Data

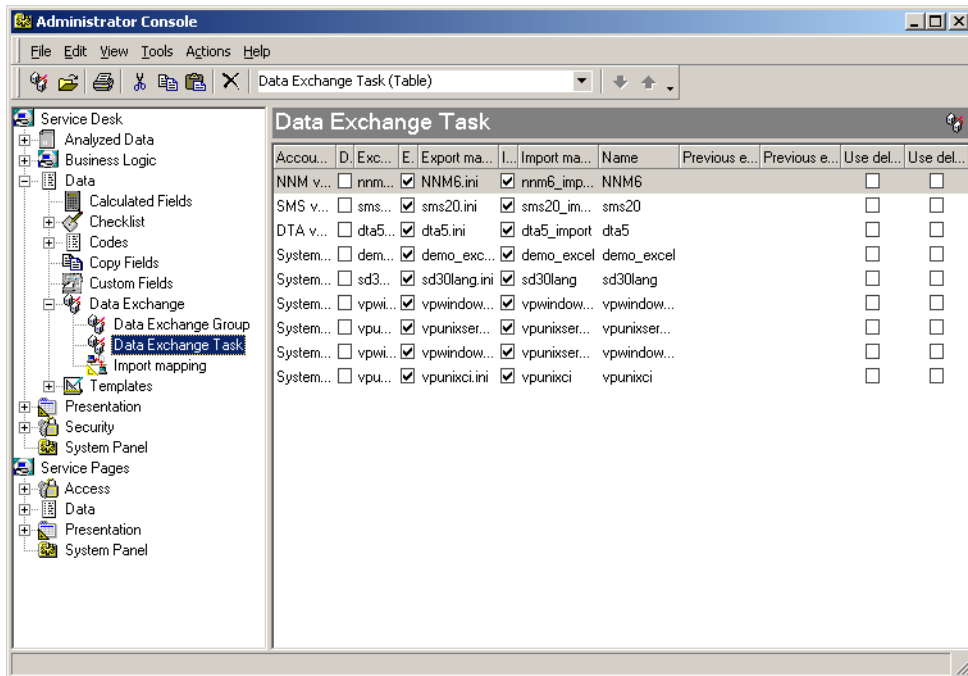
To export data follow the steps in the following procedure.

### Using a Task to Export Data From a Storage Device

**Step 1.** Open a data exchange task.

1. From the **Tools** menu click **System**, open the **Data** folder in the **Administrator Console** and double-click the **Data Exchange** icon to open it.
2. Double-click the **Data Exchange Tasks** icon. Existing data exchange tasks will be visible in the window. To export, the task needs to include the correct: **Export** mapping, **Exchange** file name , and the **Export** check box must be selected:

**Figure 2-18** Data Exchange Task

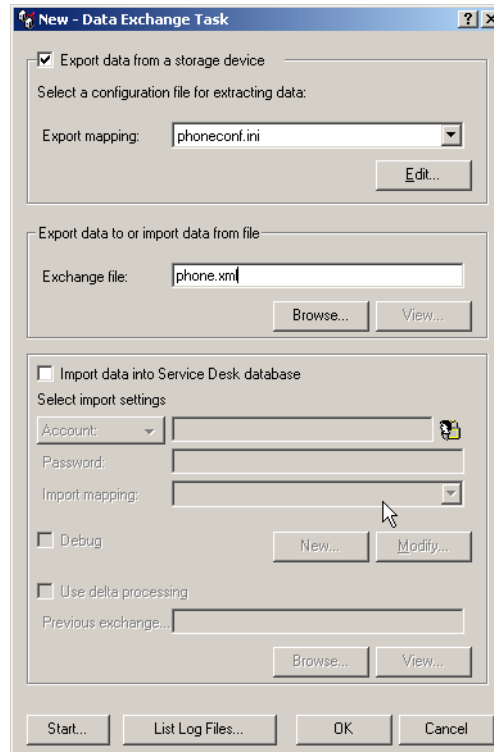


3. Double-click the task you want to use. The task will be opened in the Data Exchange dialog box.

**Step 2.** Configure the task for exporting.

1. Select the Export data from a storage device check box

**Figure 2-19** Data Exchange dialog box - Exporting Data



2. In the Export mapping field enter the configuration file you want to use with the extractor to export the data. You can choose one from the list box. Click Edit if you want to modify the configuration file selected. The file will be opened in a text editor.
3. In the Exchange file field, enter the name of the exported XML file you want to import after the export process has taken place. The XML file name is set in the extractor's configuration file. Click Browse to search for a file. Click the View button to open the exported XML file in an object-tree format to verify that it is accurate.

**Step 3.** Run the task.

1. Clear the `Import data into Service Desk database` check box and the `Account` field if you do not want to import data at this time. For information on importing data, see Chapter 4, “Importing Data in Batches,” on page 117.
2. Click `Save` to save the modified task, and `OK` to export data.
3. To view log files of the export process click `List log files` at any time during or after the data exchange. Select a log file from the list, right-click and then click `Open`.

## Creating a Task to Export Data

To create a new task for exporting data:

1. If a task does not yet exist you can modify an existing task or you can create a new task by right-clicking with the mouse anywhere in the `Data Exchange Task` window and selecting `New Task` from the menu that will appear. Another option is to select the `task` button on the toolbar. It is located directly below the `File` button:
2. An empty data exchange dialog box will appear. Enter the information you want to use for this task and click `Save` at the bottom of the dialog box. The task will appear in the `Data Exchange Task` window.  
For more information about using Data Exchange Tasks see “Creating Data Exchange Tasks” on page 124.

## Relating Data Exchange Tasks

You can relate multiple Data Exchange tasks and execute them as one Data Exchange Task Group. After creating individual tasks in Data Exchange you can relate them and execute them, in order, with one `Start` command. The Task Group is not designed to run from a command line.

For more information about relating tasks to create a Data Exchange Task Group, see “Creating a Data Exchange Task Group” on page 126.

## Exporting From the Command Line

To export data using a command line instead of the Data Exchange

dialog see “Command Line Parameters” on page 144.

## The Data Exchange Files

The extractor creates a CIM-XML formatted file, a text formatted file, and a log file. Any XML file can be imported into Service Desk as long as it complies with the CIM-DTD. Currently CIM-DTD version 2.0 is supported.

Each CIM-XML file contains a header with information in it such as the date and the application the data was exported from. Start and end tags are used to specify elements and properties within the file. Element tags are labeled INSTANCE CLASSNAME with the name for the item after it. The tag labeled PROPERTY denotes attributes of the INSTANCE. Each INSTANCE must have a PROPERTY named ID. The PROPERTY called ID must have a unique VALUE that is used to identify that specific item, such as a serial number. INSTANCES that are children will contain the tag PROPERTY.REFERENCE NAME =PARENT followed by values expressing the unique ID of the parent.

---

### NOTE

The `CIM_DTD_V20.dtd` file must be located in the same folder as the generated XML file. If you store your xml files in a location other than the default folder you must copy `CIM_DTD_V20.dtd` to that folder. If you do not, the View XML option in the Data Exchange Task dialog box will not work.

---

## The XML Format

Extensible markup language (XML) is similar to the HTML used in many Web pages. It provides a flexible way of creating common information formats making it a simple way to share the format and data across to other applications. Both XML and HTML contain markup symbols to describe the contents of a page or file. HTML, however, describes the content (mainly text and graphic images) only in terms of how it is to be displayed and interacted with. For example, a `<P>` starts a new paragraph. XML describes the content in terms of what data is being described. For example, a `<PHONENUM>` could indicate that the data that followed it was a phone number. This makes it possible for an XML file to be processed purely as data by an application. For example, depending on how the program in the receiving computer wanted to

handle the phone number, it could be stored, displayed, or dialed. XML is “extensible” because, unlike HTML, the markup symbols are unlimited and self-defining.

### Example 2-2 Example XML File

```
<?xml version="1.0" encoding='ISO-8859-1'?>
<!DOCTYPE CIM SYSTEM
"file:CIM_DTD_V20.dtd"

[<!ENTITY lt      "&#38;#60;">
  <!ENTITY gt      "&#62;">
  <!ENTITY amp      "&#38;#38;">
  <!ENTITY apos     "&#39;">
  <!ENTITY quot     "&#34;"> ]>
<CIM CIMVERSION="2.0" DTDVERSION="2.2">
<DECLARATION>
<DECLGROUP>

<DECLARATION>
  <DECLGROUP>
    <VALUE.OBJECT>
      <INSTANCE CLASSNAME="Header">
        <PROPERTY NAME="Date" TYPE="string">
          <VALUE>12/06/1999</VALUE>
        </PROPERTY>
        <PROPERTY NAME="Application" TYPE="string">
          <VALUE>NNM</VALUE>
        </PROPERTY>
      </INSTANCE>
    </VALUE.OBJECT>
    <VALUE.OBJECT>
      <INSTANCE CLASSNAME="PC">
        <PROPERTY NAME="ID" TYPE="string">
```

```

    <VALUE>1</VALUE>
  </PROPERTY>
  <PROPERTY NAME="Name" TYPE="string">
    <VALUE>Vectra</VALUE>
  </PROPERTY>
  <PROPERTY NAME="Price" TYPE="real64">
    <VALUE>3200.00</VALUE>
  </PROPERTY>
  <PROPERTY NAME="Brand" TYPE="string">
    <VALUE>Hewlett-Packard</VALUE>
  </PROPERTY>
  <PROPERTY.REFERENCE NAME="User">
    <VALUE.REFERENCE>
      <INSTANCENAME CLASSNAME="Employee">
        <KEYBINDING NAME="ID">
          <KEYVALUE>4</KEYVALUE>
        </KEYBINDING>
      </INSTANCENAME>
    </VALUE.REFERENCE>
  </PROPERTY.REFERENCE>
</INSTANCE>
</VALUE.OBJECT>
<VALUE.OBJECT>
  <INSTANCE CLASSNAME="Mouse">
    <PROPERTY NAME="Name" TYPE="string">
      <VALUE>Cordless Pro V</VALUE>
    </PROPERTY>
    <PROPERTY NAME="Brand" TYPE="string">
      <VALUE>Hewlett-Packard</VALUE>
    </PROPERTY>
    <PROPERTY.REFERENCE NAME="Parent">
      <VALUE.REFERENCE>

```

## Exporting Data

### Exporting Data

```
<INSTANCENAME CLASSNAME="PC" >  
  <KEYBINDING NAME="ID" >  
    <KEYVALUE>1</KEYVALUE>  
  </KEYBINDING>  
</INSTANCENAME>  
</VALUE.REFERENCE>  
</PROPERTY.REFERENCE>  
</INSTANCE>  
</VALUE.OBJECT>
```



## Viewing Data Exchange Files

Data exchange files can be viewed in a browser such as Internet Explorer. When the `View` button is selected the XML file is converted to HTML and structured into an object-tree format, making it easy to view. The viewer provides system administrators with a great tool for ensuring the extractor is configured correctly, and that the data exported is sufficient, prior to loading it into the Service Desk application database.

To view data exchange files prior to importing:

1. From the `Tools` menu, click `System`, then expand the `Data` folder in the Administrator Console. Double-click on the `Data Exchange` icon and then double-click the `Data Exchange Tasks` icon to open it.
2. Double-click on the task that corresponds with the exchange file you want to view. The task will be opened in the `Data Exchange` dialog box.
3. Enter the XML file you want to view in the `Exchange file` field, then click `View`:

**Figure 2-20** Example of XML file

CLASS	ATTRIBUTES	VALUES
Header	Date Application	9/11/2000 ITSM
CL_CODE_SER_STA	ID ORDERING TEXT	1 10 Being tested / evaluating
CL_CODE_SER_STA	ID ORDERING TEXT	2 20 In production
CL_CODE_SER_STA	ID ORDERING TEXT	3 30 Cancelled
CL_SERVICE	ID POOL_SEARCHTEXT SRV_ID STATUS_SEARCHTEXT DESCRIPTION NAME	4 SMS Pool 100 In production The e-mail functionality E-mail
CL_SERVICE_CI	ID CI_ID Parent	5 300682 CL_SERVICE

4. When you finish viewing the file you can import it or run the export process again before importing.

---

**NOTE**

Viewing large XML files with the Data Exchange Viewer is not recommended. A normal text editor is recommended when viewing larger XML files.

---

---

## **3** **Import Mapping**

Import mapping is done via a series of mapping screens in the Service Desk application. Classes from the external application must be mapped to Service Desk items and provided with an appropriate template in Service Desk. The template selected comes with a number of default

attributes and values. Every item in Service Desk has a template which can be used for import mapping. Service Desk can be customized, making it possible to create new fields in Service Desk to support additional value mapping.

It can be helpful to use the viewer provided to look at the exported XML file when you are conducting the mapping process. This will give you an overview of all of the items you need to map. It can also help prevent minor deviations in terminology and spelling, that could lead to data not being imported.

The *HP OpenView Service Desk:Data Dictionary* contains helpful information about the structure of the Service Desk application. It is available as an HTML file on the Service Desk 4.0 CD-ROM with the file name `Data_Dictionary.htm`.

---

**NOTE**

External classes and attributes entered when configuring the import mapping must match exactly with those present in the XML file. If the item is in uppercase instead of lowercase letters, extra spaces are present, or the spelling differs a warning will be logged in the import log file during the import process.

---

It is possible to install a number of predefined example import mappings and configurable extractor files when you install Integrations for Service Desk. These examples can be changed to fit your organizational needs. The appendices in this guide provide additional information on configuring a number of the example integrations.

## About Item Mapping

Every external class imported needs to be mapped to an item in Service Desk. Templates exist for every item in Service Desk. Templates provide a way of relating an external class to a Service Desk item and category. Templates contain default attribute values. You can create one or more import mapping templates for each external application using data exchange.

If PC inventory information is being imported you might search for a template for configuration items, selecting a template that comes closest to matching the type of data you will import. There may be a template specifically for PCs available. The template will contain a number of default attributes, and values. You will map the exported properties and values for the PC to those for the configuration item in Service Desk. The mapped data overwrites any information in the default fields of the template when it is imported. Classes and their associated properties and values must be mapped in order to be imported into Service Desk. The class name between [ ] brackets in the configurable extractor file is the class you need to map in the import mapping.

## Attribute Mapping

Attributes define and identify items, for example the exported class PC might have the properties Model and Location, Service Desk may have similar attributes called Type and Address. For those properties to be imported, they must be mapped to attributes that belong to a Service Desk configuration item so that the system knows where to put them when they are imported. ATT and PARENT\_RELATION\_NAME in the configurable extractor determine the attributes that have to be mapped when configuring the import mapping. The term properties has the same meaning as the term attributes used in Service Desk.

Import mapping for a persons gender is done by using 0 for a male and 1 for a female. The attribute name ID is reserved for internal use by the Service Desk application.

## Key Binding

Key binding is used to identify specific external classes, or items in Service Desk. An example is a serial code assigned to a printer, an employee number or even the property PC.Hostname. By setting the key

binding for an attribute you can identify similar items and recognize changes. At least one attribute in every class must be used for key binding, the `Name` property is set as a default key value in the import mapping examples. Select the `This field is used for key binding` check box in the `Field Mapping` dialog box to make that attribute a unique key.

### **Value Mapping**

Many attributes come with values that further define them. Some attributes contain values in a code table, meaning that there is a pre-defined set of possible values to choose from. Other attributes have values that cannot be predefined, such as serial numbers or employee names. When you map an attribute that has values from a code table, the value mapping button will become available in the `Field Mapping` dialog box. You will then be able to map the external property values to the attribute values that exist in the code table in `Service Desk`.

### **Required Fields**

When importing items into `Service Desk` keep in mind that required fields might exist for that item, for example the `Name` field might be required if you are importing a personnel record. To view required fields for each item; from the `Tools` menu click `System`, and then expand the `Security` directory from within the `Administrator Console`. Click `Prevention` and then `Required Fields`. More detailed information about using that dialog box can be found in the online help.

### **Defining Relations**

Items are often related to other items. For example, a PC may come with a CD-ROM and have a monitor, a mouse and a keyboard connected to it. It probably has a number of software programs running on it and is connected to a network with a specific IP address. The relation between these items is hierarchical with the PC as the parent, and the components being related to it the children. In a database this structure is stored in the form of an object tree, with the components forming branches or nodes in the tree. The nodes in the tree are named according to their position. The node above another node is a parent while the node immediately under it is the child. The node at the top of the object tree is called the root and does not have a parent. In this example the PC or item is the root of the object tree.

How relations are imported into `Service Desk` is determined by how they

were extracted and how the import mapping is done. Import mapping for relations is done by using default search attributes from Service Desk that belong to each entity. Ensure that you use search codes that exist in Service Desk. The extractor is designed to convert information from the external data source in a consistent way, maintaining the object tree structure. The API matches parent and child relationships of items being imported with that of the items already present in Service Desk and imports the data.

For additional information about configuring the extractor to export relations, see “Defining Relations in the Configuration File” on page 72. For an example see “Importing Data With Relations” on page 245.

The Data Dictionary provided with Service Desk provides a description for all items and tables in Service Desk to include the relations between those items. For example, if the attribute for an item has a 1:1 relation, e.g. Service call caller to the Person item, the table where the other item is stored will be listed. If the attribute has a 1:N relation, for example Service call History lines to History line items, the table where the items are stored will be shown along with the table column that contains the foreign key.

There are three primary types of relations that can be imported; N:1 relations based on a search key, N:N or Parent-Child relations, and 1:Rel:1 that involve multiple classes. The following sections will explain the relation types in more detail and include information on what you can do in the import mapping to import them correctly.

---

**CAUTION**

---

Old objects and relations are not automatically deleted in Service Desk. You must manually locate and delete old objects and relations to remove them.

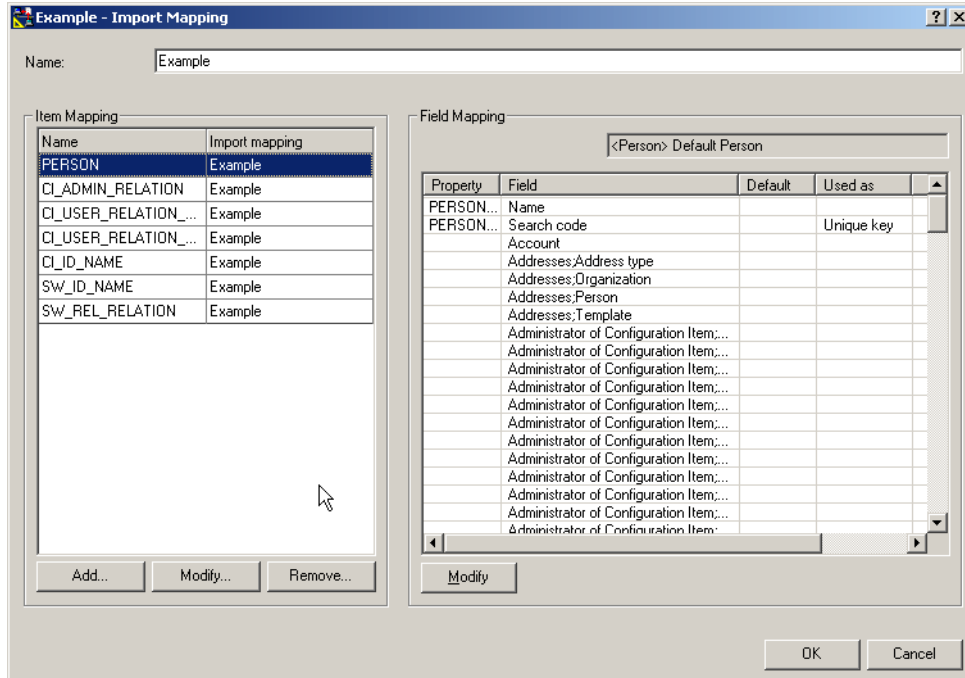
### **Search Key; N:1 Relations**

In the N:1 relation the N refers to one or more items in Service Desk related to one (1) other item in Service Desk. This type of relation uses a search code as an entity reference to determine the relationship between the items. The item that is referenced using a search code should be listed first in the exported XML file, the order in the XML file is the order it will be imported in. The order is determined by how the classes are listed in the [CLASSES] section in the configuration file.

Import Mapping  
About Item Mapping

To complete the process for importing this relation you will need to create the import mapping to use a search code as a reference to the item. For example:

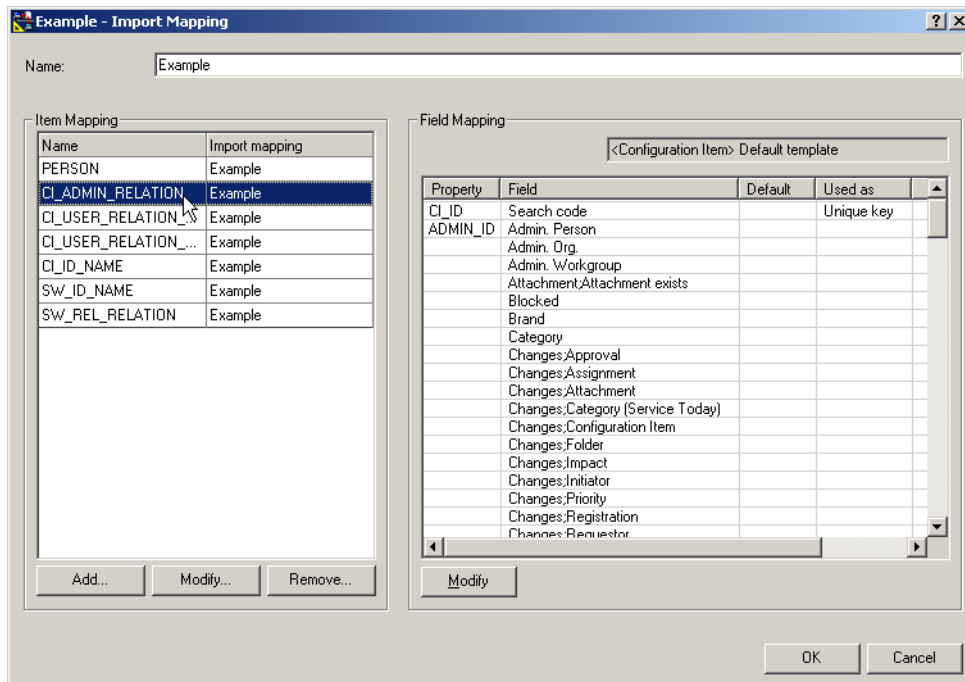
**Figure 3-1** Import Mapping for PERSON Class



Import Mapping then needs to be done for the CI\_ADMIN\_RELATION Class:

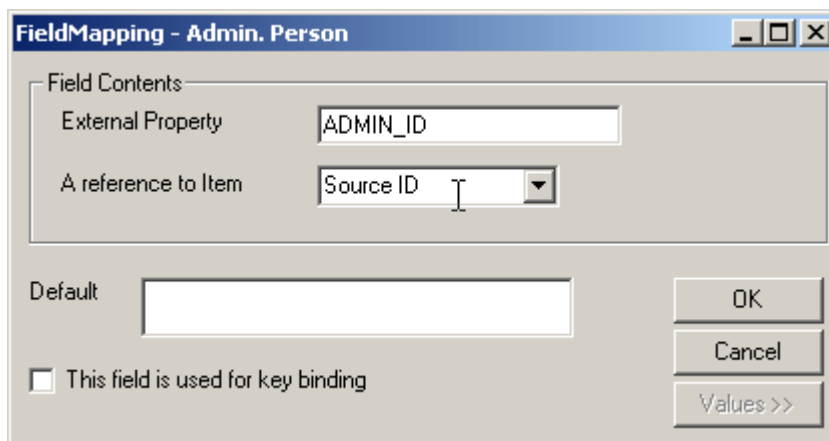


**Figure 3-2** Import Mapping for CI\_ADMIN\_RELATION Class



Next you need to make a reference code from one item to the other as is done in the following Field Mapping dialog box:

**Figure 3-3** Field Mapping for ADMIN\_ID



[Import Mapping](#)  
[About Item Mapping](#)

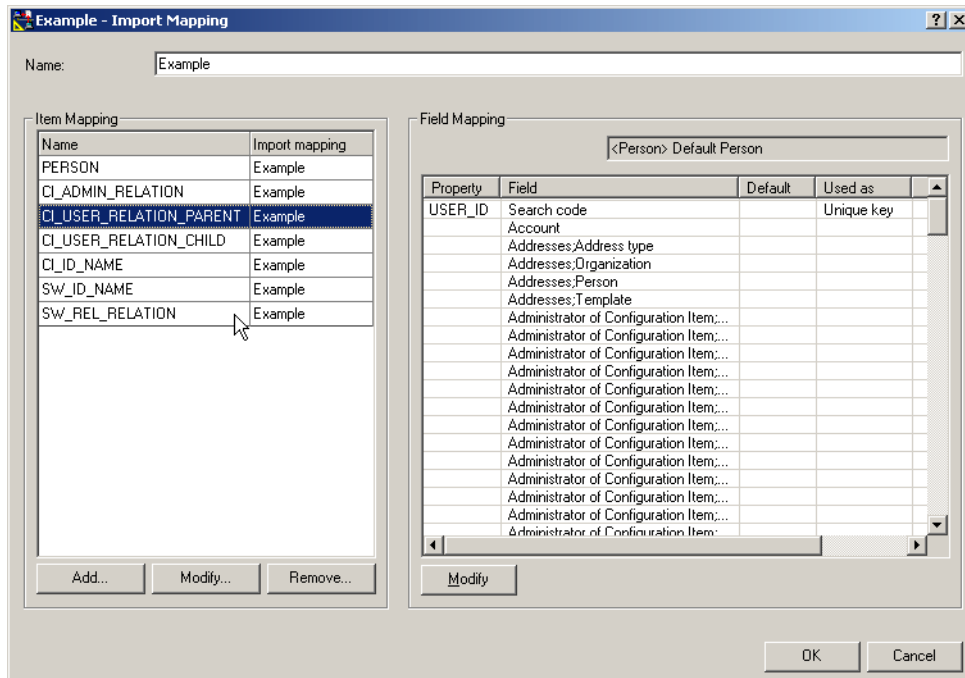
For an additional example see “Importing Data With Relations” on page 245.

**Parent-Child; N:N Relations**

In this type of relation the N refers to one or more items in Service Desk related to N, one or more other items in Service Desk. This type of relation uses multiple item search codes to make the relationship. The configuration file is modified differently for a N:N relation than a N:1 relation. For the N:N relation you can use PARENT, PARENT\_RELATION, and PARENT\_RELATION\_NAME fields to specify the Parent-Child relation.

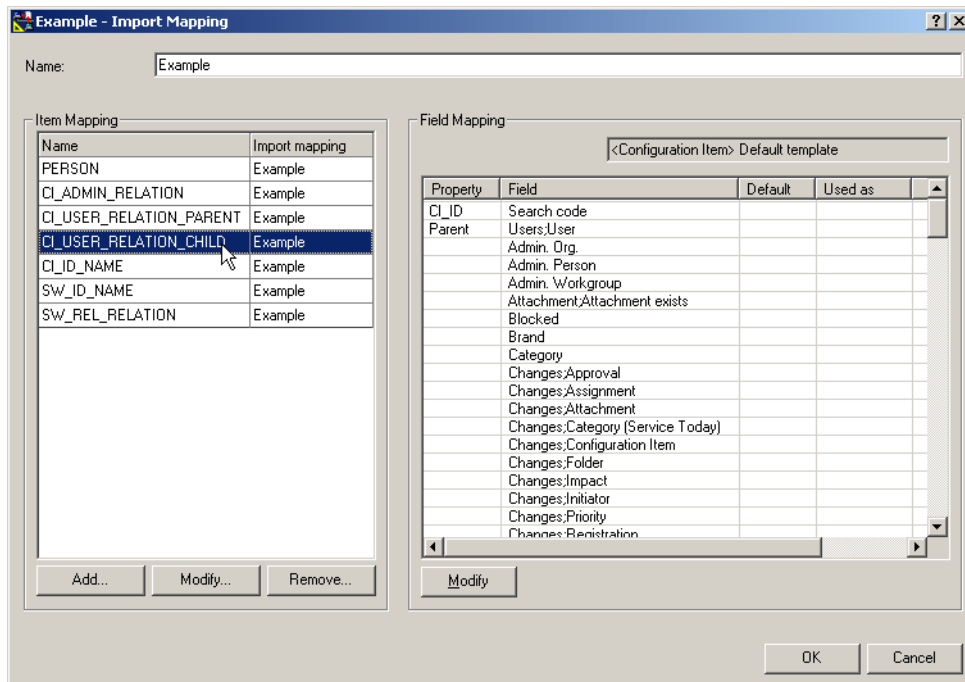
To import this relation you will need to create entity references in the import mapping as follows:

**Figure 3-4 Import Mapping for CI\_USER\_RELATION\_PARENT Class**



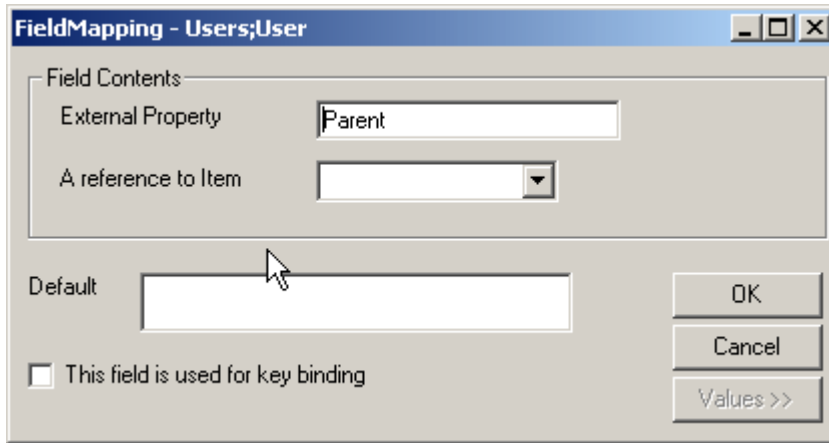
Next you will need to create the import mapping for the child relation:

**Figure 3-5** Import Mapping for CI\_USER\_RELATION\_CHILD



Next you will need to complete the mapping for the parent-child relation as in the following Field Mapping dialog box:

**Figure 3-6**                    **Field Mapping for Parent**

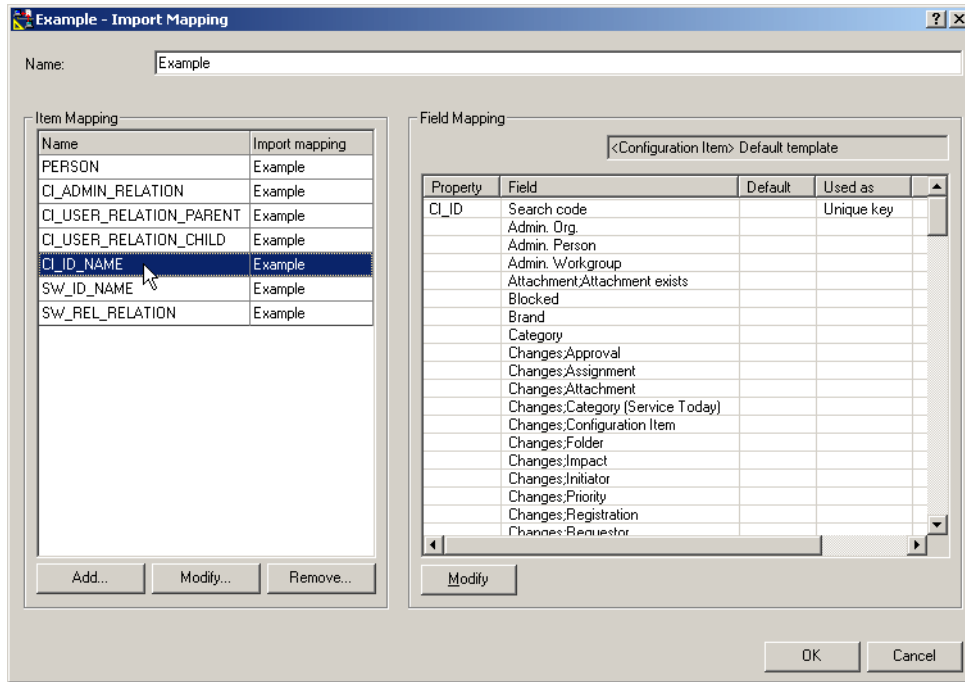


### **Relations with Relation Type; 1:Rel:1 Relations**

The 1:Rel:1 type relation is a relation that is specified by creating another class for the purpose of describing the relation between two or more classes. You can also use an entity reference with a search code and an entity reference to a relation table in Service Desk to describe the relation. When using a reference to a relation table, the relation is maintained in a separate relation code table. You can map items to the values in the relation table to define the type of relation. Another option is to enter a default value or create value mapping for the relations.

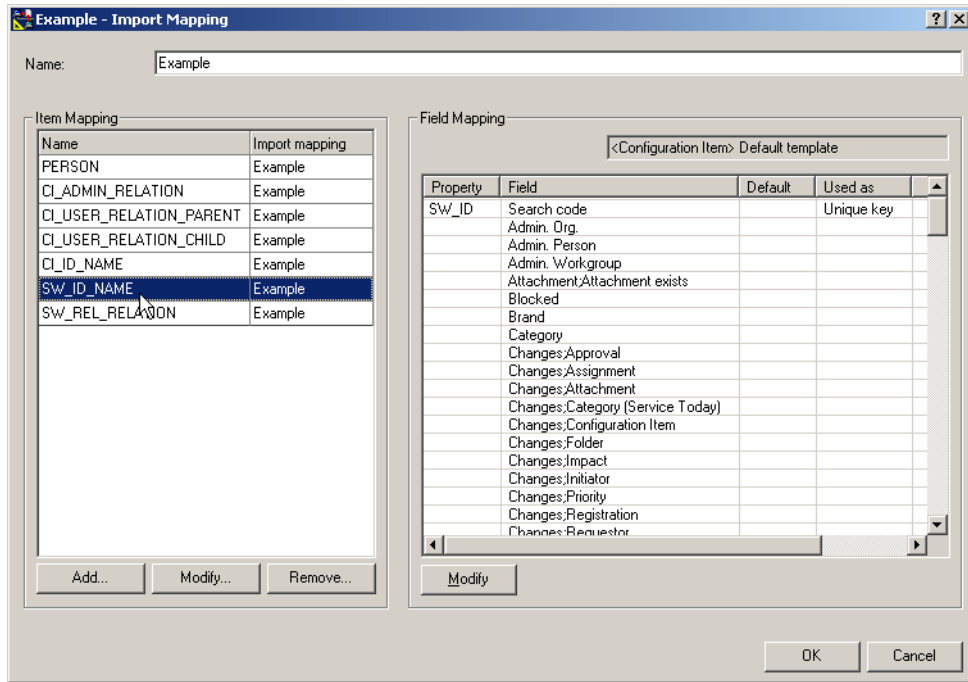
The following dialog boxes show an example of the import mapping for this type of relation. This dialog box shown the import mapping for one class:

**Figure 3-7** Import Mapping for CI\_ID\_NAME Class



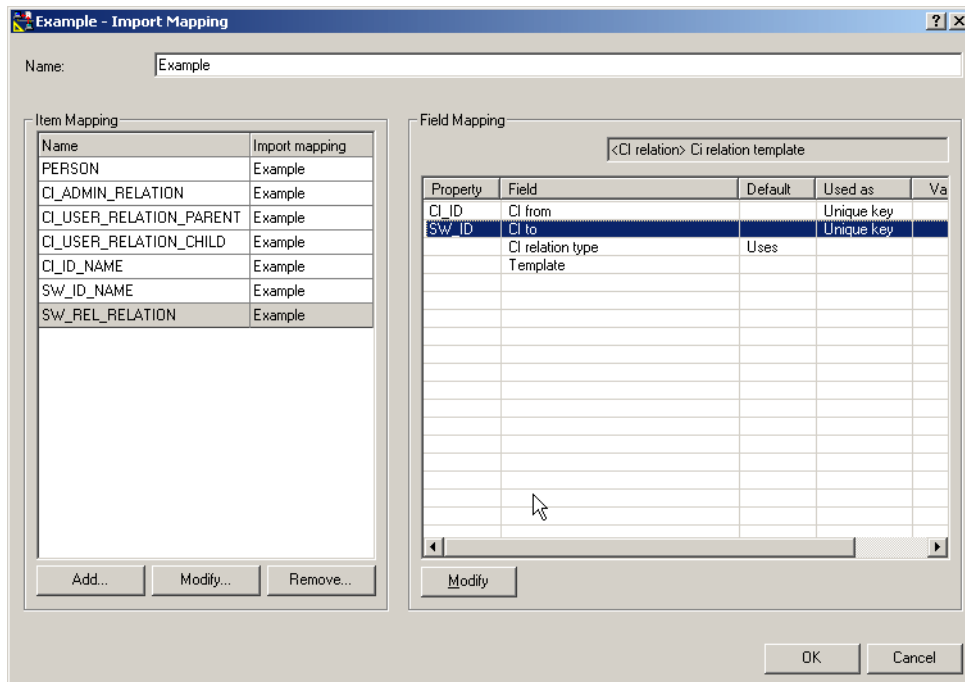
The import mapping for another class is shown in the following dialog box:

**Figure 3-8** Import Mapping for SW\_ID\_NAME Class



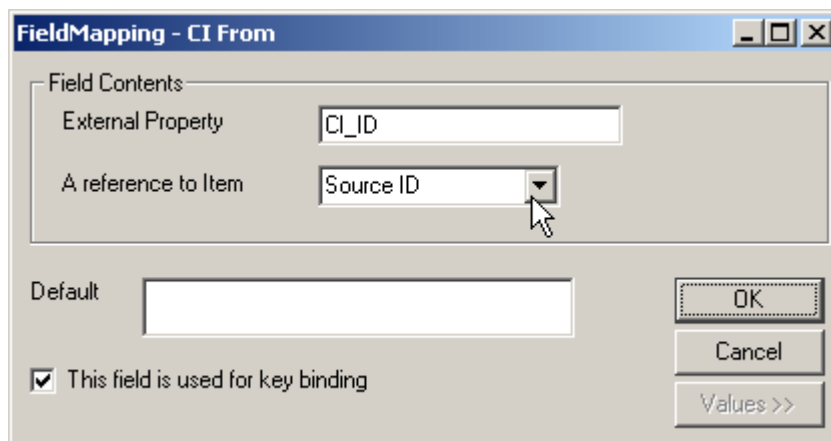
The following class is imported to define the relation:

**Figure 3-9** Import Mapping for SW\_REL\_RELATION

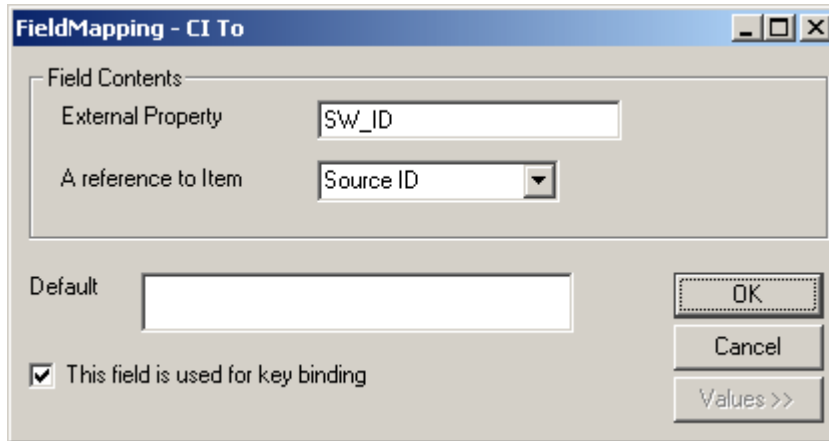


Two entity set references are made to link the classes as shown in the following dialog boxes:

**Figure 3-10** Field Mapping for CI\_ID



**Figure 3-11**      **Field Mapping for SW\_ID**



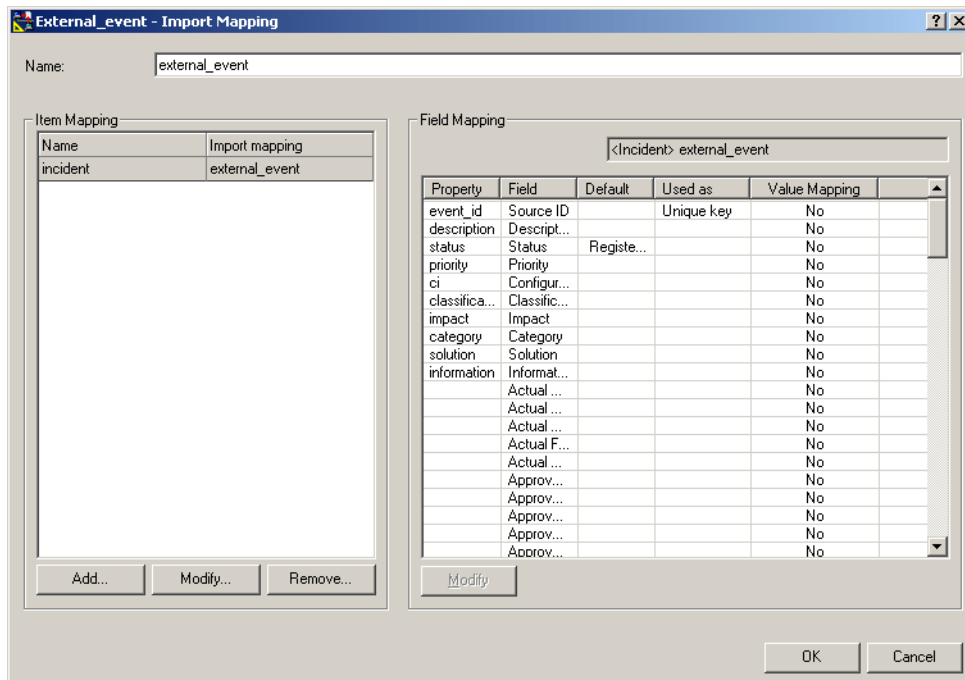


## Mapping Service Events

The same import mapping in Service Desk is used for service events. You can use import mapping settings created for importing batches of data, as long as the class you are importing and the import mapping name are the same. Another option is to set the import mapping especially for the service event you will be importing.

The import mapping Name field(-x switch) and the Class name (-c switch) must be included in the command line for the service event. For more information see “Importing Service Events” on page 149.

**Figure 3-12** Import Mapping dialog box for sd\_event



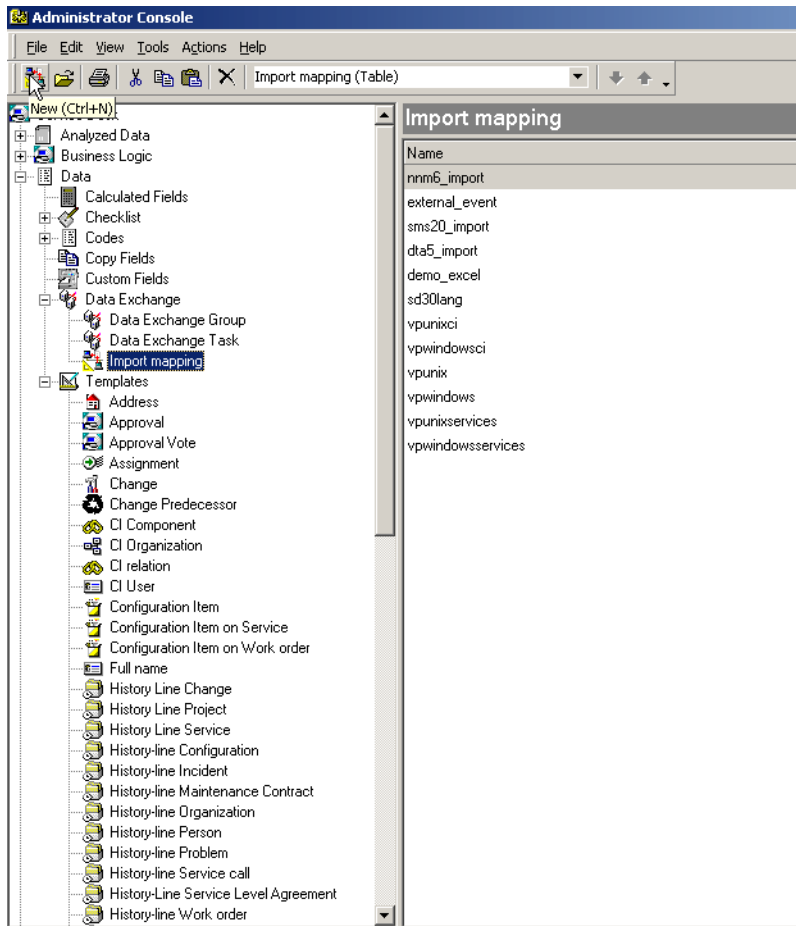
## Mapping Classes and Attributes

Import mapping can be done from the application server or a client computer. You can create a new import mapping or use the examples we provide. The available examples will be visible in the import mapping window.

### Creating a New Import Mapping

- Step 1.** Navigate to the Import mapping folder:
1. From the **Tools** menu, click **System**, then **Data**, then **Data Exchange**, and then **Import Mapping**:

**Figure 3-13** Import Mapping Window



- Step 2.** To open a New Import Mapping dialog box, right-click anywhere in the import mapping window. Select New Import Mapping from the menu that appears.
- Step 3.** Provide a name for the import mapping to identify which external application it is used for. It is recommended that you use a similar name for the import mapping, configuration file, and integration account.
- Step 4.** Map classes, properties and values to Service Desk items, attributes and values. To continue with the mapping process follow the instructions in Step 2, “To add or change a class mapping:” on page 112.

---

**NOTE**

To prevent errors when importing CIs you must map the `searchcode` field in Service Desk to a unique property when you do your import mapping. Check the import mapping you are using to verify that the `searchcode` field is mapped to a unique field property.

The reason for this is that whenever you create or edit a configuration item, Service Desk validates the search code entered and prevents you from saving the CI if the search code is already in use by another CI. If a non-unique default `searchcode` is entered in the template used for import mapping an error will result when you try to import more than one CI. The unique search code option is set in the General Settings dialog box. To view the settings, open the Administrators Console, then System Panel, and open the General Settings dialog box.

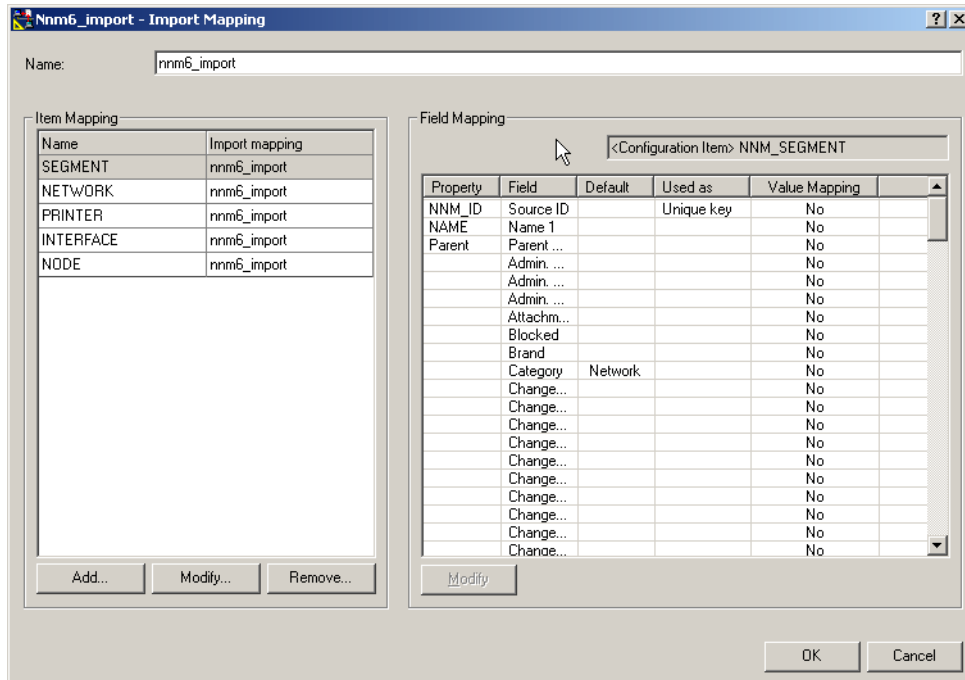
---

## Modifying an Import Mapping

**Step 1.** Open an existing import mapping:

1. Double-click the `Data Exchange` folder and then double-click `Import Mapping`. All import mappings will be displayed in the Import mapping window. Double-click one of the import mappings and the Import Mapping dialog box will appear:

**Figure 3-14** Import Mapping dialog box



2. In the Item Mapping portion of the dialog box select the class you want to modify and click `Modify` to change the class selected.

The Field Mapping portion of the dialog box will show the attribute mapping for that item:

- The `Property` column shows external properties as they should appear in the XML file.
- The `Field` column shows Service Desk attributes.
- The `Default` column shows the default values from the template the class is mapped to.

---

**NOTE**

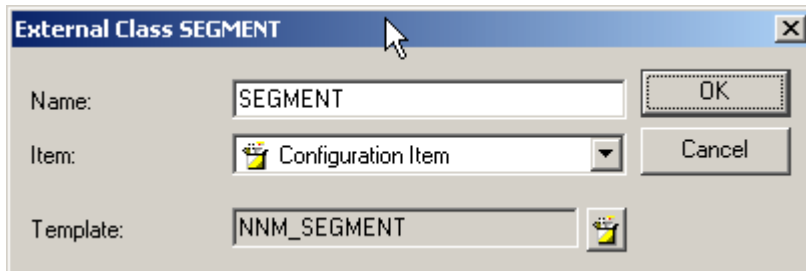
Only editable fields will be shown in the import mapping dialog box. Read-only fields will not be displayed.

---

**Step 2.** To add or change a class mapping:

1. To add a class to your import mapping click **Add** in the Import Mapping dialog box. The New External Class dialog box will appear, skip to bullet 3 for instructions on what to enter in the fields.
2. To modify a mapped class, select the class in the Item Mapping window and click **Modify**. This will open the class in the External Class dialog box.
3. In the External Class dialog box, the **Name** field must contain the name of the external class you want to import. The class name must be entered exactly as it appears in the XML file.
4. Use the drop-down arrow in the **Item** field to select the item you want to map the external class to in Service Desk, this step helps in categorizing the class and narrows the number of templates you will need to choose from:

**Figure 3-15 External Class dialog box**

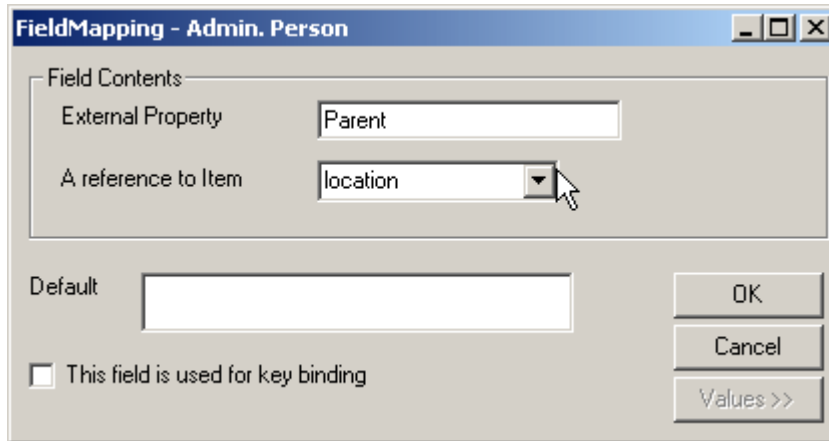


5. Click the **Template** button to select a template from a list of available templates. Every imported class must be mapped to a template. The template will provide default values for the imported class.
6. Click **OK** when done to save your mapping. Repeat this step for each class mapping you want to modify or add.

**Step 3.** Modify mapped attributes:

1. From the main Import Mapping dialog box, select the class the attributes belong to and then double-click the attribute in the **Field** column to map an external property to it. The Field Mapping dialog box will appear:

**Figure 3-16** Field Mapping dialog box



2. In the `External Property` field, enter the name of the external property you want to map to a Service Desk attribute.
3. You can use the `A reference to Item` field to create an additional reference to another item. For example, you can map the External property `OWNER` to the attribute `Person` and then reference additional attributes associated with `Person`, for example `address` or `e-mail`.
4. Attribute defaults come from the template that is used. To add a default you will need to open the template and modify it. The default attribute values will show up in the `Default` column in the main Import Mapping dialog box. If a property does not contain a mapped value the default will be used in Service Desk.
5. Select the `This field is used for key binding` check box if the external property contains a unique key value. A class must have at least one property marked for key binding.
6. Click `OK` when you are finished. If a value list is available for the attribute the `Value` button will be activated, see the following step for more information on mapping values.

**Step 4.** Map property values to attribute values:

1. If a value list exists for an attribute selected in the `Field` column, the `Value` button will become active when you are in the Field Mapping dialog box. Click `Value` and the Field mapping dialog box will be

extended, to include value mapping:

**Figure 3-17 Expanded Field Mapping dialog box**

Value for External Attribute	Value for Internal Attribute
New	Installed
Ordered	Ordered
Benchstock	Stock
Maint	Maintenance

2. In the bottom left field, enter the value from the external property that you want to map.
3. In the bottom right field, enter the Service Desk value. A drop-down arrow is available for searching for available values.
4. Use the Add To List button after mapping a value. When you have



mapped all of the values for the attribute, click **OK** to return to the External Class dialog box.

5. When you are done with the import mapping, select **OK** in the Import Mapping dialog box to save your work. If you do not perform this step your mapping will *not* be saved.

---

**CAUTION**

If you change the import mapping of a class that was already imported into Service Desk, and then import that class to a different location in Service Desk, your data may *not* be imported correctly.

---

Import Mapping  
**Mapping Classes and Attributes**

---

## **4** **Importing Data in Batches**

Service Desk provides you with the flexibility of importing data in batches or importing service events via a command line. This chapter explains the batch import process, chapter 5 “Importing Service Events” on page 149 explains the process for importing service events.

You need to use a configurable extractor for exporting, and an import mapping for importing. A number of example configuration files and import mapping files can be accessed from the Data Exchange dialog box after installing the Integrations tools and the demo database from the Service Desk CD-ROM.

The data exchange software will organize the exported data into a file for importing. The import mapping is applied and the data is arranged so that it can be quickly imported into the database. If data exists that is not mapped, that data will not be imported. The data file is then sent through the server designated for data exchange and the Service Desk database is updated.

---

**NOTE**

Old objects and relations are not removed in this version of Service Desk. If you change the unique key(s) of a record Data Exchange will consider it a new item for import. If you do not change the unique key(s), Data Exchange will update the record and overwrite the existing value.

You must manually delete old objects and relations to remove them. See “Removing Redundant Items and Relations” on page 131 for more information.

---

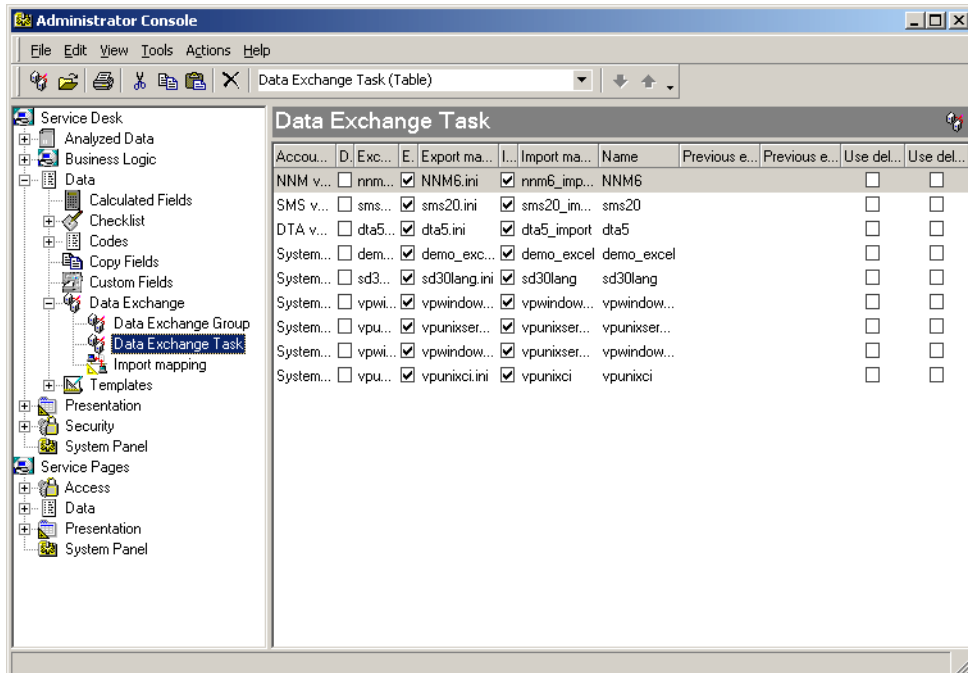
## Using a Task to Import Data

The data exchange process for batch importing is run from the Data Exchange dialog box in the Administrator Console.

**Step 1.** Open a data exchange task.

1. From the **Tools** menu, click **System**, then expand the **Data** folder from within the Administrator Console. Double-click the **Data Exchange** folder, then **Data Exchange Task**.
2. Double-click the **Data Exchange Tasks** icon. Existing data exchange tasks will be visible in the window.

**Figure 4-1** Data Exchange Task Window



3. Double-click the task you want to use to import data. The Data Exchange dialog box will open.

**Step 2.** Configure the task to import data.

Importing Data in Batches  
Using a Task to Import Data

1. Enter the name of the XML file you want to import in the Exchange file field:

**Figure 4-2 Data Exchange dialog box - Importing Data**

Export data from a storage device

Select a configuration file for extracting data:

**Export mapping:** dta5.ini

Edit...

Export data to or import data from file

**Exchange file:** dta.xml

Browse... View...

Import data into Service Desk database

Select import settings

**Account:** DTA 5

Password: xxxxxxxx

**Import mapping:** dta

Debug New... Modify... Remove

Use delta processing

**Previous excha...** dta\_200101101644..xml

Browse... View...

Save... List Log Files... OK Cancel

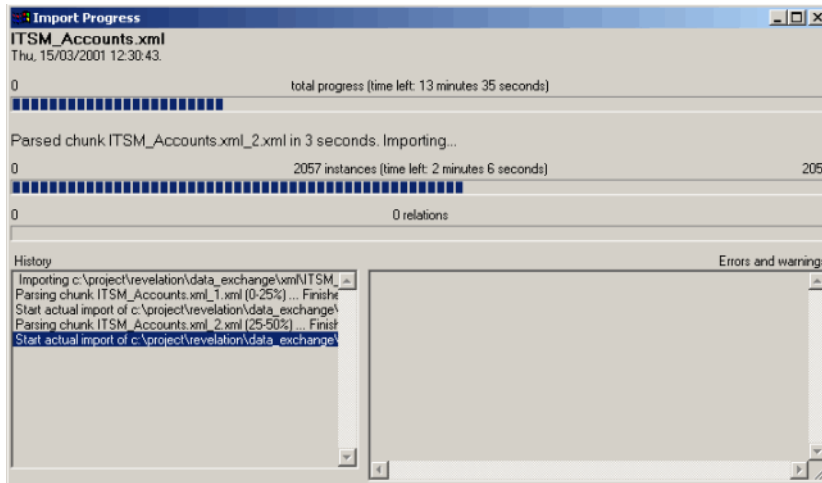
2. Select the Import data into the Service Desk database

- check box.
3. Enter the `Account` established for importing data for this integration and the password for that account. Use the `Account` drop-down arrow to search for the correct account. For more information see “Integration Accounts” on page 147.
  4. In the `Import Mapping` field use the drop-down arrow to find the import mapping you want to use during this data exchange. See “Mapping Classes and Attributes” on page 108 for more information about import mapping.
  5. Select the `Debug` check box to create a detailed log file while importing.
  6. Place a check in the `Use delta processing` check box. It is located in the lower half of the dialog box. See “Delta Processing” on page 131 for more information about delta processing.
  7. If you are using Delta processing, use the `Previous Exchange` field to enter the name of the previous data exchange xml file you will use to compare with the current data exchange xml file. A `Browse` button is available below the field, for locating the file. A date and time will be automatically added to your data exchange xml file during the data exchange process, making it easier to identify the previous Data Exchange file. The date format is `yyyy/mm/dd/HH:mm`, for example `DTA_200101101644.xml` refers to 2001, January, 10th, 16:44.

**Step 3.** Import the data.

1. Click `List log files` to get a list of the log files available for viewing. To select a file for viewing, right-click on the file and the select `Open`. Log files can be opened at any time during the data exchange process.
2. Click `Start` to run the data exchange processes selected or `OK` to save the task and run it at another time. A progress monitor will appear showing the export progress and then the import progress. When the export process is complete the sentence “EXTRACTOR finished” will appear in the log file. When the import process is finished the line “Finished loading relations at...” will be shown in the log file:

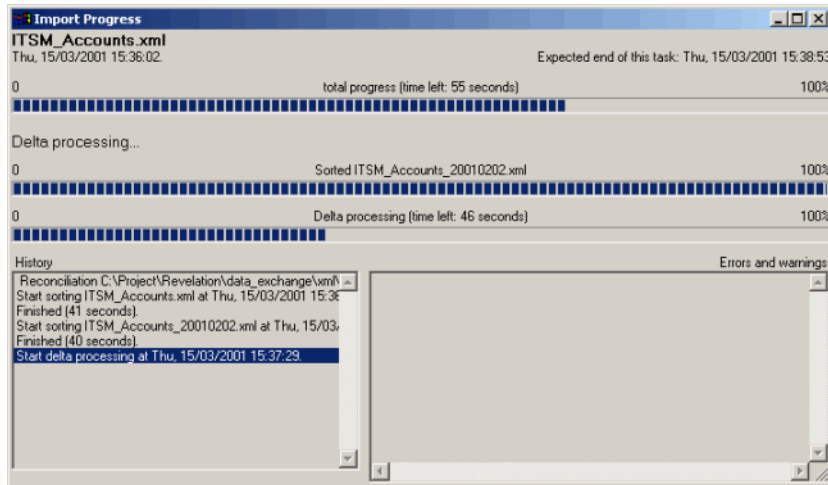
**Figure 4-3** Import Progress Monitor



If you selected the Delta processing option, your data import will take place in a series of steps. First the exchange file and the reconciliation file will be sorted. The sorting progress will be shown in a progress bar. Next, the exchange file and the reconciliation file will be compared to analyze the differences. The progress of the Delta-processing will be visible on screen at that time. Afterwards, new and updated items (entities) and relations will be imported into Service Desk:



**Figure 4-4** Import Progress Monitor with Delta Processing



## Creating Data Exchange Tasks

Once a task is created and saved you can open it from the Data Exchange folder to run it, instead of entering information in the Data Exchange dialog box again. You can create multiple tasks for exporting, importing, or both exporting and importing data. To create a task:

1. From the **Tools** menu, click **System** and then double-click the **Data** icon in the **Administrator Console**.
2. Double-click the **Data Exchange Tasks** icon.
3. Right-click in the **Data Exchange Tasks** dialog box and click **New Data Exchange Task** from the menu that appears.
4. Fill in the fields in the **New-Data Exchange Task** dialog box. For additional information on filling in these fields see “Exporting Data” on page 82 or “Using a Task to Import Data” on page 119.
5. Click **OK** to save the task and **Start** if you want to run the task. The task will be visible in the **Data Exchange Task** window with the options you have set.

---

### NOTE

You can only configure tasks from the application server and *not* from a client computer.

---

## Executing a Task

After creating and saving Data Exchange tasks you can quickly execute them. To execute an existing task:

1. From the **Tools** menu, click **System** and then double-click the **Data** icon in the **Administrator Console**.
2. Click **Data Exchange Tasks**.
3. Double-click a task in the **Data Exchange Task** window to open it.
4. If you agree with the information entered for the task, click **OK** at the bottom of the **Data Exchange** dialog box to execute it.

## Scheduling a Data Exchange Task

Data Exchange tasks can be scheduled by using the NT scheduler available with Windows NT. The Schedule service must be running when you want the task performed. Insert the `at` command into the command line used for the data exchange task. For example, the `at` command can be added to the command line syntax for exporting data as follows:

```
sd_exchange export <configfile> <exportlog> <xml file>
at[\\<computername><time>[/interactive][/every:<date>[,...]]
/next:<date>[,...]]
```

The following table explains the parameters used with the `at` command example shown above. For more information refer to the online help for Windows NT:

Parameter	Comment
\\computername	Use to specify a remote computer. Leave out if you want to schedule the command on a local computer.
time	Use to specify the time when the command should be run. Enter the time in 24 hour clock format, for example 22:30.
/interactive	Allows the scheduled process to use the desktop of the user who is logged on.
/every:date[,...]	Enter the day of the week (M,T,W,Th, F,S,SU), or one or more days in the month (1-31). Command will be run every time the day specified occurs.
/next/date[,...]	Same as above except the command will only be run on the next occurrence of the day specified.

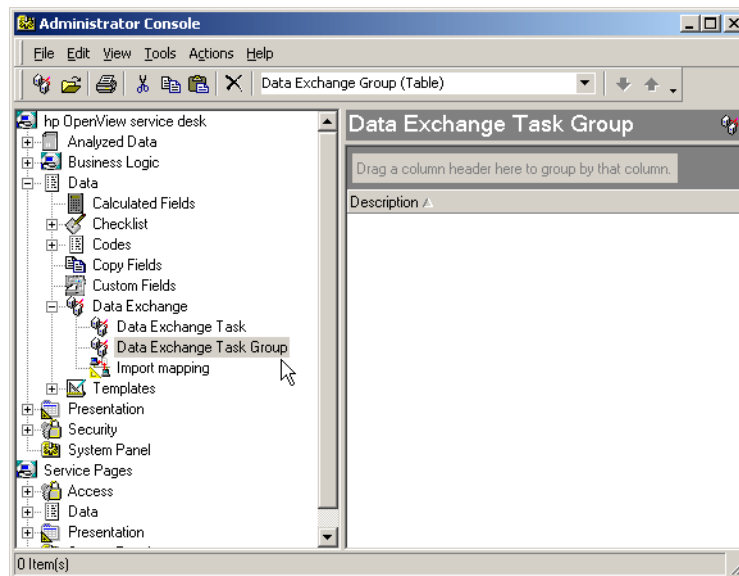
## Creating a Data Exchange Task Group

You can relate multiple Data Exchange tasks and execute them as one Data Exchange Task Group. After creating individual tasks in Data Exchange you can relate them and execute them, in order, with one Start command. The Task Group is not designed to run from a command line.

To create a Data Exchange Task Group:

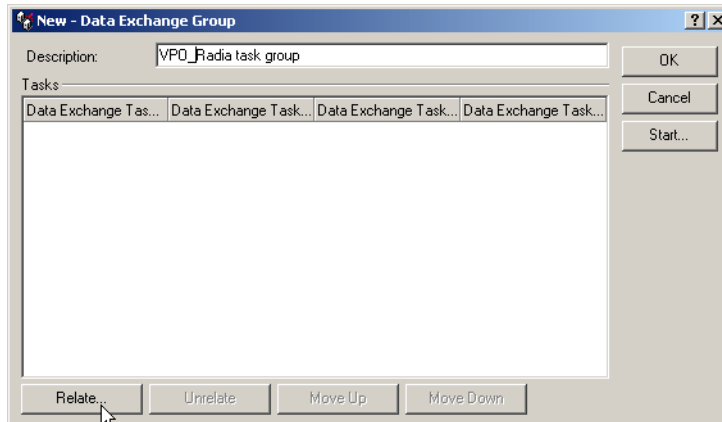
1. From the **Tools** menu select **System**, open the **Data** folder in the **Administrator Console** and double-click the **Data Exchange Task Group** icon:

**Figure 4-5** Data Exchange Task Group



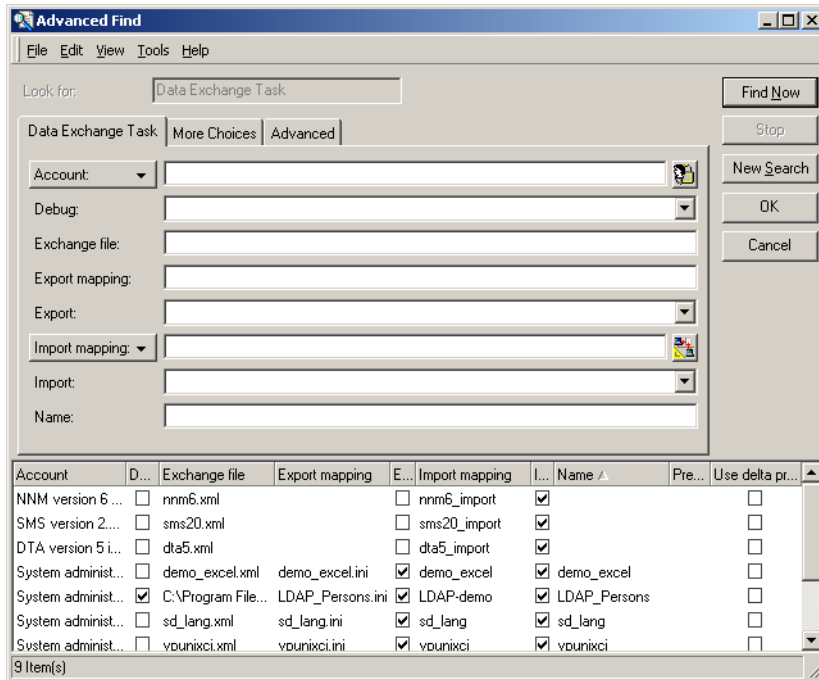
2. Right-click in the **Data Exchange Task Group** window and select **New Data Exchange Task Group** from the menu that appears.
3. Enter a description to identify this Task Group:

**Figure 4-6**      **Relate Tasks**



4. Click **Relate** to locate and add tasks to the task group.
5. In the **Advanced Find** dialog box that appears select the **Data Exchange Task** tab and click **Find Now** to locate all **Data Exchange** tasks. You can also use the available fields to define the search criteria:

**Figure 4-7 Search for Tasks**

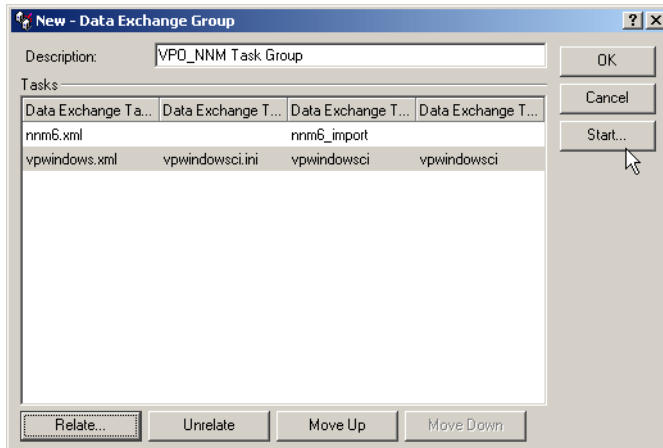


6. The Advanced Find dialog box will be extended to show the tasks that meet your search criteria. Select the task you want to add to your Task Group and click **OK**. Repeat this process until you have added all of the tasks you want in this Task Group.

If you want to change the parameters that are set for a task, you will need to open the task individually in the Data Exchange Task Group window to make the change, for example if you want to change the Debug option from Yes to No.

7. Tasks will be executed in the order they are shown in the Data Exchange Task Group dialog box. Use the **Move Up** or **Move Down** buttons to change the order.
8. Click **OK** to save the Task Group:

**Figure 4-8** Example Group Task



9. Click **Start** to run all tasks in the Task Group.

The authorizations for all tasks in your Task Group will be checked when you start the Task Group. You will be prompted for account passwords at this time. If the related tasks use different accounts you will be prompted for the account password of each different account.

10. To remove a task from your task group, select the task in the group window and click **Unrelate**, then **OK** to save the change.

---

## Reconciling your Data

The Reconciliation feature in Data Exchange minimizes the amount of data imported into Service Desk by filtering out redundant items. A reconciliation change list is created that identifies obsolete objects that may require manual removal from Service Desk.

The Reconciliation process filters out redundant information so that only the changes in your data are imported, thus reducing the load on your resources. This is accomplished by comparing the most recently exported data exchange XML file with the data exchange XML file currently exported. The differences between the two files are then recorded into an additional XML file for importing into Service Desk. To use the reconciliation feature you must have a valid Data Exchange xml file for comparison purposes. During the import process the current xml file will be compared to the older xml file.

The following tables explain which items and relations will be imported:

**Table 4-1**

**Decision Matrix for Importing Items**

<b>Item in previous exchange file?</b>	<b>Item in current exchange file?</b>	<b>Changes to the item's attributes?</b>	<b>How is item processed?</b>
No	Yes	Not Applicable	Import item
Yes	No	Not Applicable	Remove item manually
Yes	Yes	No	No action taken
Yes	Yes	Yes	Import item



**Table 4-2 Decision Matrix for Importing Relations**

<b>Item in previous exchange file?</b>	<b>Item in current exchange file?</b>	<b>Is the relation new or changed?</b>	<b>Has the relation been removed?</b>	<b>How will relation be processed.</b>
No	Yes	Yes	Not Applicable	Import relation
Yes	No	Not Applicable	Yes	Remove relation manually
Yes	Yes	No	No	No action taken
Yes	Yes	Yes	No	Import relation
Yes	Yes	No	Yes	Remove relation manually

### **Delta Processing**

The option to import your reconciled data is called delta processing and is set in the Data Exchange dialog box. When this option is selected, new items and relations will automatically be imported. Modified items and relations will automatically be updated. Redundant (obsolete) items and relations will need to be removed manually, See “Removing Redundant Items and Relations” on page 131.

To set the Delta processing option in the Data Exchange dialog box see “Using a Task to Import Data” on page 119.

### **Removing Redundant Items and Relations**

During the reconciliation process a change log is created recording all new, modified and redundant items and relations. To complete the reconciliation process in Service Desk you will need to remove redundant items and relations manually, see Table 4-2, “Decision Matrix for Importing Relations,” on page 131, and Table 4-1, “Decision Matrix for Importing Items,” on page 130. By using the change log you can quickly

find the items and relations that have become obsolete. Every item is recorded with key fields and values, making it possible to locate those items quickly in Service Desk.

The change log is created in the `data_exchange/log` directory with the current date and time. For example, `DTA_Changes_200101101644.log`. In the following example the `REDUNDANT ENTITIES` and `REDUNDANT RELATIONS` sections contain information that will help you locate those items that you will probably want to manually removal from Service Desk:

**Example 4-1**      **Change Log File**

```
-----  
ChangeList of delta processing on files  
e:\a.xml (new situation) and  
e:\b.xml (old situation).  
Date of import: Wed, 10/01/2001 16:44:19  
-----  
NEW ENTITIES  
=====
```

CL_CI with ID 282 with key-values:
CI_ID : 1000000677

```
UPDATED ENTITIES  
=====
```

CL_CI with ID 375 with key-values:
CI_ID : 1000000887
CL_CI with ID 676 with key-values:
CI_ID : 1000001347
CL_CI with ID 911 with key-values:
CI_ID : 1000001860

```
REDUNDANT ENTITIES  
=====
```

```
CL_CI with ID 105 with key-values:
  CI_ID : 1000000945
NEW RELATIONS
=====
relation between
  CL_CI_COMPONENT_CHILD with ID 1356 with key-values:
    CI_ID : 1000001279
  and
  CL_CI_COMPONENT_PARENT with ID 1350 with key-values:
    no key-values
relation between
  CL_CI_COMPONENT_CHILD with ID 1436 with key-values:
    CI_ID : 1000001302
  and
  CL_CI_COMPONENT_PARENT with ID 1432 with key-values:
    no key-values
REDUNDANT RELATIONS
=====
relation between
  CL_CI_COMPONENT_CHILD with ID 1356 with key-values:
    CI_ID : 1000001279
  and
  CL_CI_COMPONENT_PARENT with ID 1352 with key-values:
    CI_ID : 1000001275
relation between
  CL_CI_COMPONENT_CHILD with ID 1436 with key-values:
    CI_ID : 1000001302
  and
```

## Importing Data in Batches

### Reconciling your Data

CL\_CI\_COMPONENT\_PARENT with ID 1434 with key-values:

CI\_ID : 1000001297

The change list describes the changed items and relations with their class names and field names as in the xml file. To find the items and relations, you will need to follow the import mapping created for the item. In the import mapping, for example, class CL\_CI from the xml file is mapped to the Service Desk item Configuration Item. The property CI\_ID from the xml file is mapped to the ID field in Service Desk.

From the example change list the following entry represents an item in Service Desk that needs to be located and removed:

REDUNDANT ENTITIES

=====

CL\_CI with ID 105 with key-values:

CI\_ID : 1000000945

To locate the obsolete item, look for the Configuration Item in Service Desk and search for ID 105.

### Memory Usage

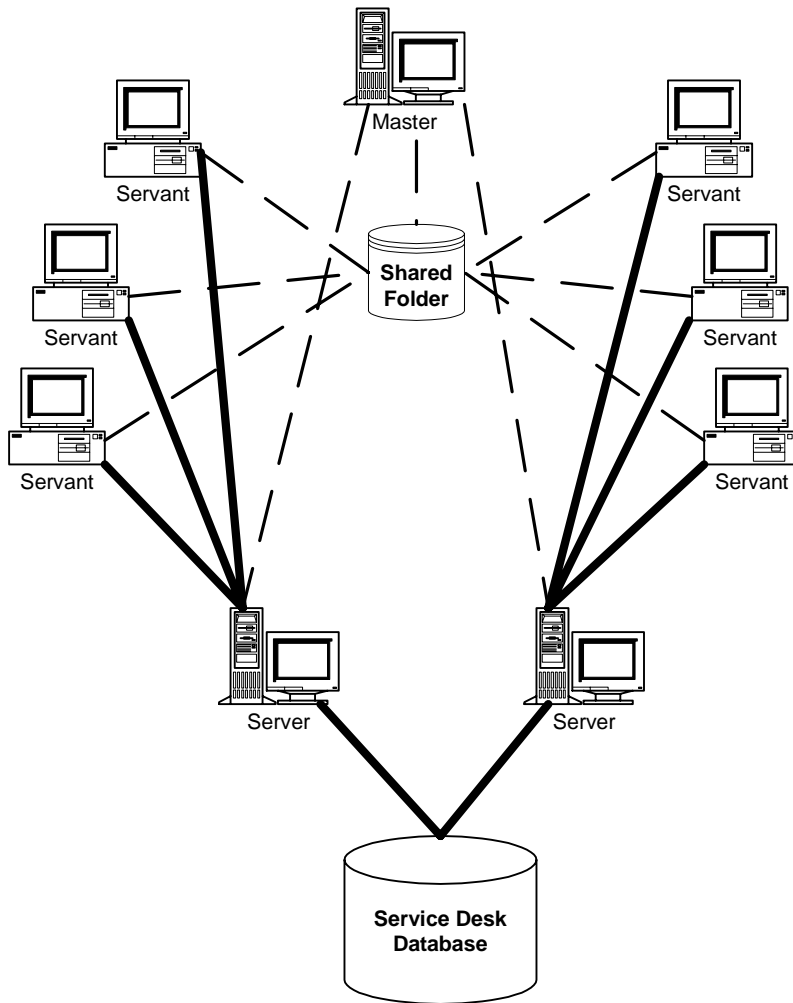
The values of all key fields are cached during import to make it possible to locate parent items and relations when processing a relation. The smallest memory format practical is used.

## Scalable Importing

Scalable importing is the process of using multiple application servers to import a single data exchange file. Scalable importing is only recommended for large imports that would normally take several hours to complete, more than 1 million items, for example.

In the following diagram the dashed lines represent the flow of information between one computer designated as the Master and a series of client computers that are used as Servants. The Master, which can be a server or client but must be running on a client installation, performs the function of taking the import work load, breaking it down into data chunks and then passing the import chunks on to the clients (servants) for execution. The solid lines represent the data flow to the database. The Master and Servants use a shared folder to communicate the work load:

**Figure 4-9 Scalable Importing**



## Requirements

The following requirements need to be met for the scalable importing process to work:

- Data Exchange files used for importing need be created with a Data Exchange task or from the command line using the `sd_export`, or `sd_exchange` cmd. These are the XML files that result from the

- export. Other types of files will not work.
- Use no more than one parent-child relation per configuration file.
  - Use unique search codes to reference the items that are related to items in other configuration files, if you had to split parent child relations for example.
  - Multiple Service Desk application servers.
  - Multiple Service Desk clients.
  - All machines must have access to a shared folder on your network.

## Preparing the Environment

To perform a scalable import you will need to set up the correct environment. Perform the following steps to create the correct environment:

1. Assign one computer the job of being the Master computer. Your Master should be the computer with the fastest, single, CPU. You will need to install the Client installation on this computer, even if it is a Service Desk server computer.

For example, the host name is `spaghetti` and it is using `local host` as a Service Desk server.

2. Create a shared folder on the Master machine or on your network that can be used by the Master and Servants for communicating the import load and import settings. Make this folder available as a shared folder. For example, the shared folder `c:\sd_import` is created and shared on the J drive as:  
`\\spaghetti\sd_import.`

3. Assign a number of Service Desk Client computers the job of being Servants. To determine the number of servants, you should have twice as much Servant processing speed as you do Service Desk server processing speed. For example, if your servers are three times as fast as each servant, you can assign six servants to every server used.

If you are using more than 20 servants you may need to increase the polling time of each servant to prevent the Master machine from spending too much time handling requests for work from the Servants. To determine the optimum polling time, multiply the number of Servants by 500. For example if you have 35 servants the

polling time should be set to 17500 milliseconds. The parameter for the polling time is `-polltime=`, for example `polltime=17500`, and is entered in the command line used for starting the servant. See “Starting the Servant Machines” on page 139.

4. Map a network drive on each servant to the shared folder you created in step 2 on the Master machine.

For example, Servant 1 has the hostname `tomato` and is connected to server `Macaroni`. Servant 2 has the hostname `cheese` and is also connected to the server `macaroni`. Server 3 has the hostname `anchovies` and is connected to the server `Pizza`. Each Servant is mapped to the location `\\spaghetti\sd_import` on the shared drive, in this example the J drive.

---

**TIP**

To create a shared folder, select the folder you want to share in your Windows explorer and select Sharing from the menu. In the Properties dialog box enter the name of the shared folder and select the Maximum Allowed option for the number of users.

When mapping the network drive the drive will be the drive where the shared folder is located, it is J in this example. The folder will be the folder where the shared folder is located on the master, in this example it is `\\spaghetti\sd_import`.

- 
5. Create an integration account that the Master computer and Servants can use for the scalable import.

For example; Username `imp_account`, Password `Italy`.

The following table provides an overview of the example environment used to explain the scalable importing process:

**Table 4-3**

**Example Environment for Scalable Importing**

User name	<code>imp_account</code>
Password	<code>Italy</code>
Shared folder	<code>\\spaghetti\sd_import</code>



	<b>Host Name</b>	<b>Server</b>	<b>Parallel</b>
Master	spaghetti	localhost	J:
Servant 1	tomato	macaroni	J:
Servant 2	cheese	macaroni	J:
Servant 3	anchovies	pizza	J:

## Exporting the XML Files

The XML files you use for scalable importing must be created with the `sd_export` command. The configuration files used for the export need to be configured to export single, not multiple references. In some cases you may need to split an existing configuration file, for example one configuration file can be used to export service calls, and another to export the history lines. When the scalable importing is done the XML file with the service calls will be imported first and then the XML file with the history lines. If service calls and history lines were exported in the same XML file, there is a possibility that a history line will be imported before the service call and this will cause an error in the import process.

---

### NOTE

In the [SYSTEM] section of your configuration file you can add the line, `CLASS_TO_XML=TRUE` to create an xml file for each class exported.

---

## Performing the Scalable Import

To perform a scalable import you need to first use a command to start the servants and then execute a command to start the Master.

### Starting the Servant Machines

Each servant needs to be started manually. On each servant computer:

1. Open a DOS prompt in the Service Desk Bin folder, If you installed the Client at the default location, this will be: `C:\Program Files\Hewlett-Packard\OpenView\Service Desk\Bin.`
2. Enter the following command: `sd_import username password server`

`-parallel=J` mapped network drive `-form`

For example: `sd_import imp_account Italy macaroni  
-parallel=J: -form`

3. The Import Progress form on the servant will appear and the Servant showing that the servant is active and waiting for work to be put in the shared folder on the mapped network drive by the Master.

---

**NOTE**

To stop a servant, enter **CTRL+ BREAK** from the DOS prompt. Closing the Progress form will not stop the servant.

---

### Starting the Master Machine

The master is started from the command line using the same information used for a Data Exchange task, or Task Group, with a few additions. To start the Master:

1. On the Master machine, open a DOS prompt from the Bin folder of your Client installation. If you installed the Client at the default location, this will be: `C:\Program`

`Files\Hewlett-Packard\OpenView\Service Desk\Server\bin.`

2. Enter the following command parameters: `sd_import username password server -exchange file -import mapping -parallel=J: -form -logfile.`

For example; `sd_import imp_account Italy localhost  
-data=persons_1.xml -mapping=person_mapping -parallel=J:  
-form -logfile=persons1`

3. The Import Progress dialog box will appear and the Master will connect to the Service Desk server, local host. The data exchange files (xml files in the command) will be divided into a number of chunks and distributed to the servants via the shared folder. The servants process the work and send it to the Service Desk server where it is then put in the Service Desk database.

You can define the size of the XML chunks created by the master and assigned to the servants by using the `-xmlsize` parameter. If you don't use this parameter the Master will split the XML file in the 1 MB chunks, you should try to have 3 times as many chunks as you do servants. For example `-xmlsize=250` will split the XML file into

250KB chunks.

4. The Master will stop and close when the entire Data Exchange file (XML file) has been imported. The results for the import will be visible in the Import Progress dialog box.

## Performing Scalable Import as a Task Group

It is possible to use one command to execute a group of data exchange tasks from the Master machine. You can do this by creating a .bat file containing all of the sequential commands that call `sd_import`. You will need to add the `-close` option to each `sd_import` command to close the Progress dialog box after each batch is finished. To stop all servants after the last command is executed, use the `-stopservants` parameter.

For example:

```
sd_import imp_account Italy spaghetti -data=persons_1.xml  
-mapping=person_mapping -parallel=J: -form -close  
-logfile=persons1 -xmlsize=250
```

```
sd_import imp_account Italy spaghetti -data=persons_2.xml  
-mapping=person_mapping -parallel=J: -form -close  
-logfile=persons2 -xmlsize=250
```

```
sd_import imp_account Italy spaghetti -data=persons_3.xml  
-mapping=person_mapping -parallel=J: -form -close  
-logfile=persons3 -xmlsize=250
```

```
sd_import imp_account Italy spaghetti -data=persons_4.xml  
-mapping=person_mapping -parallel=J: -form -close  
-stopservants -logfile=persons4 -xmlsize=250
```

## Reviewing the Log Files

The Master and each Servant will create its own log and error log file for each data exchange file that is processed. The log files will be created in the shared folder that is being used for the import in a sub-folder called `log`, for this example it is `\\spaghetti\sd_import\log` on the Master and `J:\log` from the network. A sub-folder will be created by the Master and each Servant using their hostname, for example

`J:\log\tomato\persons1.log`, and `J:\log\persons1_error.log`. If a log file by the same name is already present the next log file will add a digit, for example `J:\log\tomato\persons11.log`.

To locate all of the Servant error log files for a particular data exchange

file, enter the name of the xml file with \* as a wild card in the search. For example, to search for all errors for the data exchange file `persons_1.xml`, search for `persons1*_error.log`.

## Troubleshooting Scalable Importing Errors

An additional error log file exists for storing data chunks that could not be imported due to an error. The error folder will be created in the shared folder used for the scalable importing, in this example it is `J:\error\`. Two types of errors may occur during the scalable import process:

- **Time Out:**  
When the Master is finished assigning all work, it must wait for the Servants to finish the import process. During this period the Master will check on the Servants to make sure that they have not timed-out. The default time-out setting is at 20 minutes and if there is no activity from a servant for a period of 20 minutes the Master will take that servant's work and put it in the error folder. The work will be given the prefix `time_out`.
- **Fatal Error:** If a Servant or Master machine encounters a severe error when importing a data chunk, it will stop importing and the data chunk it was working on will be put in the error folder with the prefix `fatal_error`.

If a time out or fatal error has occurred the following steps may be helpful in locating and solving errors:

1. Search the log file for the error that caused the time out or fatal error. For example if a data chunk is in the error folder (`J:\error`) called `\fatal_error_persons1_12.xml`, search the log folder (`J:\log`) for the file, `persons1*.log`, for example. Then search that file for the text, `persons1_12.xml`, for example.
2. If that search does not lead to any error log files, and you are sure you performed the search correctly, go to step 4.
3. If the search resulted in at least one error log file, open the log file and view the errors. If possible, open the XML data chunk that is in the error folder, repair the problems that caused the error and try importing it again, see step 5 for instructions. If the error is too severe you will need to perform the entire import task again.
4. If you did not find an error log file or any indication of what went wrong, the XML file may be corrupt. If you can open the XML file and

it looks ok try importing it again as described in the following step. If you cannot open it successfully you will need to import the data exchange file again.

5. To re-import a data chunk that failed, copy the xml file (data chunk) from the error folder to the Master machine. Start the `sd_import` process again, without using the `-close` parameter and replace the data exchange file name with the name of the data chunk.

For example; `sd_import imp_account Italy localhost  
-data=fatal_error_persons_1_12.xml  
-mapping=person_mapping -parallel=J: -form  
-logfile=personsl_12`

Check the Progress dialog box for errors and warnings during the import process. If no errors or warnings occur, the import was successful.

---

## Command Line Parameters

Data exchange processes use the `sd_exchange.cmd` file in the Data Exchange folder. When using the Data Exchange features directly from the Administrator Console in Service Desk the application takes care of accessing the commands to export, import, or both export and import combined. To perform these functions from the command line you will need to use one of the following command lines:

**Table 4-4**

**Complete Importing and Exporting Commands**

Mode	Syntax
Export	<code>sd_exchange export &lt;config file&gt; &lt;exportlog&gt;&lt; xmlfile&gt;</code>
Import	<code>sd_exchange import &lt; input file&gt;&lt; username&gt;&lt; password&gt;&lt; mapping &gt;&lt;debug&gt;&lt; import log&gt;&lt; temp dir&gt; &lt;reconciliation data file&gt;&lt; reconciliation change list&gt;</code>
Export/ Import	<code>sd_exchange export_import &lt;config file&gt;&lt;export log&gt; &lt;in-outputfile&gt;&lt;username&gt;&lt;password&gt;&lt;mapping&gt;&lt;debug &gt; &lt;import log&gt;&lt;temp dir&gt;&lt;reconciliation data file&gt;&lt; reconciliation change list&gt;</code>

The following sections further explains the commands listed above.

### Exporting Parameters

The following command line can be used from the command line to export data: `sd_exchange export configfile exportlog xmlfile`. For example: `sd_exchange export ito.ini ito.log ito.xml`. See “Importing and Exporting Parameters” on page 145 for an explanation of the syntax used.

### Importing Parameters

The following command line can be used from the command line to import data: `sd_exchange import <input file>< username><password><mapping> <debug><import log>< temp dir> <reconciliation data file> <reconciliation change list>`

For example: `sd_exchange import ito.xml ITO_server1  
ITO_server1 ito_mapping Y ito.log C:\temp.`

An example command line including reconciliation follows:

`sd_import system servicedesk LOCALHOST dta5 dta5.xml Y  
dta5_imp.log c:\temp dta5_200101101645.xml dta5_changes.log`

See “Importing and Exporting Parameters” on page 145, for an explanation of the syntax used.

## Importing and Exporting Parameters

The following command line can be used from the command line to export and import data:

`sd_exchange export_import <config file><export log>  
<in-outputfile><username><password><mapping><debug> <import  
log><temp dir><reconciliation data file>< reconciliation  
change list>`

For example: `sd_exchange export_import ito.ini ito_export.log  
ito.xml ITO_server1 ITO_server1 ito_mapping Y ito_import.log  
C:\temp ito_200101101645.xml ito_changes.log.`

The following table explains the syntax of the parameters used in the command line:

**Table 4-5**

**Importing and Exporting Command Syntax Defined**

Syntax	Explanation
Configfile	Configurable .ini file containing the export configuration settings for the <code>sd_export.exe</code> command.
exportlog	The export log file is produced by <code>sd_export.exe</code> .
In-outputfile	The XML file, produced by <code>sd_export.exe</code> that will be used to import the data.
Username	A Service Desk account used for the integration
Password	The password that corresponds to the account mentioned above.

**Table 4-5 Importing and Exporting Command Syntax Defined**

<b>Syntax</b>	<b>Explanation</b>
Mapping	The name of the import mapping that you have defined in the Service Desk application, using Data Exchange.
Debug	Y will turn on the debug feature and provide you with a detailed log. N will turn it off.
Importlog	The logfile produced by <code>sd_import.exe</code> .
Tempdir	A folder for placing temporary files.
Reconciliation data file	The name of the XML file produced by your previous data exchange session. This file will be compare with the current xml file.
Reconciliation change list	The file name used for creating the change log file.



## Integration Accounts

All external applications integrating with Service Desk need an account for access to the Service Desk application. Accounts can be created for users who do not need access to the user interface but need to access the application for integration purposes or for using the Service Pages. This type of account can be set from the **TOOLS** menu in the Administrator's Console. Detailed information on creating accounts can be found in the *HP OpenView Service Desk: Administrator's Guide*.

Keep the following in mind when creating your integration accounts:

- You can enter a host name for an external application. For example, if you set up an integration to send incidents detected in the network to Service Desk, database rules can be used to send a confirmation message back to the application host that sent the incident. The following example shows how this looks when you send service events to Service Desk from three different ITO servers:

Account	Host name
ito_server1	server1.companyaccount.com
ito_server2	server2.companyaccount.com
ito_server3	server3.companyaccount.com

- Create integration account names that can be easily recognized. Use the same name for your import mapping to avoid confusion. For example:

Account	Import mapping
nnm6_import	nnm6_import
DTA_import	DTA_import

- Establish all roles that will be needed for the integration to perform its functions. If importing data, access needs to be granted to view and change certain areas within Service Desk, depending on what type of data you import.

Importing Data in Batches  
Integration Accounts

- When any account is given access to the user interface, that account will be counted as a registered (paid for) user in Service Desk.

For additional information about accounts refer to the Online help, or the *HP OpenView Service Desk: Administrator's Guide*

---

**NOTE**

Account information used to access an external data source for exporting is set in the configurable extractor file. Service Desk accounts only provide access to Service Desk for the import process.

---

---

**TIP**

Service events can operate with any valid Service Desk account. Using an account developed especially for the external application makes it easier to identify where the service event originated.

---

---

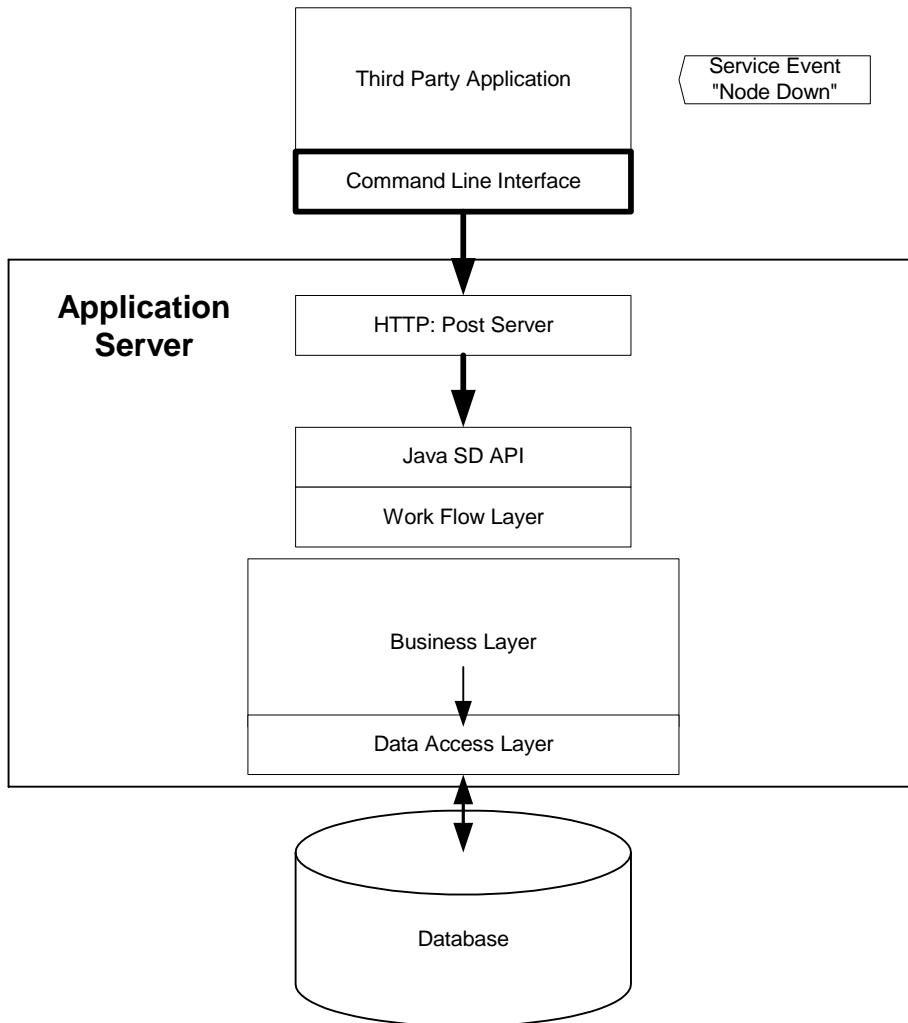
## 5 **Importing Service Events**

You can use a command line to import individual classes into Service Desk. For example, when a network management application detects a new node, the node information can be automatically inserted into Service Desk by the service event command, `sd_event.exe`.

## The Service Event Import Process

To import service events a command line is entered in the external application. The command line directs the service event data into the Service Desk database:

**Figure 5-1** Inbound Service Events



## Service Desk Event Command Line

`Sd_event.exe` makes it possible to insert, update, or delete an item in Service Desk. The command can be entered on the command line as follows:

```
sd_event.exe[-l log file][-s server][-p port][-a
account/password][-x mapping][-c class][-m modus][-v Value list][-o
alternate server:port]
```

An example of this command line follows:

```
sd_event.exe -l sd_event.log -s southside -p 30980 -a
system/servicedesk -x external_event -c incident -m insert
-v description="test 1234" status=s1 event_id=654321
information="my info" impact=2 priority=Low ci=SERVICEDESK
-o jackpot:30980,dustbin:30980
```

The easiest way to use the `sd_event` command line is to put parameters in the configuration file and use a reference to the configuration file in the command line with the `-f` switch and a values list `-v`. The structure is:

```
sd_event.exe -f<ConfigurationFileName.ini> -v <Value List>
```

for example:

```
C:\project\revelation\bin\sd_event.exe -f sd_event.ini -v
description="test 1234" status=s1 event_id=654321
information="my info" impact=2 priority=Low ci=SERVICEDESK
```

The following table provides definitions for the switches used:

**Table 5-1**

### Service Event Switches

Switch	Name	Description
-l <small>(small L)</small>	LOGFILE	Name of the file where information about the server is recorded.
-e	ERROR_LOGFILE	Used to specify the error log file.
-s	SERVER	Name of the application server used to find the HTTP:Post process

**Table 5-1 Service Event Switches**

Switch	Name	Description
-p	PORT	The port number to access the HTTP:Post process, (default is 30980).
-a	ACCOUNT/PASSWORD	A Service Desk user account, with a forward slash / followed by the account password.
-x	MAPPING	The name of mapping to be applied for default classes, property mapping and value mapping.
-c	CLASSNAME	The name of the class defined in the Import Mapping dialog box, for example: pc, node, network, incident.
-m	MODUS	Modus being used: insert, update or delete.
-v	VALUE_LIST	List with property names and property values [property=value]. You must use existing Service Desk values.
-o	ONFAIL	Optional list of alternate servers to use if the primary application server cannot be reached. <i>servername::port,servername:port</i>

The -x mapping and the -c class name options are taken from the import mapping set in Service Desk. You can use an existing import mapping or create a new one for every service event, as long as the import mapping used includes the class you want to import into Service Desk. For more information about import mapping see “Mapping Service Events” on page 107.

## Special Characters

`sd_event.exe` can handle the use of special characters like tabs, single and double quotes, percent signs, dollar signs and numerous international characters. Since `sd_event.exe` is a command line tool, the following characters need to be represented differently:

New line	<code>\n</code>
Tab	<code>\t</code>
Double quote	<code>\"</code>

---

### NOTE

`Sd_event` is 8-bit and uses the UNICODE character set. Any additional character code conversion necessary when using other character sets with `sd_event` is the users' responsibility.

---

## Using the Service Event Configuration File

The default name for the configuration file is `.\sd_event.ini`. This file will need to be modified to fit your organizational needs. The following example configuration file is configured to insert incidents from ManageX:

### Example 5-1

#### Example Service Event Configuration File

```
[SD_EVENT]
LOGFILE=c:\temp\sd_event_managex.log
ERROR_LOGFILE=c:\temp\sd_event_managex_error.log
ACCOUNT=managex4_event/servicedesk
SERVER=localhost
PORT=30980

ONFAIL=jackpot:30980,dustbin:30980
MAPPING=external_event
CLASSNAME=incident
```

MODUS=insert

LANGUAGE=GB

The parameters in the configuration file are used for the following:

**Table 5-2 Service Event Configuration File**

<b>Parameters</b>	<b>Definition</b>
LOG FILE	Log file name
ERROR LOGFILE	Error log file name
ACCOUNT	Login name/password
SERVER	Server name
PORT	Port number
ONFAIL	One or more Servers with port numbers to be used when an event cannot reach your primary server. The format is: servername:port, servername:port, servername:port
MAPPING	Mapping name
CLASS NAME	Class name
MODUS	Modus type (insert, update, delete)
LANGUAGE	Language for error messages. A corresponding language file must exist. Two message files are provided in the Bin directory; <code>sd_event_gb.msgs</code> for English (the default), and <code>sd_event_nl.msgs</code> for Dutch.

---

**NOTE**

Values that come from parameter switches will take precedence over default values in the configuration file.

---

## Service Event Examples

Detailed information and examples for exchanging data using the service



event command are available for:

- HP OpenView Network Node Manager, see “Integration With Network Node Manager” on page 173 for detailed information.
- HP OpenView VantagePoint, see the VantagePoint Operation Integration Guide for detailed information.
- HP OpenView ManageX, “Integration With ManageX” on page 189 for detailed information.

## Resending Service Events

When events are sent to Service Desk and Service Desk cannot be reached you can use the ONFAIL option in your `sd_event` configuration file to automatically send the event to one or more alternate servers. You can also save events that fail to make it to Service Desk in an error log file and send them at another time. The events will be written to the error log file if they do not reach your primary or one of your ONFAIL servers.

---

### NOTE

If an event fails due to syntax errors you will need to locate the errors and correct them before attempting the resend option. If you don't, the event will fail again.

---

To send the failed events, rename the error log file so that you can continue to gather incoming errors while you send the failed events. The events will be sent to the last server tried that failed. For example if your primary server `jumpstart` cannot be reached, and your ONFAIL server `dustbin` cannot be reached, the resend command will use the server `dustbin`. The command line to use is:

```
sd_event.exe -f [error log file] -r
```

For example, to resend one or more events this will look like: `sd_event -f sd_resend.log -r`.

The parameter `-b` can be used to send only events with `line send=True` to the application server.

The import mapping that was used when the error log file was created will be used to resend the service events.

The default location for the error log file is

`c:\temp\sd_event_error.log`. An example error log file follows:

### Example 5-2

#### Error Log File

```
[EVENT_280]  
VALUE_LIST="username=service#password=desk#mapping=scotttiger#  
className=EMP#modus=INSERT#EMPNO=1234#ENAME=Mr .  
Jones#DEPTNO=20#"
```

```
SERVER=server1
PORT=30980
CLIENT_ERROR=
LANGUAGE=GB
TRY=1
LOGFILE=sd_event_scott.log
ERROR_LOGFILE=sd_event_error_scott.log
TIMESTAMP= 4/14/2000 17:04:27
SEND=true

[ EVENT_371 ]
VALUE_LIST="username=service#password=desk#mapping=scotttiger#
className=EMP#modus=INSERT#EMPNO=1234#ENAME=Mr.
Jones#DEPTNO=20#"
SERVER=server1
PORT=30980
SERVER_RESPONSE=ERROR: You must fill in the Search code box.
LANGUAGE=GB
TRY=1
LOGFILE=sd_event_scott.log
ERROR_LOGFILE=sd_event_error_scott.log
TIMESTAMP= 4/14/2000 17:04:52
SEND=true
```

## Preventing Event Storms

When hundreds of events occur at almost the same time it is referred to as an event storm. If unprepared for, event storms can degrade the performance of your system. You can prevent event storms by implementing the tools explained in this section, which will help control how incoming events are handled.

---

### NOTE

Service Desk is not designed to handle a large number of events administratively. Normally it is the task of the event source application to handle event storms with an event correlation system.

---

## Queuing Events

To prevent event storms, you will need to implement the following queuing tools for all your service event processes. The queuing tools manage incoming events so that they are handled one at a time. Three tools and a script are provided for this purpose:

- *Enqueue tool*: responsible for adding new entries to an existing queue.
- *Dequeue tool*: removes the entries one by one and executes the command in each entry.
- *Queuectl tool*: Allows for blocking, pausing, flushing and reporting the status of the queue.
- *Addentry.sh script*: Adds entries to the queue and starts the dequeuer as necessary. (The NT version of this script is called `addentry.cmd`)

## Creating a Queue

A queue is a directory containing a file for each queue entry. You can create a new queue with `#mkdir clock_queue`. An example of this on UNIX:

```
# mkdir clock_queue
# ./enqueue clock_queue xclock
# ./dequeue clock_queue
# for i in 1234
do
```

```
./addentry.sh clock_queue xclock
done
```

If the command for an item to be queued contains any spaces, it must be enclosed by quotes. On UNIX machines, use single quotes (apostrophes) if the command itself contains quotes, and if you use variables. For example: `./enqueue testqueue "my command $USER"`

On Windows NT machines, the quotes that are part of the command must be preceded by a backward slash: `\"`.

## Enqueue

`enqueue <queuename><command>` creates a new entry in the queue containing the command. The queue directory must be created prior to this action. The following exit codes are also applicable for `dequeue` and `queuctl`:

Exit Code	Description
0	The command was queued successfully
1	A command line parameter is missing (usage error).
2	There was an error accessing the queue because it was locked or possibly non-existent.

## Dequeue

`dequeue <queuename>` processes the entries in the queue by reading, executing, and deleting the entries one by one. A new process is started for each entry in the queue. The dequeuing process pauses until each new process finishes. The dequeuer exits when all entries in the queue are executed or when the queue is blocked.

## Queuctl

`queuctl <queuename> <flag>` This tool controls access to the queue processes. Available flags are:

BLOCK	Blocks all access to the queue. If a dequeuer is running it will exit. No new entries will be added.
-------	--

UNBLOCK	Removes blocks from the queue. Entries can again be added to the queue, and the dequeuer can be restarted.
ENTRIES	Returns the number of entries in the queue.
STATUS	Returns the current status of the queue. Possible STATUS return codes are:  [IDLE] = exit code 10, no dequeuer is running on this queue.  [RUNNING] = exit code 11, dequeuer is busy processing this queue.  [BLOCKED] = exit code 12, queue has been blocked by queuctl.  [HUH] = exit code 13, queue is in an unknown state.  Exit code 0= command completed successfully.

### Addentry

`addentry.sh <queuename><flag>`, and `addentry.cmd<queuename><flag>` for Windows NT. Adds entries to a queue, and starts the dequeuer if necessary. The script file calls the `enqueue` tool, and the `queuctl STATUS` and will start the dequeuer if the queue specified is in the STATUS IDLE.

## Service Desk Event Communicator and Database Rules

Bi-directional service event integrations, (for example, ManageX and ITO integrations) use database rules to send event information from Service Desk. Service Desk Agents are used to execute the commands on the third-party application. You will need to install the event communicator; including the `sd_event` program and the Service Desk agent for the bi-directional integrations. For detailed information about the database rules, see the *HP OpenView Service Desk: Administrator's Guide*.

Uni-directional integrations, that send events to Service Desk, use the `sd_event` program in the Event Communicator and not the agent.

### Resending Failed Service Events

When a service event is sent from Service Desk, and cannot reach the agent it is intended for, that action is stored on the Service Desk application server with the agent's name. When the agent is active again it calls the Service Desk application server and gathers the actions assigned to it.

---

#### NOTE

If your Service Desk application server is shut down, the service events that are stored on the server waiting for an agent to become active will not be saved.

---

Importing Service Events  
Service Desk Event Communicator and Database Rules



---

# 6 **Auditing and Troubleshooting**

There are a number of options available for auditing and troubleshooting data exchange. You can use the Viewer to view the exported XML file, you can analyze log files generated during the export and the import process, and you can use the `Debug` mode to create a detailed import log.

## The Viewer, Log Files, and the Debug Mode

You may be able to locate data exchange problems by viewing the exported file with the Viewer. The Viewer will automatically translate your XML file into HTML format and present it in an object tree structure in your browser. The object tree view gives a clear picture of what was exported into the data exchange file. To view an XML file, click `View` in the Data Exchange dialog box.

For more information on the Viewer See “Viewing Data Exchange Files” on page 89.

Another option available when checking the data exchange process is to look at the log files. Log files can be created for both the export and import processes by selecting that option from the Data Exchange dialog box. Log files are located in the `data_exchange\log` folder after they are created. An additional log file is created containing only errors. The error file is named *your data exchange task\_imp\_error.log*.

A detailed log file of the import process can be generated by selecting the `Debug mode` from within the Data Exchange dialog box. The `Debug mode` will generate a very detailed log of the entire import process. It can help you in identifying items that were not imported and potential causes, such as inaccurate mapping, or other errors.

## Troubleshooting

The data exchange process can only be run from the application server or from a data exchange server. If you are importing large quantities of data, a dedicated data exchange server is needed. If the data volume is low, the application server is all that is required. Configuration, XML, and log files can also only be accessed from the server and not from a client machine. The only action you can perform from a client computer is the import mapping process. The following directories must be present in the application server's directory:

- data\_exchange\config
- data\_exchange\XML
- data\_exchange\log

The `sd_export.exe` file, its dll files, and the Windows NT command scripts must be installed in the `bin` folder located in the Service Desk directory. The items in the following table must be in the correct folder for the data exchange process to work correctly:

**Table 6-1 File Directory**

File	Folder
sd_export.exe sd_import.exe sd_import_export.exe	bin
sd_exchange.cmd	bin
msvc60.dll version 6.00.8168.0	\WINNT\SYSTEM32
mfc42.dll version 6.008168.0	\WINNT\SYSTEM32
msvcrt.dll version 6.00.8397.0	\WINNT\SYSTEM32
sd_event.exe	bin
nmm6.ini	data_exchange\config

**Table 6-1 File Directory**

File	Folder
dta5.ini/dta4.ini	data_exchange\config
sms20.ini	data_exchange\config
extractor.xsl	repo\xsl

If you receive an error when trying to export, view or import data, one of the following problems may be causing it:

**Table 6-2 Problem Solving**

Problem	Cause	Solution
The XML file cannot be viewed.	The XML format is incorrect.	Check the configuration file and check the log files made during the export process to locate the error.
The XML file cannot be viewed.	The XML file is incomplete	Check the configuration file and check the log files made during the export process to locate the error.
The XML file cannot be viewed.	The DTD file is not in the correct location.	The file CIM_DTD_V20.dtd must be in the same file as the XML file.
The XML file cannot be viewed.	The XML file is too large to be viewed with the Service Desk viewing tool.	The viewer does not work with large XML files. View large XML files with a text editor.

**Table 6-2 Problem Solving**

<b>Problem</b>	<b>Cause</b>	<b>Solution</b>
Errors exporting data to an XML file.	The ID field name was used in the configuration file. The sd_export program uses the automatically inserts a field called ID to give each record a unique ID in the SML files. If an additional ID field is entered by the user it will cause errors during the extraction process	Don't use the ID field, use an alternative field name, for example NNM_ID, or Event_ID.
SQL errors during export process	The export log file contains all SQL statements, if something is inaccurate in the syntax, errors may occur.	Check the syntax of the SQL statement, or copy and paste the SQL statement directly to MS Access.
Not all data in the XML file were imported.	Syntax errors in the import mapping.	Check the log file for errors related to: case, extra spaces, and spelling. Correct any errors listed. Select the Debug mode to generate a detailed log and run the process again.
Not all data in the XML file were imported.	Incorrect or incomplete import mapping	Verify that all items are mapped in Service Desk accurately then run the process again.

**Table 6-2**                      **Problem Solving**

<b>Problem</b>	<b>Cause</b>	<b>Solution</b>
Not all items in the XML file were imported.	If the import mapping of a class that has already been mapped and imported into Service Desk is changed, to import that class to a different location in Service Desk, your data may not be imported correctly.	Always map classes to the same items in Service Desk.

Some additional tips which may help:

**Table 6-3**                      **Tips**

Set LOADTABLE=TRUE (default) in the configuration file if you think that it will fit in the memory space available. This can speed up the export process considerably.
If you experience errors during the import process. Try importing small XML files first. This is quicker and makes it easier to locate the problem causing the error.
Use the MAXRECORDS setting in the configuration file to limit the number of records selected during export, when you are initially exporting data.
Class and attribute names in the export configuration file, XML file, and import mapping are case sensitive. The import log file will show all attributes and classes that could not be matched.
Do not use the attribute name <i>ID</i> , it is reserved for internal use only.
Do not view large XML files with the Viewer, use a text editor instead.

**Table 6-3**

**Tips**

During the export of parent-child relations, it is important to note that: Columns that appear in PARENT\_RELATION have to be put into the column list when tables are cached [LOADTABLE=TRUE], because the extractor uses this information to find all children for a particular parent in memory.

## Error Messages

During the process of configuring an .ini file the following error messages may be generated. These problems are most likely to occur when you are modifying a previously configured file.

---

### NOTE

The use of upper case and lower case must be consistent within an .ini file (for example a table can't be named Sites\_DATA in one place and Sites\_Data in another). Although this does not produce an error message itself it may be the cause of other messages.

---

## Possible Errors While Reading the .ini File

Cannot find parent '<parent\_name>'.  
The class that is mentioned as parent does not exist.

Cannot read from file '<file\_name>'.  
The file is not readable.

Class '<class\_name>' cannot have itself as parent.  
A class cannot have itself as parent defined.

Connection to <dsn\_name> as <user\_name>/<password> failed.  
Connection to the external source using the settings in the DSN section failed.

Misplaced brackets in line '<line>'  
The order or number of brackets [ ] in the line is not correct.

No '=' in parent relation.  
No equal sign can be found in the parent relation.

No operator found in condition '<condition>'.  
The condition does not have an operator.

No references to tables found in condition '<condition>'.  
There must be 1 (filter) or 2 (join) references to tables or columns in the condition. The table name and column name, or column name only, defined in a filter or join must be found in the database.

No table and column found in ordering '<ordering>'.  
The ordering should refer to a table and column. The table name and



column name, or column name only, defined in a filter or join must be found in the database.

Parsing stopped.

Parsing has been stopped due to previous errors.

Unknown columnname '<column\_name>' in parent relation.

The parent's column mentioned in the parent relation is unknown.

Unknown tablename '<table\_name>' in parent relation.

The source table mentioned in the parent relation is unknown.

Wrong number of values in condition '<condition>'.

The number of values does not match the operator in the condition.

### **Possible Errors While Writing the .ini File**

Cannot write to file '<file\_name>'.

The file is not writable.

Auditing and Troubleshooting  
Error Messages

---

# **A**      **Integration With Network Node Manager**

The examples in this appendix were produced using NNM 6.1. The details of the integration are explained in following sections.

## Integration Possibilities

Network Node Manager (NNM) provides tools for fault, configuration, and performance management of multi-vendor TCP/IP and IPX/SPX networks. By integrating NNM with Service Desk you can:

- automatically send events from NNM to Service Desk;
- import NNM Nodes into Service Desk as Configuration Items.

### Creating Incidents in Service Desk

You can use `sd_event.exe` to automatically send event information to Service Desk. Events that can be sent from NNM to Service Desk include:

- Insert an incident with `OV_Node_Down`
- Insert a configuration item with `OV_Node_Add`
- Insert an incident with `OV_Segment_Critical`
- Insert an incident with `OV_Network_Critical`

### Importing NNM Nodes into Service Desk

Node information from Network Node Manager can be extracted and imported into Service Desk as configuration items. A Data Exchange task can be created to extract the information from Network Node Manager and import it into Service Desk. The extraction is done via an open database connectivity link (ODBC) to the NNM database. The data will first need to be exported to an Oracle database, see “Exporting NNM Data to a Data Source” on page 179 for additional information on creating an Oracle account for exporting NNM node information using ODBC.

---

#### NOTE

To prevent errors when importing CIs you must map the `searchcode` field in Service Desk to a unique property when you do your import mapping. Check the import mapping you are using to verify that the `searchcode` field is mapped to a unique field property.

The reason for this is that whenever you create or edit a configuration

item, Service Desk validates the search code entered and prevents you from saving the CI if the search code is already in use by another CI. If a non-unique default searchcode is entered in the template used for import mapping an error will result when you try to import more than one CI. The unique search code option is set in the General Settings dialog box. To view the settings, open the Administrators Console, then System Panel, and open the General Settings dialog box.

---

## Installation

This section provides an overview of the integration installation. For detailed installation instructions, refer to the Service Desk Installation Guide.

### Service Desk Application Server

After installing the Service Desk application server, you will need to install *Integrations*, with the Data Exchange and NNM options selected, on the Service Desk application server from the Service Desk CD-ROM. The following tools and files will be installed:

**Table A-1 Service Desk Server**

<b>Installed Items</b>	<b>Location</b>	<b>Status</b>
sd_event.exe	\Program files\Hewlett-Packard\OpenView\Service Desk 4.0\Server\bin	ready
external_event (import mapping)	Service Desk database (Can be configured from the client)	default values
sd_event.ini	\Program files\Hewlett-Packard\OpenView\Service Desk 4.0\Server\bin	default values

## Configuration

The Network Node Manager integration with Service Desk can be configured in the graphical user interface of that application, or from the `trapd.conf` file.

### Configuring Network Node Manager

The following configuration tasks need to be performed in the Network Node Manager application.

#### Select an NNM Event

Open NNM and select an event.

1. Start the Network Node Manager Services.
2. Start the Network Management Console.
3. In NNM file menu select `Event Configuration` from the `Options` menu.
4. In the `Enterprises` list, click `OpenView`. The `Events for Enterprise` window will show events that are generated by NNM processes.

As an alternative, you can work directly from the `conf/C/trapd.conf` file. By adding the `EXEC` keyword to the `trapd.conf` file you can pass an incident or a newly detected configuration item to Service Desk.

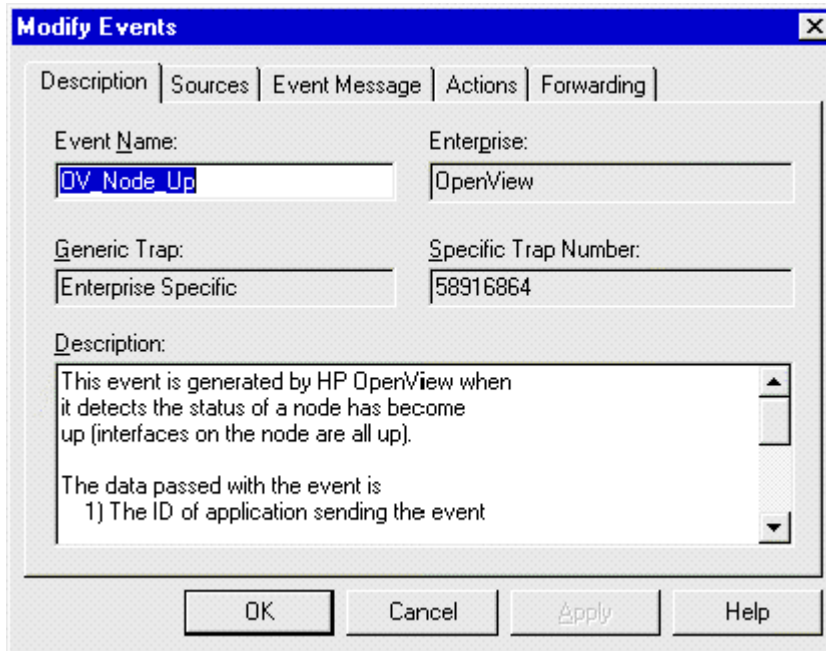
#### Modify an Event

Modify the event in NNM:

1. Double-click one of the events from the `Events for Enterprise OpenView` window, for example `OV_Node_Up`.

The `Modify Events` dialog box appears:

**Figure A-1**      **Modify an Event in NNM**



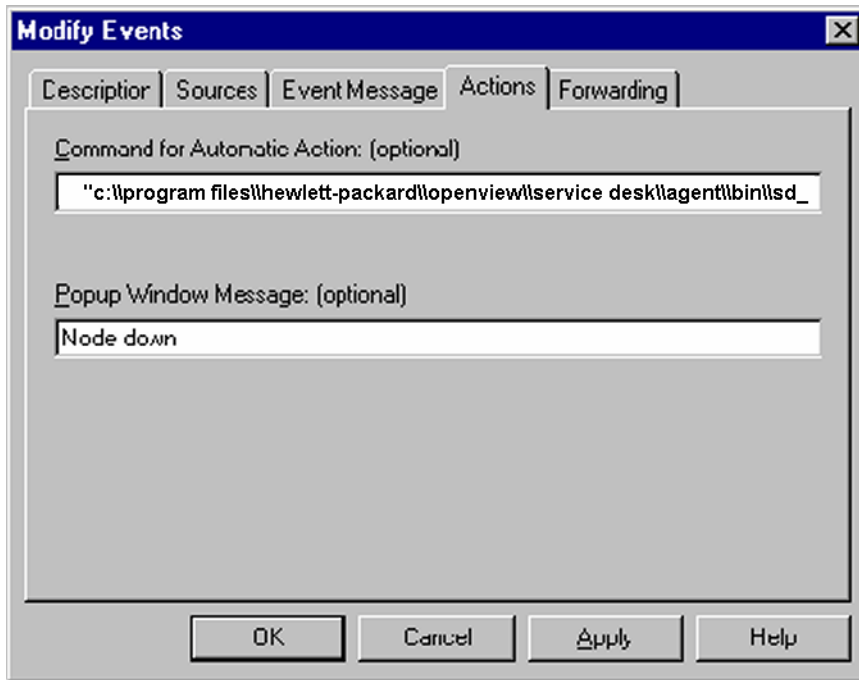
2. In the NNM application, click the **Actions** tab and enter the command line in the **Command for Automatic Action** field. The complete command line will look similar to:

```
"C:\\program files\\hewlett-packard\\openview\\service  
desk 4.0\\client\\bin\\sd_event.exe" -f  
"C:\\program files\\hewlett-packard\\openview\\service  
desk 4.0\\client\\bin\\sd_event.ini" -v  
event_id= \"$2 $x $X\" description= \"$2 went down at $x  
$X\"
```

Where **client** can be either **client** or **server**.



**Figure A-2 Service Event Command Line in NNM**



3. Click OK.
4. From the File menu click Save and then Close, to save and activate the new settings in NNM.

---

**NOTE**

Actions in NNM are executed by an action daemon. If the syntax of the action command line is not correct, error messages will appear in the NNM 6 directory at: log\ovactiond.log.

---

### **Exporting NNM Data to a Data Source**

HP OpenView Network Node Manager must have a data warehouse installed on an Oracle or SQL Server database to be able to access it with ODBC.

---

**NOTE**

---

For detailed information on how to create a data warehouse for NNM, refer to the NNM documentation supplied with your NNM application. The procedures that follow are to be used as a guideline only.

The following procedures for exporting NNM version 6 topology data to an Oracle or SQL Server database as a guideline:

For NNM on Windows NT:

1. Create an Oracle or SQL Server account called `ovdb` with the password `ovdb`.
2. Create an Oracle or SQL Server ODBC connection named `NNM6` to the Oracle or SQL Server account `ovdb`.
3. Create the Network Node Manager table structure in the `ovdb` account, with the following command:

**USAGE:**

```
ovdwconfig.ovpl [-rdb ODBC datasource] [-u user]
[-password password] [-type embedded/msSqlSrvr/oracle] [-load]
[-reload] [-env envVar=value]
```

For example, for an Oracle database:

**USAGE:**

```
ovdwconfig.ovpl -rdb NNM6 -u ovdb -password ovdb -type oracle -load
```

For example, for a SQL database:

```
ovdwconfig.ovpl -rdb nnm6sql -u ovdb -password ovdb -type msSqlSrvr
-load
```

4. Run `ovdwtopo -export -rdb nnm`. This will put the topology data in the Oracle account via the ODBC connection.

For NNM on HP-UX for Oracle:

1. Create an Oracle account called `ovdb` with the password `ovdb`.
2. The ODBC connection should be present after you have installed Network Node Manager, and is named `OVOracle`. The `odbc` file is: `/etc/opt/OV/share/conf/analysis/system_odbc.ini`, and is pointing to the OpenView database by default using the SQL\*Net® Alias `ov_net`.
3. Create the Network Node Manager table structure in the `ovdb`

```
account, with the following command: #ovdwconfig.ovpl -rdb  
OVOracle -u ovdb -password ovdb -type oracle -load
```

4. The following command will load the topology data into the Oracle ovdb account via the ODBC connection:  

```
# ovdwtopo -export -rdb OVOracle -v
```

## Configuring Service Desk

This section explains the configuration steps that must be done in the Service Desk application.

### Configuring the ODBC Connection

You will need to set the ODBC connection so that you can connect to the NNM data source.

To configure the ODBC connection:

1. Open the ODBC Data Sources on your computer. This is usually located in the System panel.
2. Open the System DSN tab and select one of the following drivers: Oracle 8, or Oracle 7.3 Version 2.5, or SQL server.
3. Select Finish.
4. Fill in the following fields:  
Data Source Name, example: NNM6  
Description, example: Network Node Manager  
SQL\*Net connect string, example: nnm\_server  
User ID: ovdb

Before you can connect and import the data, you will first need to export the NNM data, see “Exporting NNM Data to a Data Source” on page 179.

### Configure the Sd\_event File

From the \Program files\Hewlett-Packard\OpenView\Service Desk 4.0\Server\bin folder open the sd\_event.ini file, rename it and configure it as follows:

1. In the `Server` line enter the name of your Service Desk application server.

2. After Account enter the account created for this service event integration and the password.
3. Additional items which need to be configured can be seen in the example `sd_event.ini` file shown below with default values:

```
[SD_Event]
LOGFILE=c:\temp\sd_event.log
ERROR_LOGFILE=c:\temp\sd_event_error.log
ACCOUNT=nnm6_event/servicedesk
SERVER=localhost
PORT=30980
MAPPING=external_event
CLASSNAME=incident
MODUS=insert
LANGUAGE=GB
```

---

**NOTE**

You can open configurable `.ini` files for editing from the Data Exchange dialog box by entering the `.ini` file name, then click Edit. The file will open in a text editor.

---

### **Modify the Import Mapping**

In Service Desk, modify the `external_event` import mapping as needed.

### **NNM Variables**

A number of variables are available for use in NNM, which enable you to control the formatted output sent from NNM to Service Desk. Special variables can be used in the Event Log Message, Popup Window Message and Command for Automatic Action of the Add Event, Modify Events or Copy Event fields.

### **Special Characters**

Standard `Cprintf` formats will be converted to the equivalent ASCII format. All non-printable characters will be converted to the octal (`\ooo`) equivalent for display in the event browser, `trapd.log`, and when passed to the operator (manual) actions. The two exceptions are that a tab is displayed as `\t` in the event browser and `tapd.log` and as a space in pop-up messages. A new line is displayed as `\n` in the event browser and

trapd.log and as a new line in pop-up messages. All non-printable characters are passed in their original form to automatic actions executed by ovactiond.

### Sequential Attribute Variables

The \$ variables, shown in the following table, are used to access the sequential attributes that were received with the event. Each event has associated attributes, however the event can have no associated attributes. They are accessed using the \$n notation, where n is the positional attribute with 1 being the first attribute. The printing format is based on the ASN1 type of the attribute. These attributes are equivalent to the variable bindings (varBinds) in an SNMP trap. The variables are as follows:

**Table A-2 Sequential Attribute Variables**

Variable	Definition
\$#	Print the number of attributes in the event.
\$*	Print all the attributes as seq name (type): value strings, where seq is the attribute sequence number.
\$n	Print the nth attribute as a value string. It must be in the range of 1 to 99.
\$-n	Print the nth attribute as a seq name (type): value string. It must be in the range of 1 to 99.
\$(+n	Print the nth attribute as a name: value string. It must be in the range of 1 to 99.
\$>n	Print all attributes greater than n as value strings. This is useful for printing a variable number of arguments. \$>0 is equivalent to \$* without sequence numbers, names or types.
\$>-n	Print all attributes greater than n as seq name (type): value strings.
\$>(+n	Print all variables greater than n as a name: value strings.

### Special Information Variables

You can also include information from the incoming event by using the \$arg format specification. The following \$ variables are valid regardless of the type of event (SNMPv1, SNMPv2C, CMIP, GENERIC).

**Table A-3 Special Information Variables**

Variable	Definition
\$x	print the date the event was received using the local date representation.
\$X	Print the time the event was received using the local time representation.
\$@	Print the time the event was received as a number of seconds since the Epoch (Jan 1, 1970) using the time_t representation.
\$O	Print the name (object identifier) of the received event.
\$o	Print the name (object identifier) of the received event as a string of numbers.
\$V	Print the event type, based on how the event was transported. See NNM documentation for a list of supported types.
\$r	Print the implied “source” of the event in text format. This may not be the true source of the event if the true source is proxy for another source, such as when a monitoring application running locally is reporting information about a remote node.
\$R	Print the true source of the event in text format. This value is inferred by means of the mechanism that delivered the event. If the event was forwarded, this will display the address of the remote event framework.
\$c	Print the category the event belongs in.
\$s	Print the severity of the event.

**Table A-3 Special Information Variables**

Variable	Definition
\$N	Print the name (textual alias) of the event format specification used to format the event, as defined in trapd.conf.
\$F	Print the textual name of the remote event's machine, if the event was forwarded, otherwise print the local machine's name.
\$\$	Print the \$ character.

**SNMP-Specific Variables**

The following variables are meaningful for events created from SNMPv1 or SNMPv2 traps/informs:

**Table A-4 SNMP Specific Variables**

Variable	Definition
\$C	Print the trap community string. Set to public for non-SNMPv1 events.
\$E	Print the trap enterprise as a text string if possible, otherwise as in the \$e argument below. This option tries to use the enterprise name as formatted in trapd.conf, as opposed to \$O which formats using the MIB definitions, (typically a longer string). The event object identifier for non-SNMPv1 events implies this number.
\$e	print the trap enterprise as an Object ID string of numbers. The event object identifier for non-SNMPv1 events implies this number.
\$A	Print the trap agent addresses as defined in the trap PDU. This may be different from the agent that actually sent the event. If the name server knows about this node, the node name will be printed, otherwise the address will be printed.
\$G	Print the trap's generic trap number. The event object identifier for non-SNMPv1 events implies this number.

**Table A-4**    **SNMP Specific Variables**

<b>Variable</b>	<b>Definition</b>
\$S	Print the trap's specific trap number. The event object identifier for non-SNMPv1 events implies this number.
\$T	Print the trap's sysUpTime time stamp. This is the remote machine's time in hundredths of a second between the last initialization of the device and the generation of the trap. It is not the time the event was received (see \$s, \$X, and \$@). For non-SNMPv1 events this value is 0.



---

## Example of NNM on UNIX Server

If you are running NNM on a UNIX server you will need to install the items mentioned in “Service Desk Application Server” on page 176, and you will also need to install the Event Communicator from the HPOVSD depot on your NNM for UNIX application server. For instructions on installing the HPOVSD depot files, refer to the Service Desk Installation Guide.

**Table A-5 NNM UNIX Server**

<b>File</b>	<b>Location</b>	<b>Status</b>
sd_event.ini	NNM server opt/OV/SD/bin	default values
sd_event	NNM server /opt/OV/SD/bin	ready

## Example of NNM on UNIX

It is possible to send service events from NNM on a UNIX system to Service Desk. The following example is for the OV\_Node\_Down event. From the Options menu in your NNM application:

1. Select Event Configuration, and then OpenView in the dialog box that opens.

2. Scroll down the Event Name list and double-click OV\_Node\_Down.

3. The following command and parameters can be used:

```
cd /opt/OV/SD/bin; /opt/perl5/bin/perl  
/opt/OV/SD/bin/sd_event -f /opt/OV/SD/bin/sd_event.ini -v  
event_id="$2 $x $X" description="$2 went down at $x $X"  
ci=$2
```

## Managing Event Storms on UNIX

You can install the queuing tools from the HPOVSD depot file to handle event storms. The queuing tools are not mandatory, the integration will work fine without installing them. See “Preventing Event Storms” on page 158 for additional information on the queuing tools available with

Integration With Network Node Manager  
Example of NNM on UNIX Server

Service Desk. Refer to the *HP OpenView Service Desk: Installation Guide* for installation instructions.

---

## **B** **Integration With ManageX**

This example in this appendix were created using ManageX 4.23. The following sections provides information useful in setting up an integration with ManageX for the purpose of sending service events to Service Desk and sending service events from Service Desk back to ManageX.

## Integration Possibilities

The ManageX Management Console allows system administrators to explore domains and machines in the network. You can observe, administer, and manage network machines, all from a central console. A performance monitor provides detailed real-time examination of performance data. Events detected by ManageX can be forwarded to Service Desk and registered as service calls so that help desk personnel and specialists can quickly react, solving the problem. The following features are available with this integration:

- Service Events detected in ManageX can be sent to Service Desk. The incoming service events are registered as incidents in Service Desk.
- Service Events can be sent from Service Desk to ManageX when an incident is created, when the status changes, or when it is closed. This part of the integration can be made by creating a database rule to send an acknowledgment message.

## Installation

This section provides an overview of the items that must be installed for the integration to work. For installation instructions refer to the *HP OpenView Service Desk: Installation Guide*.

### Service Desk Application Server

The integration needs to be installed on your Service Desk server and your ManageX server.

After installing the Service Desk application server you will need to install Integrations with the Data Exchange and ManageX options selected, from the Service Desk 4.0 CD-ROM.

The following integration files will be installed:

**Table B-1 Service Desk Server**

Installed Item	Default Location	Status
sd_event.exe	\Program files\Hewlett-Packard\OpenView\Service Desk 4.0\Server\bin	ready
external_event (import mapping)	Service Desk database (Can be configured from the Data Exchange GUI)	default values
sd_event_managex.ini	\Program files\Hewlett-Packard\OpenView\Service Desk 4.0\Server\bin	default values

**Table B-1 Service Desk Server**

<b>Installed Item</b>	<b>Default Location</b>	<b>Status</b>
ServiceDeskSamplePas sThru.mxc	\Program Files\HP OpenView ManageX\Policies \LightsOut	default values
sendmessage.exe	\Program files\Hewlett-Pa ckard\OpenView\S ervice Desk 4.0\Server\bin	ready

For additional installation information, refer to the *HP OpenView Service Desk 4.0 Installation Guide*.

## **ManageX Application Server**

If ManageX is running on the same server as Service Desk, the files and tools needed for the integration will be installed when you install Integrations from the Service Desk CD-ROM and the ManageX option is selected.

The ManageX LightsOut Console policy called: ServiceDeskSamplePassThru.mxc will be installed in C:\Program Files\HP OpenView ManageX\Policies. You will need to perform some additional steps to complete the installation as explained in the next section.

If ManageX is running on a server different from the ManageX server see “ManageX on a Different Server” on page 193.

### **ServiceDeskSamplePassThru**

ServiceDeskSamplePassThru.mxc is installed out-of-the box to Service Desk\Bin when you install ManageX from the Service Desk CD-ROM. Some user actions still need to be performed manually as follows:

1. From the ManageX console, select the ManageX Server from the Device Selector dialog box and click Apply. If you don't see the Device Selector dialog box when you start the ManageX console, right-click

- the OpenView ManageX node and choose Device selector.
2. Double-click Policies from the ManageX console tree then double-click Available Policies.
  3. In the results pane, right-click to select the ServiceDeskSamplePassThru policy, select All Tasks , then select Install from the shortcut menu.
  4. If the policy was installed successfully, two messages will appear in the ManageX Message Reader.

---

**TIP**

If the LightsOut Console Policy ServiceDeskSamplePassThru.mxc is not listed, right-click on the Policies folder then click All Tasks. Click Set Directory and then navigate to the directory with the ServiceDeskSamplePassThru.mxc policy and click OK.

---

### **ManageX on a Different Server**

We recommend that the ManageX console runs on the same server as the Service Desk application server. If you run ManageX on a different server you will need to perform the following additional steps:

- The sd\_event\_manageX.ini, sd\_event.exe and ServiceDeskSamplePassThru.mxc files need to be copied to the ManageX server.

The ServiceDeskSamplePassthru policy contains a reference to SD\_Home. Because Service Desk is running on a different server, the file will not be able to find the sd\_event.exe and sd\_event\_manageX.ini files. To correct this, create a directory tree that mirrors SD\_HOME on your ManageX machine, and place the sd\_event.exe and sd\_event\_manageX.ini files in the Bin folder of that directory.

- Database rules need to be changed, reflecting the ManageX server as the agent name.
- Modify sd\_event\_manageX.ini to refer to the correct Service Desk server.

## Configuration

This section explains the configuration tasks you will need to perform in Service Desk followed by the configuration tasks you will need to perform in the ManageX application.

### Configuring Service Desk

This section explains the configuration steps that must be done in Service Desk.

#### Configure the .ini File

Copy the file `sd_event_managex.ini` in Service Desk. The default location of `sd_event_managex.ini` is: `\Program files\Hewlett-Packard\OpenView\Service Desk 4.0\Server\bin`

1. In the `Server` line, enter the name of your Service Desk application server.
2. After `Account` enter the account created for this service event integration and the password. The installation program creates the account `managex4_event` by default with the password `servicedesk`.
3. Additional items which need to be configured can be seen in the sample `sd_event.ini` file shown below with default values:

```
[SD_Event]
LOGFILE=c:\temp\sd_event.log
ERROR_LOGFILE=c:\temp\sd_event_error.log
ACCOUNT=managex4_event\servicedesk
SERVER=localhost
PORT=30980
MAPPING=external_event
CLASSNAME=incident
MODUS=insert
LANGUAGE=GB
```

#### Database Rules

Configure database rules in the business logic portion of Service Desk to send events back to ManageX. A database rule will be installed when you install the ManageX integration from the Service Desk CD-ROM. Verify



that you agree with the default values set for the rule, and turn it on. Database rules can be used to send an acknowledgment to ManageX when an incident is created, when the status of an incident changes, and when the incident is closed, for example.

1. From the Service Desk Tools menu, click System, then Business Logic, then Database Rules, and select Incident.
2. The rule Send message to ManageX is provided for you with default values. Right-click and select Open to modify the rule.
3. The Conditions for the rule should be similar to:  
*specify Registration\_login=manageX4\_event  
status=closed*
4. The Command Exec action to send service events to ManageX must be configured to be executed on the ManageX machine. If ManageX is installed on the local host you will still need to specify the machine's real host name.
5. An example command line for the closed status is:  
*sendmessage.exe, -d="incident<ID> has been closed"*
6. Verify that the rule is not blocked. If it is blocked it will not be executed.

### **ManageX Server Account**

A ManageX account is created automatically when you install the ManageX integration. The account is called `manageX4_event`. If you have additional ManageX servers integrating with Service Desk you may want to create additional accounts, for example `manageX_<machine>`.

Additional information on creating accounts is available in the Online Help and in the *HP OpenView Service Desk: Administrator's Guide*.

## **Configuring ManageX**

This section explains the configuration tasks you must perform in the ManageX application

### **Configure the Acceptance Filter**

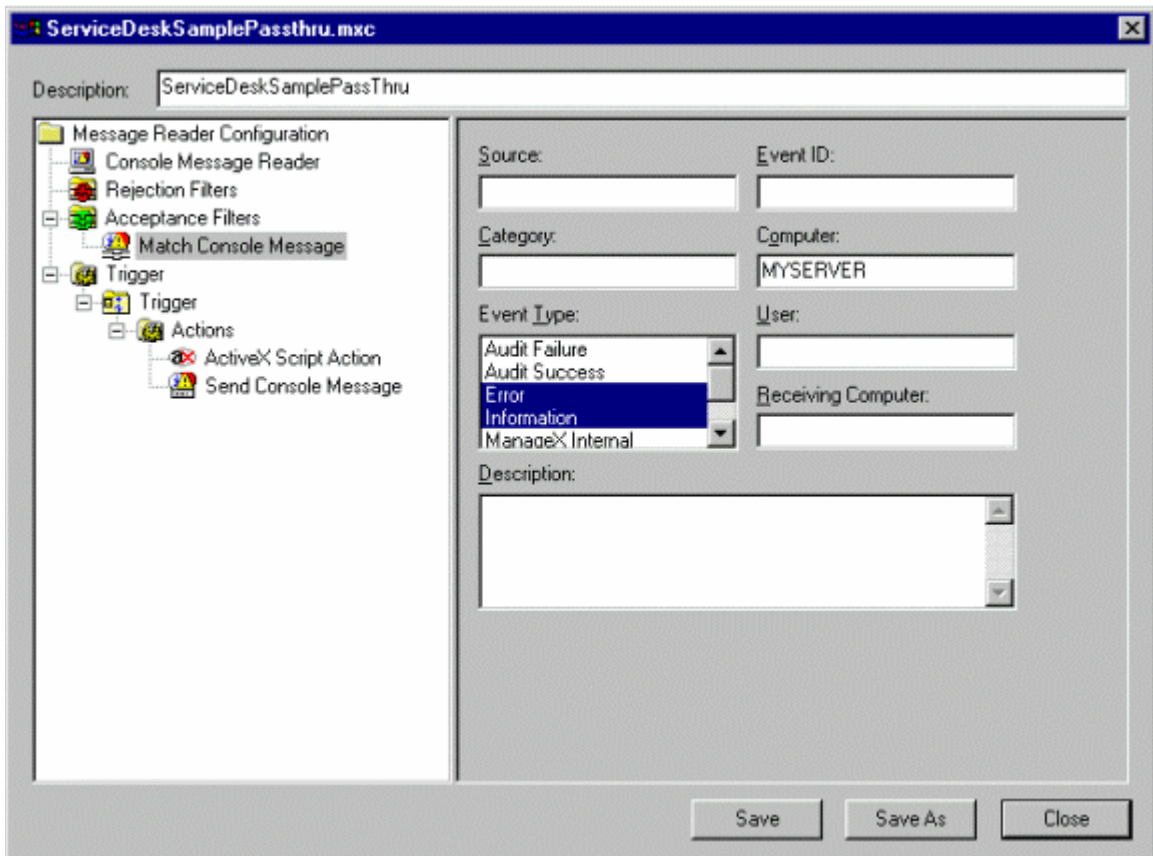
Set the acceptance filter in ManageX. This will define what events are forwarded to Service Desk:

1. From the ManageX console double-click Policies, then Available,

## Integration With ManageX Configuration

- and then LightsOut.
2. Double-click `ServiceDeskSamplePassthru.mxc`, and the ActiveX script action will show the `mxc` file in Visual Basic format. You can edit the script to send the correct event information
  3. In the Message Reader Configuration folder click Acceptance Filters and then double-click Match Console Message:

**Figure B-1 Set ManageX Acceptance Filters**



4. In the results screen in the `Computer` field, enter the computer you want to accept events from, this should be your ManageX console. In the `Event Type` field select the type of events you want sent to Service Desk from the list.

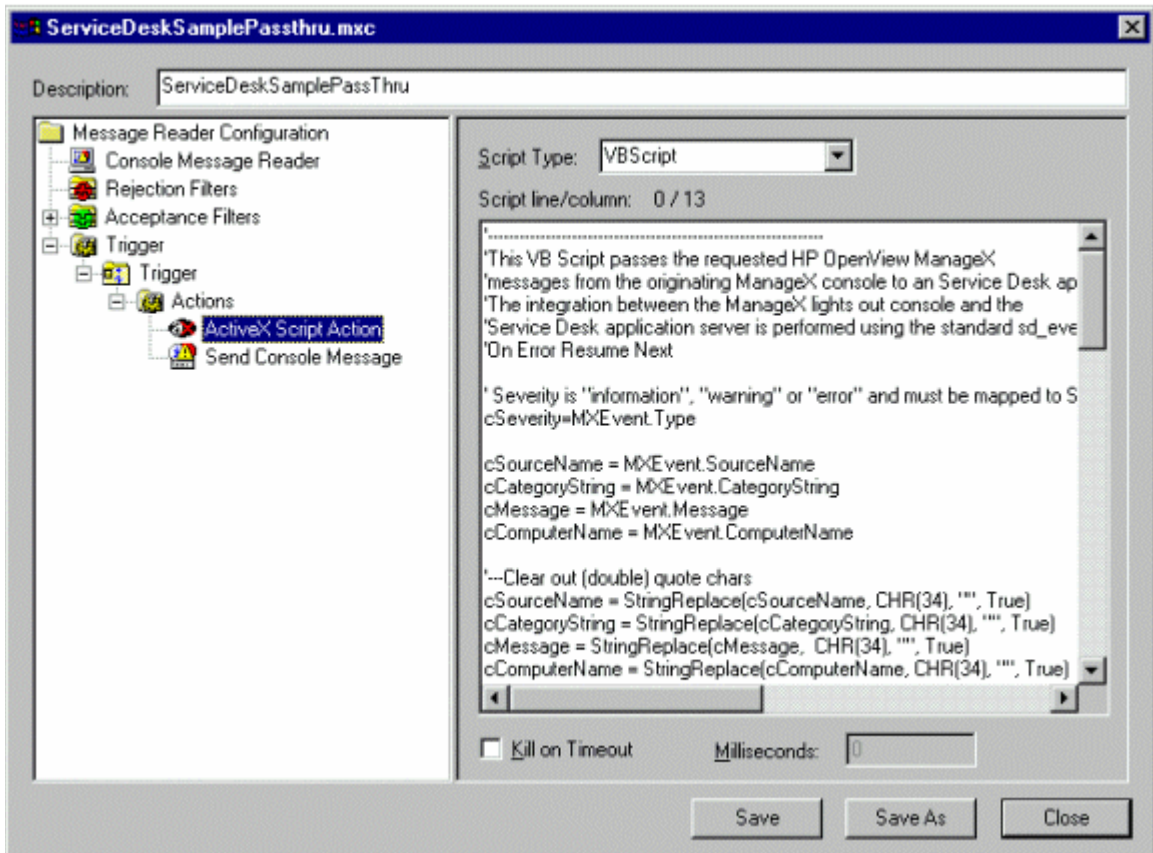
5. Click `Save` when you are finished.
6. After changing a policy you will need to install it. To install the policy, right-click on the policy, select `All Tasks`, then `Install`.

### **ManageX Lights Out Policy**

Configure the ManageX LightsOut policy. The LightsOut policy gathers events coming from ManageX agents and sends them to Service Desk, even when the ManageX application is not running.

1. From the ManageX console double-click `Policies`, then `Available`, and then `LightsOut`.
2. Double-click on `ServiceDeskSamplePassThru.mxc`, an ActiveX script will show the `mxc` file in Visual Basic format. You can edit the script to send the correct event information:

**Figure B-2** ManageX ActiveX Script Action



3. Select Send Console Message. Change Specific Additional Destination dialog box that appears to refer to your console.
4. The following information may be helpful while configuring the policy:
  - The lights out policy ServiceDeskSamplePassthru.mxc contains information you will need for the import mapping. The default information is as follows:

```
cDescription = "description=" & chr(34) & cMessage & chr(34)
cCI = "ci=" & chr(34) & cComputerName & chr(34)
cInformation = "information=" & chr(34) & "Message:" & cMessage
& "\nEventid: "& "$(EventID)"& chr(34)
cImpact = "impact=" & chr(34) & cSeverity & chr(34)
cClassification="classification=" & chr(34) & cCategoryString &
chr(34)
cEventID= "event_id="& chr(34) & cComputerName &
"&MXEvent.TimeGenerated & chr(34)
```

- The following incident fields in Service Desk are frequently mapped:
  - event\_id (this is a required field, a unique key field)
  - description
  - information
  - solution
  - category
  - impact
  - status
  - priority
  - CI
- The following information can be used when editing the Visual Basic text to change the configuration of event messages going from ManageX to Service Desk. After changing a policy you will need to install it by right-clicking the policy, selecting All Tasks, and then Install.

**Table B-2 ManageX Event String Substitutions**

<b>String</b>	<b>Description</b>
\$(Category)	Category of the triggering Console Message.
\$(CategoryNumber)	Category of the triggering Event Log Message, only the event log.
\$(CategoryString)	Category string of the triggering Console Message.
\$(Computer)	Name of the machine that sent the triggering Message or NT Event Log event.

**Table B-2 ManageX Event String Substitutions**

<b>String</b>	<b>Description</b>
\$(Description)	Contents of the Description field from the triggering Console Message.
\$(EventID)	Contents of the EventID field from the triggering Console Message.
\$(Identifier)	A unique record number from the triggering Event Log message.
\$(OriginalTime)	The time the triggering Event Log item was written or the triggering Console Message was sent.
\$(RawDescription)	The unexpanded description of the triggering message.
\$(ReceiveComputer)	The computer receiving the triggering Console Message.
\$(ReceiveSource)	Application, Security, or System if the triggering event came from the Windows NT Application. Security OpenView Console, or System Log OpenView Console, if the trigger was an OpenView Console Message.
\$(ReceiveTime)	The time the Event Log message was read from the file or the time when the Console Message was received.
\$(Source)	The source of the triggering message.
\$(Type)	The type of triggering message. One of the following numeric values will be returned: 1=Information, 2=Warning, 3=Error, 8=Success, 16=Audit Failure.
\$(User)	The name of the user who sent the triggering message.

**Table B-3 ManageX Event COM Objects**

<b>Field/Syntax</b>	<b>Comment</b>	<b>Type</b>
CATEGORY MxEvent.Category	user-defined message category.	Unsigned Integer (UINT)
CATEGORYSTRING MxEvent.CategoryString	User-defined message category.	String
COMPUTERNAME MxEvent.ComputerName	Machine originating event.	String
SERVICENAME MxEvent.DeviceName	User-defined identifier for device type, such as printer or hub for hardware.	String
EVENTIDENTIFIER MxEvent.EventIdentifier	User-defined integer that uniquely identifies the event.	Unsigned integer (UINT)
MESSAGE MxEvent.Message	Free-form information sent to console operator.	String
SOURCENAME MxEvent.SourceName	User-defined string identifying application	String
TIMEGENERATED MxEvent.TimeGenerated	Date field that defaults to time event was sent; user may override with a specific value.	Date
TYPE MxEvent.Type	User-defined string identifying severity level of the event; typically includes Information, Warning, and Error. Other strings may be defined as appropriate.	String
USER MxEvent.User	User-defined string naming the user responsible for generating the event	String

**Table B-3** ManageX Event COM Objects

Field/Syntax	Comment	Type
HELPURL MxEvent.HelpURL	String pointing to a location with additional information to assist you in dealing with the event	String

### The Event Queue

Configure the queuing tools if you expect to experience event storms. An example for ManageX follows:

#### Example B-1 Configuring the Event Queue for the ManageX Integration

The job queue executable for ManageX will be installed automatically by the installation program:

To use the `addentry.sh` call, modify `sd_event.sh` so that it looks like the following example:

```
cd /opt/OV/SD/event_queuing
/opt/OV/SD/event_queuing/addentry.sh
/opt/OV/bin/OpC/extern_intf/sd_event_queue
"cd/opt/OV/SD/bin;
/opt/OV/bin/OpC/extern_intf/get_ManageX_attributes SD sd
$ManageX_Message|/opt/OV/bin/OpC/extern_intf/sd_eventins.pl"
```



---

## **C** **Integrating with LDAP**

Lightweight Directory Access Protocol (LDAP) defines a standard method for accessing and updating information contained in a directory. This integration makes it possible to import information contained in an LDAP directory to Service Desk.

## Integration Possibilities

Directories are often used to store data related to objects, administrative details for a person, for example. The person object will include additional attributes, for example an e-mail address, phone number and address. Directories are designed so that a user can easily search for information using a variety of criteria.

Data Exchange can be used to connect to a directory using an LDAP server and export data specified in the configurable extractor. The data can then be imported into Service Desk based on the import mapping you have specified. To change data in an LDAP directory you must make the change in that directory and not in Service Desk.

The integration can be set up to work just like any other data exchange batch import. The only difference is that an LDAP connection is made instead of an ODBC connection. The Service Desk integration using LDAP consists of:

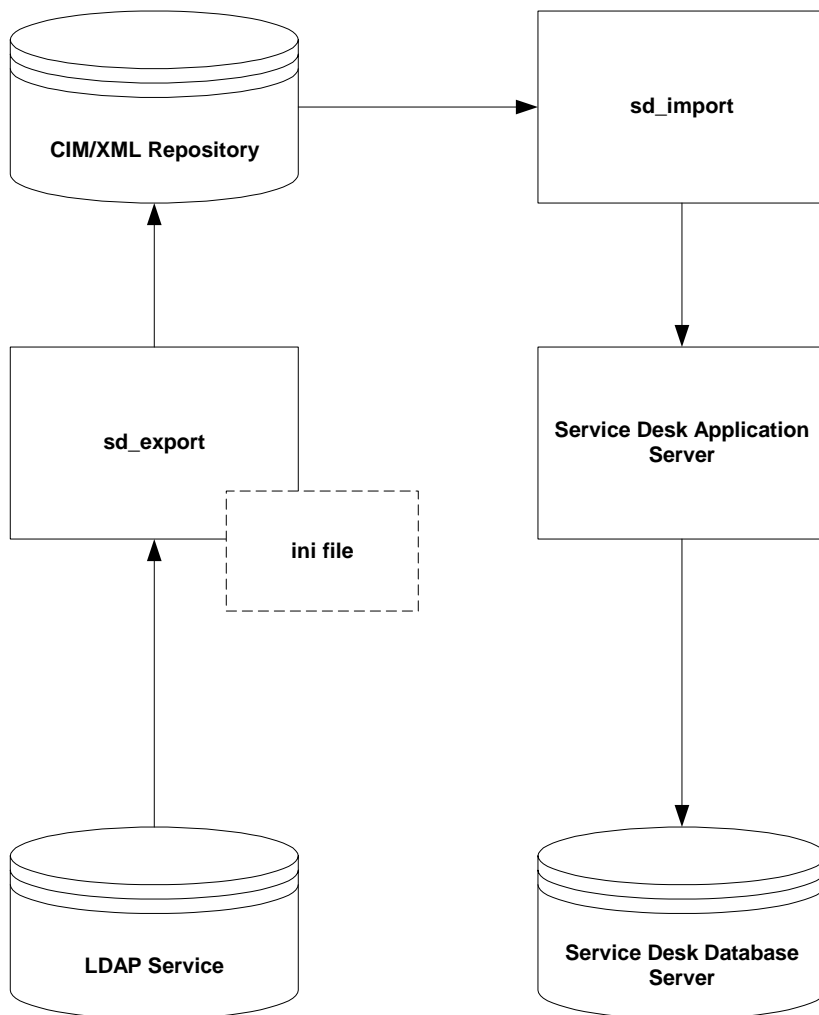
- Ability to import your LDAP data set using the Data Exchange Batch import features. Only string entries can be imported. Binary entries, graphics for example, can not be imported at this time.
- Use of Data Exchange's reconciliation function to keep the Service Desk database synchronized with your LDAP directory.
- Event handling to keep the Service Desk database updated when changes occur in the LDAP directory.

---

## A Technical Description

The following diagram shows the LDAP integration process. The data source and the .ini file are the only two differences between this integration and most other Data Exchange integrations for importing batch data:

**Figure C-1**      **The LDAP Integration**



## Installation

On the Service Desk application server you will need to install Integrations from the Service Desk 4.0 CD-ROM. Refer to the *HP OpenView Service Desk: Installation Guide* for additional installation information.

If you select the option of doing a custom installation of the integrations, select Data Exchange and LDAP as a minimum. An example configuration file LDAP.ini is provided, see “Configuring the LDAP.ini File” on page 207 for more information.

---

### NOTE

This integration requires an LDAP server.

---

## LDAP Server Account

Before installing the integration on the LDAP server, create an account for the integration with Service Desk:

1. Make the account a non-user interface account.
2. Assign each account a helpdesk user role, as a minimum.

Refer to the *HP OpenView Service Desk: Administrator's Guide* for additional information about creating accounts in Service Desk.

---

## Configuration

This section explains the configuration steps that need to be performed before you can exchange data with your LDAP directory.

### SD\_Export

The `sd_export` program can export data from both an LDAP server or ODBC. The `.ini` files used for the two types of connections are different.

### Configuring the LDAP.ini File

The configuration file used for the LDAP integration is different than other Data Exchange configuration files which are based on an ODBC connection. You need to specify that you are using this file for LDAP in the first section [CONNECTION]. If this section is left blank the `sd_export` program will assume it needs to export from ODBC.

---

#### NOTE

The extraction configuration wizard provided for Data Exchange is not compatible with the `LDAP.ini` files. The wizard can only be used with ODBC based `.ini` files at this time.

---

An example `LDAP.ini` file follows with an explanation of the different sections:

```
[CONNECTION]
TYPE=LDAP
[LDAP]
SERVER=your_complete_server_name
PORT=389
PRINCIPAL=
AUTHENTICATION=
CREDENTIALS=
```

## Integrating with LDAP Configuration

```
[ SYSTEM ]
LOG=TRUE
XML=TRUE
LOG_FILE=Persons.log
OUTPUT_FILE=Persons.txt
XML_OUTPUT_FILE=Persons.xml
APPLICATION_NAME=LDAP_Persons

[ CLASSES ]
NAME=CL_WORKGROUP_EMP, CL_CODE_WORKGROUP
-----
-- WORKGROUP CODES --
-----

[ CL_CODE_WORKGROUP ]
SOURCE=ou=Groups, o=LDAPPersons
ATT=[CN], [UniqueMember]
SEARCHSCOPE=SUBTREE_SCOPE

[ CL_WORKGROUP_EMP ]
ATT=[CN], [SN], [MAIL], [TELEPHONENUMBER]
PARENT=CL_CODE_WORKGROUP
PARENT_RELATION=dn=[UniqueMember]
```

### Explaining the LDAP .ini File

The following table describes the configuration file:

**Table C-1**

**LDAP Configuration File**

<b>CONNECTION Section</b>	<b>Default</b>	<b>Description</b>
TYPE	ODBC	Mandatory. Enter LDAP. Selects the data source to be used.

LDAP Section	Default	Description
SERVER		Mandatory. Enter the DNS IP of a fully qualified DNS IP.
PORT	389	TCP/IP Port
AUTHENTICATION		Enter the security level you want to use. No Authentication is the lowest level of security possible. Refer to your LDAP documentation for information about the appropriate security level.
PRINCIPAL		Enter the identity of the principal for the service. Refer to your LDAP documentation for more information.
CREDENTIALS		Enter the credentials of the principal to use for the service. Refer to your LDAP documentation for more information.

SYSTEM	Default	Description
LOG	false	Generate log file, true or false.
XML	FALSE	Generate an XML file, true or false.
LOG_FILE	Extracto r.log	Name of log file
XML_OUTPUT_FILE		The name to be given your exported XML file.
APPLICATION_NAME		The name of your directory application. The name is used in the XML file.
ENCODING	UTF-8	Refers to the character set used.

<b>SYSTEM</b>	<b>Default</b>	<b>Description</b>
LANGUAGE	GB	Select the language for error messages. A corresponding message file must exist.

<b>CLASSES Section</b>	<b>Default</b>	<b>Description</b>
NAME	No default	Mandatory. Enter the names of the classes you want to export.
SOURCE	No default	Mandatory. Enter the source (location) of the classes.
SEARCHSCOPE	ONELEVEL_SCOPE	Mandatory. The start point of a search is defined by the SOURCE tag. Because LDAP is hierarchical you can enter the depth of the search (the SEARCHSCOPE) by selecting one of the following values: OBJECT_SCOPE to return one or zero elements, situated at the base point. ONELEVEL_SCOPE, to return all elements situated at the base point. SUBTREE_SCOPE, to return all elements, including those that appear in subtrees. Refer to your LDAP documentation for more information.
ATT	No default	Mandatory. Enter the attributes you want to export to the XML file. These attributes will be reported in the XML file.
COLUMNS	No default	Mandatory. Define the attributes that you want queried from the LDAP system.



<b>CLASSES Section</b>	<b>Default</b>	<b>Description</b>
PARENT	No default	Enter the class name of the parent.
PARENT_RELATION	No default	Enter the name of the parent relation.

---

**NOTE**

Unique attributes may not be visible when using the XML viewer in Data Exchange. If you open the XML file in Notepad, all unique attributes will be visible.

---

### **XML File**

The resulting XML file that is created by the data exchange process:

```
<?xml version="1.0" encoding='ISO-8859-1'?>
<!DOCTYPE CIM SYSTEM "file:CIM_DTD_V20.dtd"
[<!ENTITY lt      "&#38;#60;">
  <!ENTITY gt      "&#62;">
  <!ENTITY amp     "&#38;#38;">
  <!ENTITY apos    "&#39;">
  <!ENTITY quot    "&#34;"> ]>
<CIM CIMVERSION="2.0" DTDVERSION="2.2">
<DECLARATION>
<DECLGROUP>
<VALUE.OBJECT>
<INSTANCE CLASSNAME="Header">
  <PROPERTY NAME="Date" TYPE="string">
    <VALUE>08/01/2001</VALUE>
  </PROPERTY>
  <PROPERTY NAME="Application" TYPE="string">
```

## Integrating with LDAP Configuration

```
<VALUE>LDAP_Persons</VALUE>
</PROPERTY>
</INSTANCE>
</VALUE.OBJECT>
<VALUE.OBJECT>
<INSTANCE CLASSNAME="CL_CODE_WORKGROUP">
  <PROPERTY NAME="ID" TYPE="string">
    <VALUE>1</VALUE>
  </PROPERTY>

  <PROPERTY NAME="CN" TYPE="string">
    <VALUE>Directory Administrators</VALUE>
  </PROPERTY>

  <PROPERTY.ARRAY NAME="UNIQUEMEMBER" TYPE="string">
    <VALUE.ARRAY>
      <VALUE>cn=Rosanna Lee, ou=People, o=LDAPPersons</VALUE>
      <VALUE>cn=Scott Seligman, ou=People,
o=LDAPPersons</VALUE>
      <VALUE>cn=Jon Ruiz, ou=People, o=LDAPPersons</VALUE>
      <VALUE>cn=Vinnie Ryan, ou=People, o=LDAPPersons</VALUE>
    </VALUE.ARRAY>
  </PROPERTY.ARRAY>

</INSTANCE>
</VALUE.OBJECT>

<VALUE.OBJECT>
<INSTANCE CLASSNAME="CL_WORKGROUP_EMP">
```

```
<PROPERTY NAME="ID" TYPE="string">  
  <VALUE>2</VALUE>  
</PROPERTY>
```

```
<PROPERTY NAME="CN" TYPE="string">  
  <VALUE>Rosanna Lee</VALUE>  
</PROPERTY>
```

```
<PROPERTY NAME="SN" TYPE="string">  
  <VALUE>Lee</VALUE>  
</PROPERTY>
```

```
<PROPERTY NAME="MAIL" TYPE="string">  
  <VALUE>Rosanna.Lee@LDAPPersons.com</VALUE>  
</PROPERTY>
```

```
<PROPERTY NAME="TELEPHONENUMBER" TYPE="string">  
  <VALUE>+1 408 555 1856</VALUE>  
</PROPERTY>
```

```
<PROPERTY.REFERENCE NAME="Parent ">  
  <VALUE.REFERENCE>  
    <INSTANCENAME CLASSNAME="CL_CODE_WORKGROUP">  
      <KEYBINDING NAME="ID">  
        <KEYVALUE VALUETYPE="string">1</KEYVALUE>  
      </KEYBINDING>  
    </INSTANCENAME>  
  </VALUE.REFERENCE>
```

## Integrating with LDAP Configuration

```
</PROPERTY.REFERENCE>

</INSTANCE>
</VALUE.OBJECT>

<VALUE.OBJECT>
<INSTANCE CLASSNAME="CL_WORKGROUPEMP">
  <PROPERTY NAME="ID" TYPE="string">
    <VALUE>3</VALUE>
  </PROPERTY>

  <PROPERTY NAME="CN" TYPE="string">
    <VALUE>Scott Seligman</VALUE>
  </PROPERTY>

  <PROPERTY NAME="SN" TYPE="string">
    <VALUE>Seligman</VALUE>
  </PROPERTY>

  <PROPERTY NAME="MAIL" TYPE="string">
    <VALUE>Scott.Seligman@LDAPPersons.com</VALUE>
  </PROPERTY>

  <PROPERTY NAME="TELEPHONENUMBER" TYPE="string">
    <VALUE>+1 408 555 5252</VALUE>
  </PROPERTY>

  <PROPERTY.REFERENCE NAME="Parent">
```

```
<VALUE.REFERENCE>  
<INSTANCENAME CLASSNAME="CL_CODE_WORKGROUP">  
  <KEYBINDING NAME="ID">  
    <KEYVALUE VALUETYPE="string">1</KEYVALUE>  
  </KEYBINDING>  
</INSTANCENAME>  
</VALUE.REFERENCE>  
</PROPERTY.REFERENCE>
```

## Import Mapping

You will need create a new Import Mapping, this is easiest to do after you have configured the .ini file and exported the data to an XML file successfully. The property names and values in the XML file need to be mapped to class names, attributes and values in Service Desk. Chapter 3, “Import Mapping,” on page 93 provides detailed instructions for import mapping.

## Importing your LDAP Directory Information

The LDAP integration uses the Data Exchange exporting and importing tools for batch data. You can create a Data Exchange task and run the process just like any other data exchange task for batch importing. For information on how to create a task for batch importing, see Chapter 4, “Importing Data in Batches,” on page 117.

Integrating with LDAP  
**Configuration**

---

## **D** **Integrating with Radia**

This integration makes it possible to import Radia data using Data Exchange features in Service Desk. Novadigm Radia provides enterprise configuration management, software distribution and inventory collection. The integration is explained in the following sections.

## Installation

On the Service Desk application server you will need to install Integrations from the Service Desk 4.0 CD-ROM. Refer to the *HP OpenView Service Desk: Installation Guide* for additional installation information.

An example configuration file `Radia.ini` is provided, see “Configuring the Radia.ini File” on page 226 for more information. An example import mapping is also provided, see “Import Mapping” on page 229. Both the import mapping and configuration file can be modified.

You will need to create an account in Service Desk for the Radia integration, refer to the Online Help or the *Administrator's Guide* for details.

---

### NOTE

The SQL Server Client Utilities must be installed on the machine that is running Data Exchange to connect with the machine running SQL Server.

---

---

### NOTE

You must execute the command `ODBCCMPT SD_EXPORT.EXE` on the Client machine that you will use to run the Data Exchange task to export data. You only need to run the command once. This step is necessary to improve the importing of string values.

---



## Configuration

This section explains the configuration steps that need to be performed before you can exchange data with Radia.

### Configuring an ODBC Data Source

Before you can export data from the Radia Inventory Manager database, you must configure an ODBC data source.

The specifics of configuring an ODBC data source for a particular database system are described in that database system's documentation. This chapter provides an example for a SQL Server data source.

The ODBC data source must be defined on the computer which is running Data Exchange, the database tables may be located on any machine accessible to that computer. To configure the ODBC data source you will need to install the appropriate driver. Drivers are normally provided with the database system, or may be obtained separately from the database system vendor.

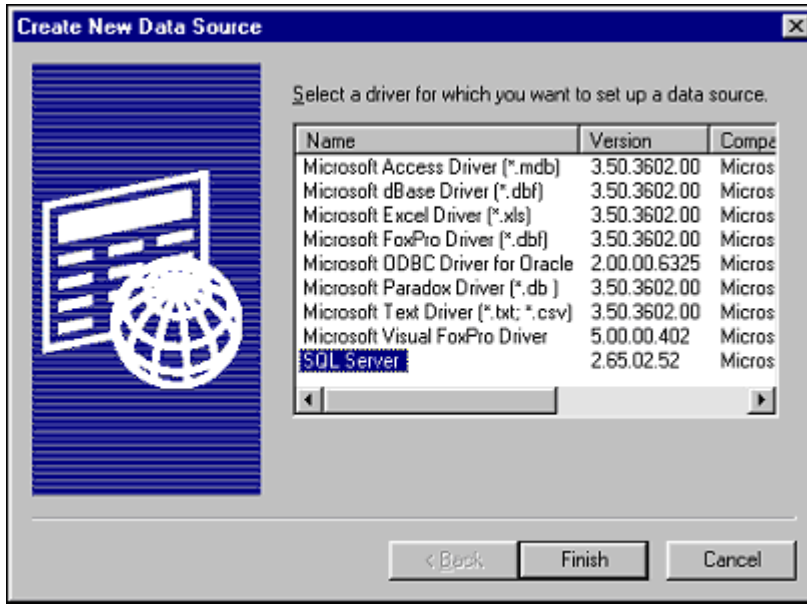
### Creating the SQL Server ODBC Data Source

The following is an example of how you can setup a SQL Server ODBC data source. Variations may occur due to differences in operating systems.

To create a SQL Server ODBC Data Source:

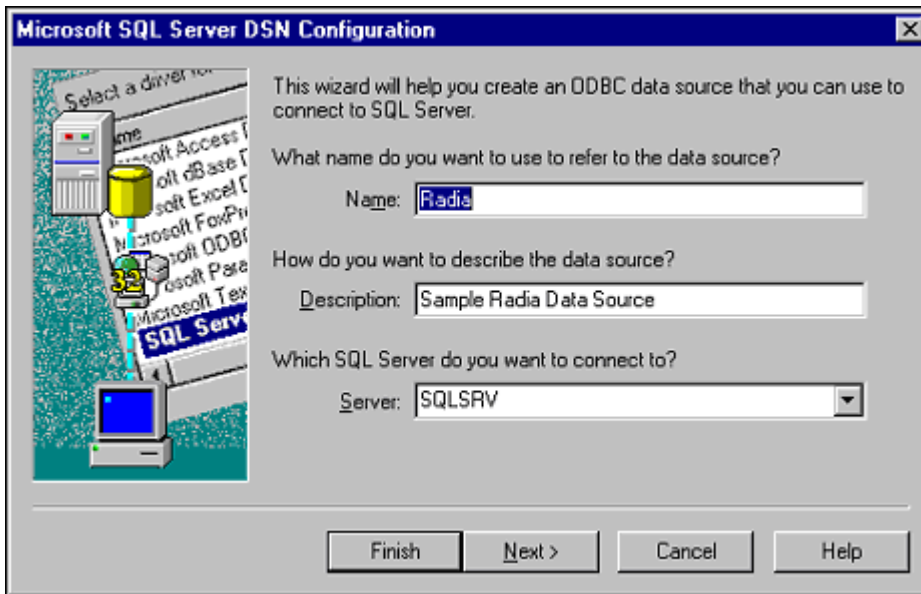
1. Open the ODBC Data Source Administrator, and open the System DSN tab. Select SQL Server Driver, then click Finish:

**Figure D-1 Create New Data Source for SQL Server dialog box**



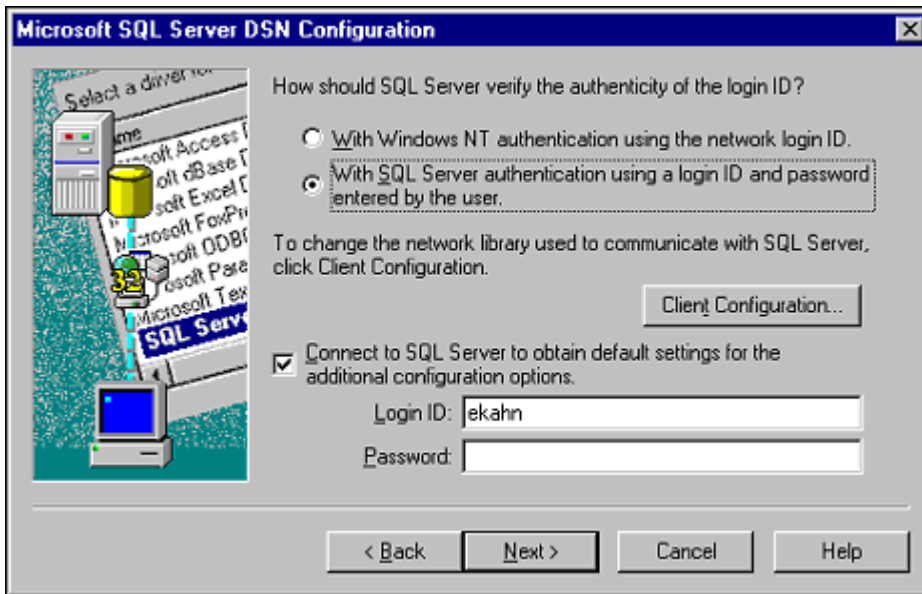
2. This invokes a wizard which leads you through the process of defining the ODBC data source. In the Name field, enter a name for the data source definition.

**Figure D-2** SQL Server DSN Configuration wizard



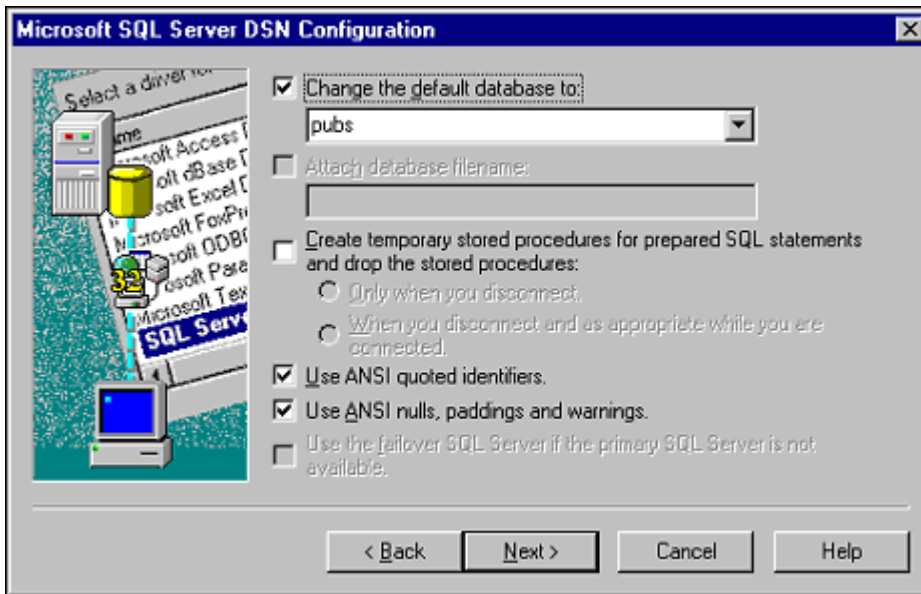
3. The Description field is free text. Enter a description that denotes the purpose or use of the data source definition.
4. In the Server field, select the SQL Server hosting the Radia inventory data, from the Server drop-down list. Click Next to continue and the following dialog box is displayed:

**Figure D-3** SQL Server DSN Configuration wizard



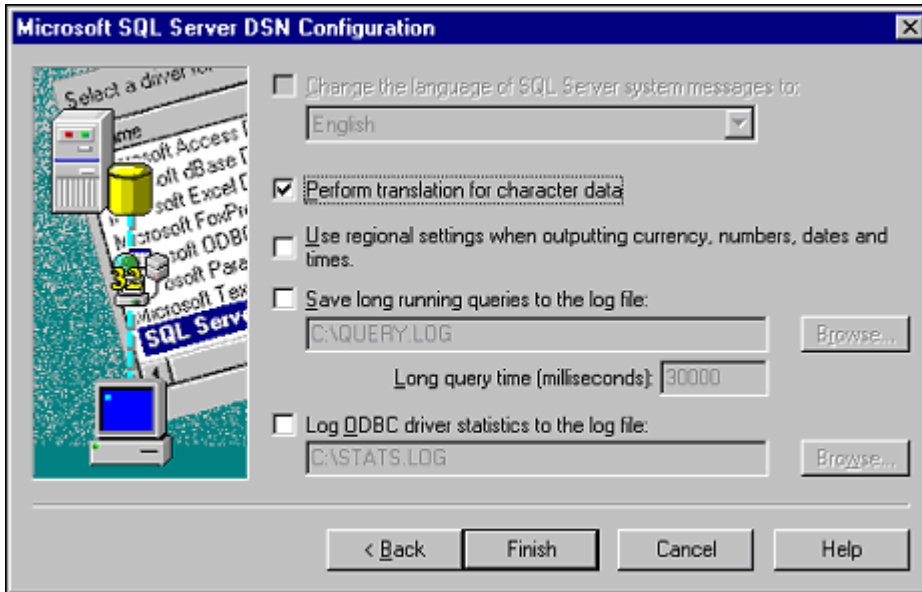
5. Enter the SQL Server's I/P address and port number in the Connection parameters area. Because we've equated SQLSRV with the correct I/P address in the HOSTS file on this machine, we can refer to the SQL Server's I/P address using this name.
6. Click OK to return to the previous panel. Click Next on that panel to proceed:

**Figure D-4** SQL Server DSN Configuration wizard



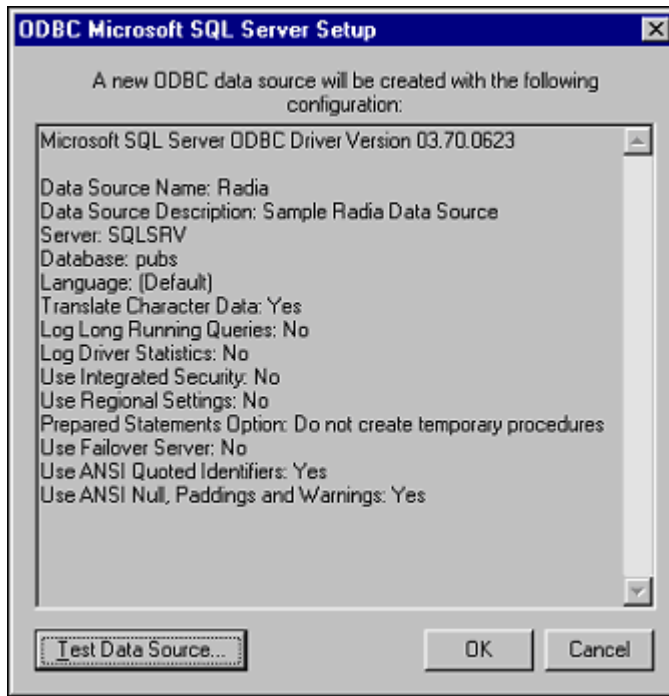
7. Select or type the name of the desired SQL Server database into the Change the default database to combo box.
8. Set the remaining controls in this panel per the requirements of the selected database (see your SQL Server administrator for the necessary information). Click Next to proceed:

**Figure D-5** SQL Server DSN Configuration wizard



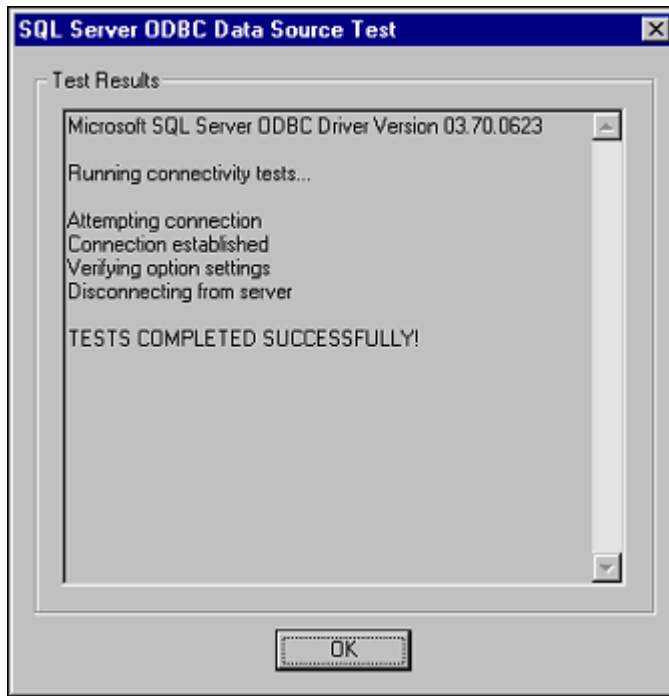
9. Set the controls in this panel per the requirements of the selected database (see your SQL Server administrator for the necessary information). Click **Finish** to proceed:

**Figure D-6 ODBC SQL Server Setup dialog box**



10. Click Test Data Source to perform a test of your ODBC data source definition. If you have correctly entered the necessary information to define the ODBC data source the system will display a response similar to the following:

**Figure D-7** SQL Server ODBC Data Source Test dialog box



11. Click the **OK** button to save the data source definition and close the dialog box.

## Configuring the Radia.ini File

An example configuration file is provided for the Radia integration. You can modify it by opening it with text editor.

The .ini file provided for the Radia integration with Service Desk is similar in most respects to the other example.ini files. See “Configuring the Extractor” on page 45 for basic information about the keywords used.

An additional keyword in the [SYSTEM] section is included to preclude problems from occurring when exporting Parent-Child relations with SQL Server:

```
USE_MULTIPLE_CONNECTIONS=TRUE
```

The REPLACE function is added to remove certain characters from the



exported data. Service Desk does not allow for specific characters to be part of a search code, so the REPLACE command in the following example is used to remove spaces and underscores:

```
COLUMNS=REPLACE(REPLACE('RADIA' + [DEVICECONFIG].[OS] +
[DEVICECONFIG].[OS_LEVEL], ' ', ''), '_','') as [OSSEARCHCODE], \
```

The + command is included to concatenate strings. The resulting search code is a concatenation of the string RADIA, the OS and the OS\_LEVEL of an item exported from the Radia DEVICECONFIG table. You can see an example of this in the Radia.ini file that follows:

```
[DSN]
NAME=Radia DSN sqlserver
USR=
PWD=

[SYSTEM]
LOG=TRUE
XML=TRUE
LOG_FILE=RADIA.log
XML_OUTPUT_FILE=RADIA.xml
APPLICATION_NAME=RADIA
USE_MULTIPLE_CONNECTIONS=TRUE

[CLASSES]
NAME=ADMIN,OS,DISK,MOUSE,CPU,VIDEO
```

-- The condition is just an example for setting a filter on the machines to be exported

```
[ADMIN]
SOURCE=DEVICECONFIG
ATT=[ADMINSEARCHCODE], \
    [MACHID], \
    [MACHINENAME], \
    [IPADDRESS]
COLUMNS=REPLACE('RADIA' + [DEVICECONFIG].[DEVICE_ID], ' ', '')
as [ADMINSEARCHCODE], \
    [DEVICECONFIG].[DEVICE_ID] as [MACHID], \
    [DEVICECONFIG].[PERSON] as [MACHINENAME], \
    [DEVICECONFIG].[IPADDR] as [IPADDRESS]
CONDITION=[DEVICECONFIG].[DEVICE_ID] is NOT NULL
LOADTABLE=TRUE

[OS]
SOURCE=DEVICECONFIG
```

## Integrating with Radia Configuration

```
ATT=[OSSEARCHCODE], \
    [OSTYPE], \
    [OSVERSION]
COLUMNS=REPLACE(REPLACE('RADIA' + [DEVICECONFIG].[OS] +
[DEVICECONFIG].[OS_LEVEL], ' ', ''), '_ ', '') as [OSSEARCHCODE], \
    [DEVICECONFIG].[DEVICE_ID] as [MACHID], \
    [DEVICECONFIG].[OS] + ' ' + [DEVICECONFIG].[OS_LEVEL]
as [OSTYPE], \
    [DEVICECONFIG].[OS_LEVEL] as [OSVERSION]
CONDITION=[DEVICECONFIG].[OS] is NOT NULL
LOADTABLE=TRUE
PARENT=ADMIN
PARENT_RELATION=[DEVICECONFIG].[DEVICE_ID]=MACHID

[CPU]
SOURCE=DEVICECONFIG
ATT=[CPUSEARCHCODE], \
    [CLOCKSPEED], \
    [CPUTYPE]
COLUMNS=REPLACE('RADIA' + [DEVICECONFIG].[CPU_SPEED] +
[DEVICECONFIG].[CPU], ' ', '') as [CPUSEARCHCODE], \
    [DEVICECONFIG].[DEVICE_ID] as [MACHID], \
    [DEVICECONFIG].[CPU_SPEED] as [CLOCKSPEED], \
    [DEVICECONFIG].[CPU] as [CPUTYPE]
CONDITION=[DEVICECONFIG].[CPU_SPEED] is NOT NULL and \
    [DEVICECONFIG].[CPU] is NOT NULL
LOADTABLE=TRUE
PARENT=ADMIN
PARENT_RELATION=[DEVICECONFIG].[DEVICE_ID]=MACHID

[DISK]
SOURCE=DEVICECONFIG
ATT=[DISKSEARCHCODE], \
    [DISKTYPE], \
    [DISKCAPACITY]
COLUMNS=[DEVICECONFIG].[DEVICE_ID] as [MACHID], \
    REPLACE('RADIA' + [DEVICECONFIG].[DEVICE_ID] +
[DEVICECONFIG].[SYSDRV], ' ', '') as [DISKSEARCHCODE], \
    [DEVICECONFIG].[DEVICE_ID] + ' ' +
[DEVICECONFIG].[SYSDRV] as [DISKTYPE], \
    'Total: ' + STR(DEVICECONFIG.SYSDRV_TOTAL) + ' \ Free:
' + STR([DEVICECONFIG].[SYSDRV_FREE]) as [DISKCAPACITY]
CONDITION=[DEVICECONFIG].[SYSDRV] is NOT NULL
LOADTABLE=TRUE
PARENT=ADMIN
PARENT_RELATION=[DEVICECONFIG].[DEVICE_ID]=MACHID
```

```
[MOUSE]
SOURCE=DEVICECONFIG
ATT=[MOUSESEARCHCODE], \
    [MOUSETYPE]
COLUMNS=REPLACE('RADIO' + [DEVICECONFIG].[MOUSE], ' ', '') as
[MOUSESEARCHCODE], \
    [DEVICECONFIG].[DEVICE_ID] as [MACHID], \
    [DEVICECONFIG].[MOUSE] as [MOUSETYPE]
CONDITION=[DEVICECONFIG].[MOUSE] is NOT NULL
LOADTABLE=TRUE
PARENT=ADMIN
PARENT_RELATION=[DEVICECONFIG].[DEVICE_ID]=[MACHID]

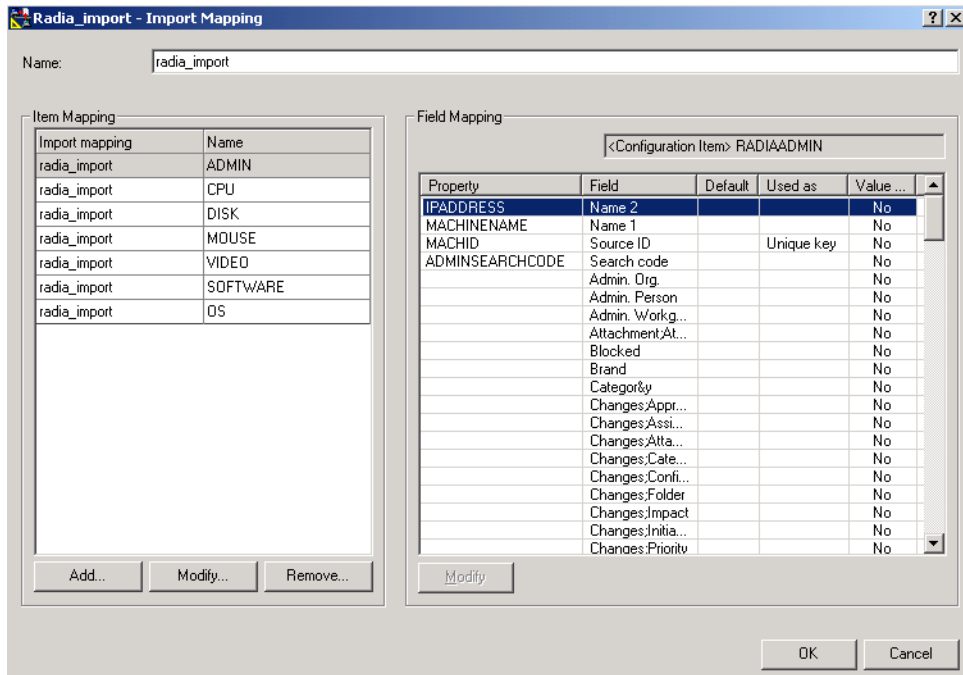
[VIDEO]
SOURCE=DEVICECONFIG
ATT=[VIDESEARCHCODE], \
    [VIDEOCARD]
COLUMNS=REPLACE('RADIO' + [DEVICECONFIG].[VIDEO], ' ', '') as
[VIDESEARCHCODE], \
    [DEVICECONFIG].[DEVICE_ID] as [MACHID], \
    [DEVICECONFIG].[VIDEO] as [VIDEOCARD]
CONDITION=[DEVICECONFIG].[VIDEO] is NOT NULL
LOADTABLE=TRUE
PARENT=ADMIN
PARENT_RELATION=[DEVICECONFIG].[DEVICE_ID]=[MACHID]
```

## Import Mapping

An example of the Import Mapping is provided for integrating with Radia. If you want to modify the import mapping file, Chapter 3, “Import Mapping,” on page 93 provides detailed instructions for import mapping.

The default import mapping looks similar to the following:

**Figure D-8 Radia Default Import Mapping**



## Exporting and Importing Radia Data

You can create a Data Exchange task and run the process just like any other data exchange task for batch importing. For information on how to create a task for batch importing, see Chapter 4, “Importing Data in Batches,” on page 117.

Before running the export task, from the DOS prompt execute the command `ODBCMPT SD_EXPORT . EXE` on the Client machine that you will use to run the Data Exchange task to export data. This step only needs to be performed once and is necessary to improve the importing of string values.

---

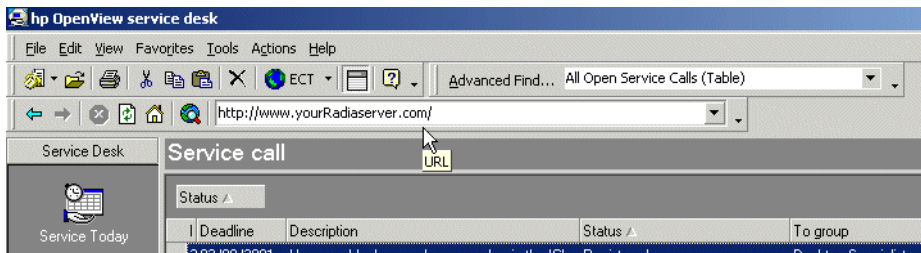
## Opening Radia Browser from Service Desk

You can use the Web toolbar in Service Desk to open the Radia Inventory Manager browser while you are working in Service Desk.

To activate the Web application in Service Desk:

1. From the View menu in Service Desk, select Toolbars, and then Web.
2. Enter the URL for your Radia integration server in the Web address field of the Web toolbar:

**Figure D-9** Web Application with Radia



3. You can now use the features in the Radia browser to query the Radia Inventory database.

For more information refer to the Radia Inventory Manager documentation.

Integrating with Radia  
**Opening Radia Browser from Service Desk**

---

# **E** **Examples**

The following sections contain examples for how to use an example configuration file, how to import data from an MS Excel spreadsheet, from an ASCII text file, and how to import relations.

## Using an Example Configuration File

You will need to adapt the configuration files to fit your environment. The configuration files contains the knowledge about what data to extract and about the source you will extract data from.

If you have added or changed the classes or fields in your external application, you will need to modify the export mapping in the example configurable extractor and the import mapping in Service Desk. In some cases it may also be necessary to use the customizing features in Service Desk to create new fields to import your external fields to. For example, a field present in your application may not be included in the configuration file. In that case you will need to adjust the NNM.ini file and the NNM import mapping.

The following example will be used to explain the different parts of the configuration file:

- Step 1.** Copy an existing configuration file and save it with a new name. For this example a portion of the NNM6 example configuration file is used:

```
[DSN]
NAME=nnm6
USR=testingdb
PWD=testingdb

[SYSTEM]
LOG=TRUE
XML=TRUE
LOG_FILE=C:\temp\NNM6.log
OUTPUT_FILE=c:\temp\NNM6.txt
XML_OUTPUT_FILE=c:\temp\NNM6.xml
APPLICATION_NAME=NNM6
ENCODING=ISO-8859-1
[CLASSES]
NAME=NETWORK, SEGMENT

[NETWORK]
SOURCE=[NNM_NETWORKS], [NNM_OBJECTS]
ATT=[NAME], [INTERFACE_COUNT], [NNM_ID]
COLUMNS=[NNM_NETWORKS].[IP_NETWORK_NAME] AS
[NAME], [NNM_NETWORKS].[TOPM_INTERFACE_CNT] AS
[INTERFACE_COUNT], [NNM_OBJECTS].[OVW_ID] AS [NNM_ID]
CONDITION=[NNM_NETWORKS].[TOPO_ID]=[NNM_OBJECTS].[TOPO_ID]
```



```
[ SEGMENT ]
SOURCE=[NNM_SEGMENTS], [NNM_OBJECTS]
PARENT=NETWORK
PARENT_RELATION=[NNM_SEGMENTS].[NET_ID]=[NNM_ID]
ATT=[NAME],[NNM_ID]
COLUMNS=[NNM_SEGMENTS].[SEGMENT_NAME] AS
[NAME],[NNM_SEGMENTS].[NET_ID] as
[NET_ID],[NNM_OBJECTS].[OVW_ID] AS [NNM_ID]
CONDITION=[NNM_SEGMENTS].[TOPO_ID]=[NNM_OBJECTS].[TOPO_ID]
```

**Step 2.** Enter the data source (DSN) information.

1. Enter the name of the data source you will export data from. For this example nnm6 is being used as the data source. You must use this same name when setting up your ODBC connection.
2. Enter the name this integration can used to login to that data source.
3. Enter the password that goes with the login name.

**Step 3.** Specify the system settings.

1. The first four headers are used to specify if a log file will be created (LOG=TRUE), if the output file is of the XML type (XML=TRUE)
2. Enter the locations that you want the LOG FILE, OUTPUT\_FILE and XML\_OUTPUT\_FILE placed after creation.
3. Enter the APPLICATION\_NAME for the application you will be exporting data from. For this example the data source application is NNM6.
4. In the Encoding line, enter the language type you will be viewing the configuration file in. ISO-8859-1 is entered in the example so that it can be viewed with a European language character set. There are numerous other character sets available to support different languages.

**Step 4.** Define the classes that will be exported.

1. Enter the names of all classes you will export. You can decide what the class names are, these class names will be used later with the import mapping. In the example the class names are NETWORK, and SEGMENT.

## Examples

### Using an Example Configuration File

2. Enter a CLASS section for each class you will export.
3. Under each class section enter the SOURCE database the class data needs to be exported from.
4. For each class you must define attributes in the ATT section. The ATT section is used to identify the item and defines where in Service Desk the data must go. The attributes can literally be the same as the columns or aliases. The attributes will be captured from the columns that are written in the COLUMNS section.
5. Enter the COLUMNS you want to select with an alias [columnname1] AS [alias 1]. Columns that appear in the PARENT\_RELATION have to be put into the column list when tables are cached [LOADTABLE=TRUE], because the extractor uses this information to find all children for a particular parent in memory. If an alias is not specified the column name will be used as the alias name.
6. Enter the CONDITION or selection criteria for the records to be retrieved.

For additional information on configuring the extractor, see the section “Configuring the Extractor” on page 45.

## Importing From an MS Excel Spreadsheet

This example demonstrates how to import data from an MS Excel spreadsheet. The examples in this section were produced using a Microsoft Office 2000 version of Excel:

**Step 1.** Adjust the Excel spreadsheet

1. Add a heading record.
2. Select all data from the Excel sheet.
3. Enter a name for the selection in the upper left corner. The data selected for this example was given the name; PHONELBL.

**Figure E-1 Adjust the Excel Spreadsheet**

	A	B	C	D
1	LASTNAME	FIRSTNAME	JOBTITLE	SEARCHCODE
2	Galster	Steven	Director of OEM & ISV Sales	SGALSTER
3	Hinman	Dave	VP Business Development	DHINMAN
4	Myers	Stephen	Senior Sales Engineer	SMYERS
5	Barrea	Donna	Sales Administrator	DBARREA
6	Bentz	Ray	Director of ATG	BENTZ
7	Bergenholtz	Erik	ATG Engineer	BERGEN
8	Bergman	Barry	Sr. Director of Sales	BBERGMAN
9	Booth	David	Director of Training	BOOTH
10	Byrd	Megan	Inside Sales Rep-BD	MBYRD
11	Campana	Sal	ATG	SCAMPANA
12	Dondero	Robert	Training	DONDERO
13	Druss	Byron	Worldwide Sales Manger, Enterprise Services	BYRON

4. Save the Excel spreadsheet. The example is saved as `phone.xls`.

**Step 2.** Create an ODBC Data Source for the Excel spreadsheet

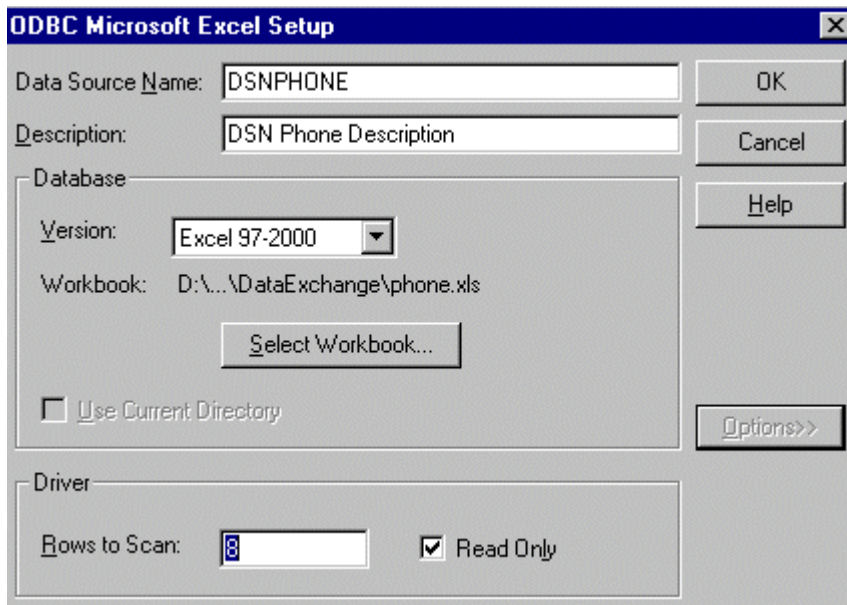
1. Start the ODBC Data Sources from the Windows Control Panel.
2. Select the `System DSN` tab and click `Add`.

## Examples

### Importing From an MS Excel Spreadsheet

3. Select the driver you want to use. **Microsoft Excel Driver (\*.xls) version 4.00.4202.00** was used for this example, then click **Finish**.
4. In the ODBC Microsoft Excel Setup dialog box enter a **Description**, then use the **Select Workbook** button to select the spreadsheet you selected in Step 1. Click **OK** and the **Data Source Name** field will automatically be filled. In this case it becomes **DSNPHONE**.

**Figure E-2 ODBC Microsoft Excel Setup dialog box**



#### **Step 3.** Configure the extractor.

1. Open one of the example configurable extractor provided with Service Desk, `sd_event.ini`.
2. Create a new `.ini` file. In this example the new `.ini` file is named `phoneconf.ini`.
3. Add the lines in the example so that they match your environment, and include the data you want to extract from the Excel file as follows:

```
[ DSN ]
NAME=DSNPHONE
USR=
PWD=

[ SYSTEM ]
LOG=TRUE
XML=TRUE
LOG_FILE=\phone.log
OUTPUT_FILE=\phone.xls
XML_OUTPUT_FILE=\phone.xml
APPLICATION_NAME=PHONELIST
ENCODING=ISO-8859-1

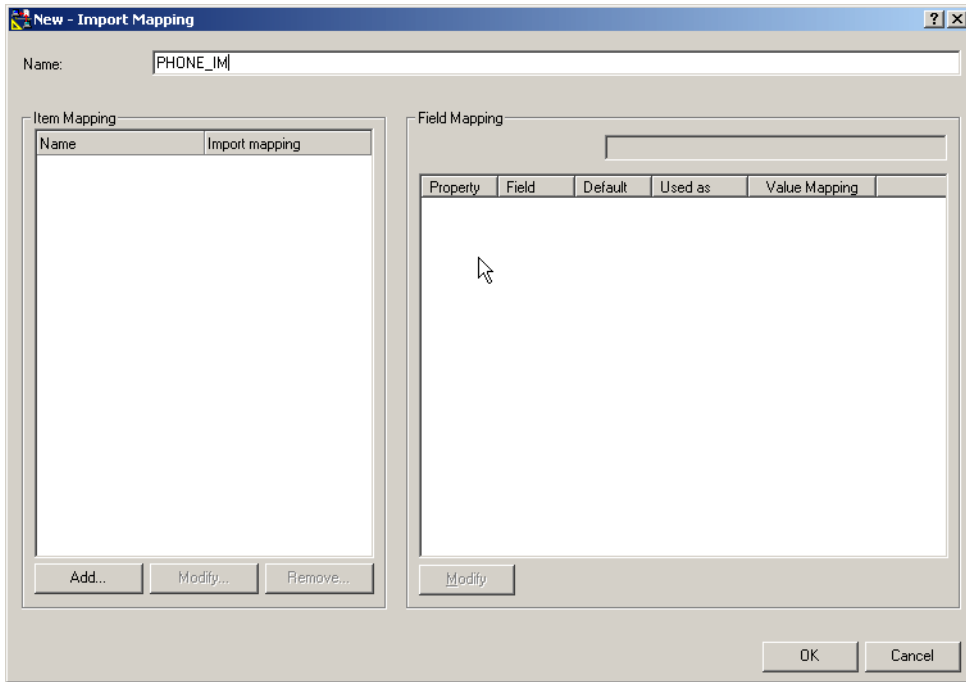
[ CLASSES ]
NAME=CLASSPHONE

[ CLASSPHONE ]
SOURCE=PHONELBL
ATT=[ LASTNAME ], [ FIRSTNAME ], [ JOBTITLE ], [ SEARCHCODE ]
COLUMNS=[ LASTNAME ]. [ FIRSTNAME ], [ JOBTITLE ], [ SEARCHCODE ]
LOADTABLE=TRUE
```

**Step 4.** Create an import mapping.

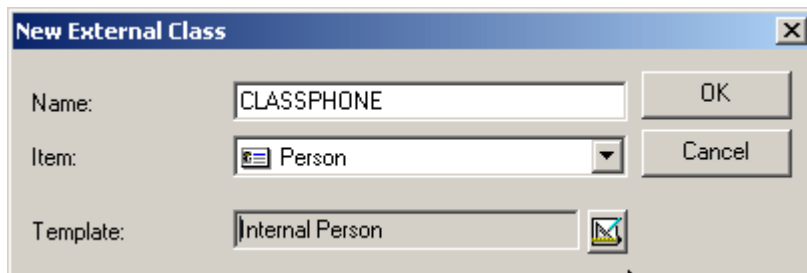
1. From the **Tools** menu in **Service Desk** select **System** and then open the **Data** folder. Double-click **Import Mapping**. Right-click to create a new import mapping:

**Figure E-3** New Import Mapping dialog box



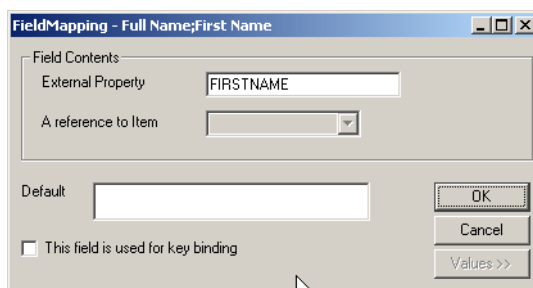
2. Give the import mapping a name, for this example it is **PHONE\_IM**.
3. Create a New External Class for the Excel sheet. In this example it is **CLASSPHONE**. The mapping you create is dependent upon the type of information you are importing, because the example contains personnel information it is mapped to the `Person` item and the `Internal Person` template as follows:

**Figure E-4** New External Class dialog box



4. Click OK. The attributes assigned to the template and item selected for the class will be shown in the lower portion of the New Import Mapping dialog box.
5. Double-click the attributes you want to map to rows and columns in the Excel spreadsheet. In the Field Mapping dialog box you can enter the External Property name that corresponds to the Service Desk attribute you selected:

**Figure E-5 Mapping Attributes**



---

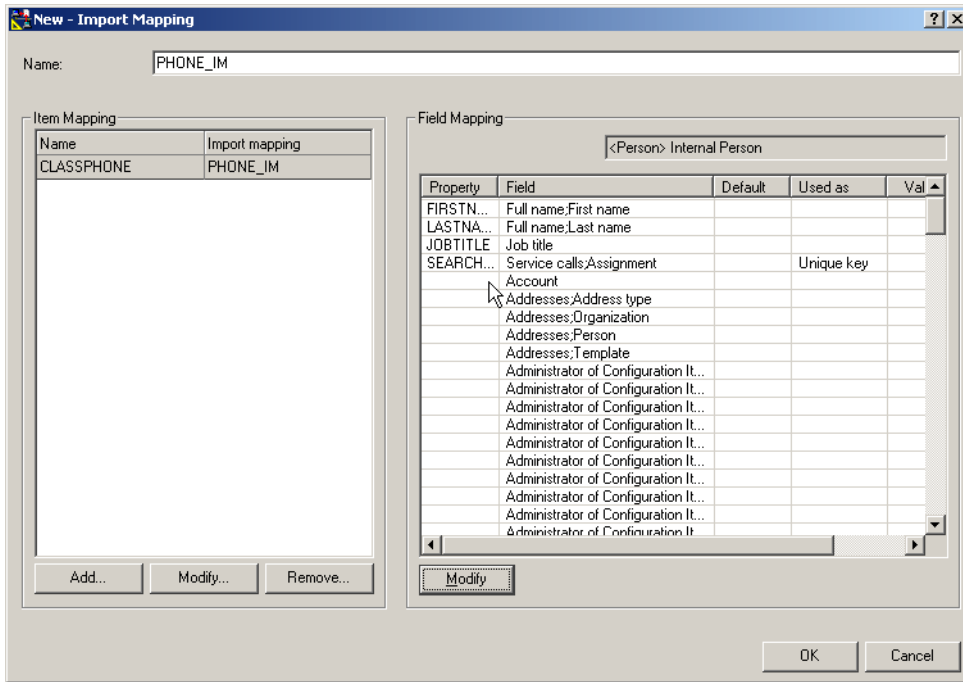
**NOTE**

You must have at least one unique attribute designated as the binding key, this is usually the search code or another unique key.

---

6. Click OK to save your mapping and return to the Import Mapping dialog box. The following dialog box shows the completed import mapping for this example:

**Figure E-6 Complete Import Mapping**

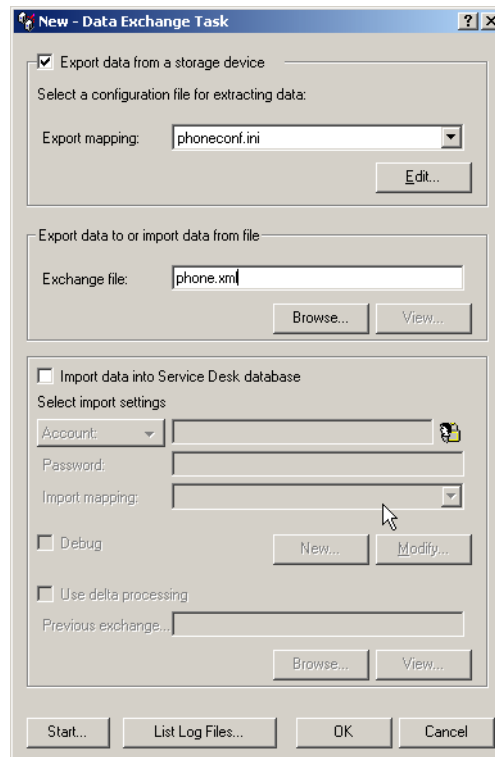


**Step 5. Export the Excel spreadsheet data.**

1. Right-click in the Data Exchange Task window to create a New Data Exchange Task. An empty Data Exchange dialog box will open.
2. Select the Export Data from a storage device check box.
3. Enter the name of the .ini file you created in the Export mapping field. For this example it is `phoneconf.ini`.
4. In the Exchange file field, enter the name you specified for it when configuring the .ini file. For this example the file will be called `phone.xml`:



**Figure E-7 Example Export Data Task**



5. Click Save to save this task for use another time, and then OK to export the data from the Excel file.
6. Check the phoneconf\_exp.log file to verify that the export was successful.

**Step 6. Import the Excel data.**

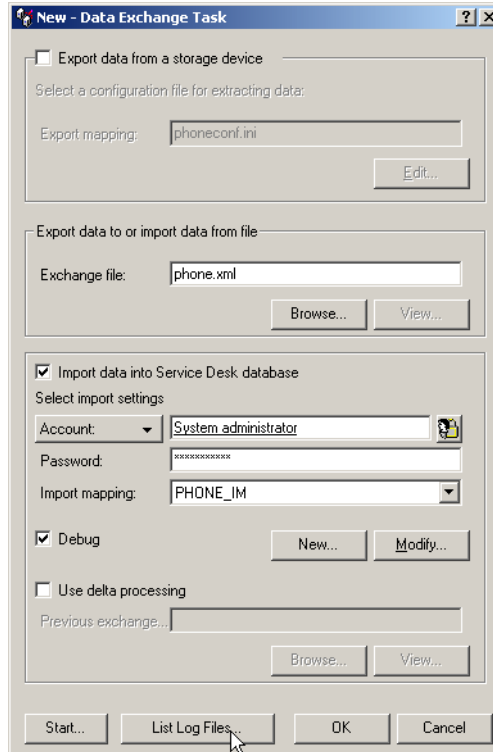
1. Right-click in the Data Exchange Task window to create a New Data Exchange Task. An empty Data Exchange dialog box will open.
2. In the Exchange file field, enter the name of the phone.xml file.
3. Select the Import data into Service Desk database check box.
4. Enter a Service Desk account and password created for integration purposes.

## Examples

### Importing From an MS Excel Spreadsheet

5. In the Import mapping field use the drop-down arrow to find the import mapping you created for importing the Excel file. In this example it is PHONE\_IM:

**Figure E-8 Example Import Data Task**



6. Select the Debug check box and click OK to import the data.
7. Check PHONE\_IM\_imp.log to verify that the import was successful.

## Importing Data With Relations

This example focuses on the process of configuring the extractor file and creating the import mapping to import data from a data source. Special attention is given in this example to how you can import relations. The information about adjusting the configuration file and creating the import mapping is combined for each class in this example, so that you can directly see the relationship between the two. The overall process contains the following ordered steps:

1. Analyze your data source to determine what you will export and how it is stored.
2. Create an ODBC link
3. Configure the Extractor
4. Export data
5. Map the exported data to templates, items and fields in Service Desk.
6. Import the data.

### Data Source

To configure the extractor (.ini file) you need to know how information is stored in your data source and what you want to export. Many applications provide a table description or similar document that can be used for this purpose. In Service Desk a Data Dictionary is provided with this type of information. The Service Desk Dictionary is useful when mapping where in Service Desk the data will be imported to and the relations between items in Service Desk.

The data used for this example was taken from a Microsoft Excel spreadsheet. The major differences between importing data from an Excel spreadsheet and a database is in how the ODBC link is made see “Defining the ODBC Link” on page 248, and the task of defining areas in your excel file, “Preparing the Excel Sheet for Data Exchange” on page 247. When looking at your data you should always think in terms of the SOURCE or database table, and COLUMNS within the tables. In the Excel spreadsheet the information is also grouped into tables, SOURCES, and columns, COLUMNS.

**Figure E-9**            **EMPLOYEE SOURCE**

A	B	C	D
<b>EMPLOYEE</b>			
SEARCHCODE	FIRSTNAME	LASTNAME	EMAIL
EMP100003	Maarten	van Dijk	maarten_vandijk@acme.com
EMP100004	John	Smith	john_smith@acme.com
EMP100005	Shelley	Long	shelley_long@acme.com
EMP100006	Guy	François	guy_francois@acme.com
EMP100007	Helen	Morris	helen_morris@acme.com
EMP100008	Monica	van Velden	monica_vanvelden@acme.com
EMP100009	Frank	Zappa	frank_zappa@acme.com
EMP100010	Joe	Barbarese	joe_barbarese@acme.com
EMP100011	Pete	McPherson	pete_mcpherson@acme.com
EMP100012	Sandra	Berke	sandra_berke@acme.com
EMP100013	Michael	Sasek	michael_sasek@acme.com
EMP100014	Zem	Philips	zem_philips@acme.com

**Figure E-10**            **EMPLOYEE SOURCE - Continued**

E	F	G	H
TELEPHONE	NUMBER	ORG	WG
303	303	HPHOLLAND	ORGHELP
304	304	HPFRANCE	ORGPC
305	305	HPHOLLAND	ORGPC
306	306	HPSPAIN	ORGHELP
307	307	HPUK	ORGIT
308	308	HPFRANCE	ORGPC
309	309	HPGERMANY	ORGIT
310	310	HPHOLLAND	ORGIT
311	311	HPSPAIN	ORGHELP
312	312	HPGERMANY	ORGHELP
313	313	HPSPAIN	ORGIT
314	314	HPSPAIN	ORGIT

**Figure E-11 ORGANIZATION, SOFTWARE & WORKGROUP SOURCES**

ORGANISATION		SOFTWARE	
SEARCHCODE2	NAME	SEARCHCODE3	NAME3
ORG20001	HPHOLLAND	NT4	NT4.0
ORG20002	HPFRANCE	W2000	WIN2000
ORG20003	HPUK		
ORGANISATION		WORKGROUP	
SEARCHCODE4	NAME4	SEARCHCODE4	NAME4
ORG20004	HPGERMANY	ORGHELP	ORG-Helpdesk
ORG20005	HPSPAIN	ORGPC	ORG-PC
		ORGIT	ORG-IT

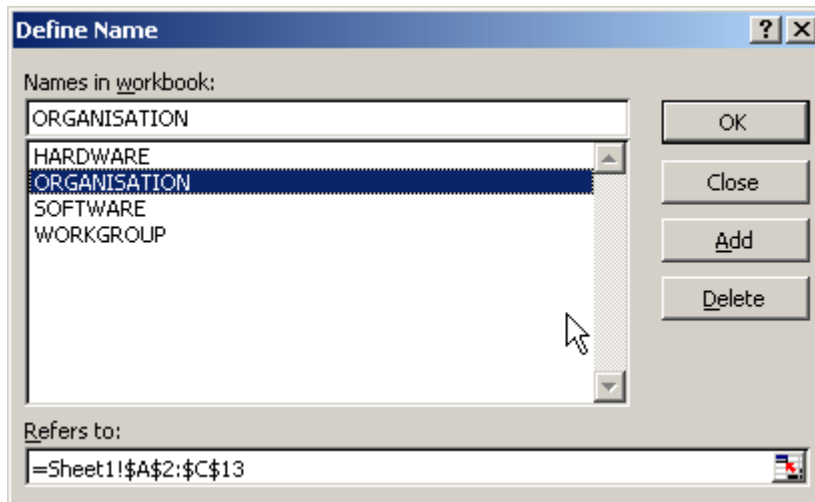
**Figure E-12 HARDWARE SOURCE**

HARDWARE			
SEARCHCODE5	NAME5	SW	USER
ORGPC001	ORGPC1	NT4.0	EMP100005
ORGPC002	ORGPC2	WIN2000	EMP100009
ORGPC003	ORGPC3	WIN2000	EMP100006
ORGPC004	ORGPC4	NT4.0	EMP100014
ORGPC005	ORGPC5	NT4.0	EMP100012
ORGPC006	ORGPC6	WIN2000	EMP100012
ORGPC007	ORGPC7	WIN2000	EMP100008

### Preparing the Excel Sheet for Data Exchange

When exporting data from an Excel sheet you will need to define the areas of data in your excel sheet. In essence you will be grouping the data into tables that are easier to identify for import mapping. In this example the following areas are defined in the Excel sheet; EMPLOYEE, ORGANISATION, SOFTWARE, WORKGROUP, and HARDWARE. To define an area in an Excel sheet; from the Excel Toolbar select Insert, Name, and then Define. You can then set or view the definition for each area as shown in the following example:

**Figure E-13**      **Excel File Name Definition**



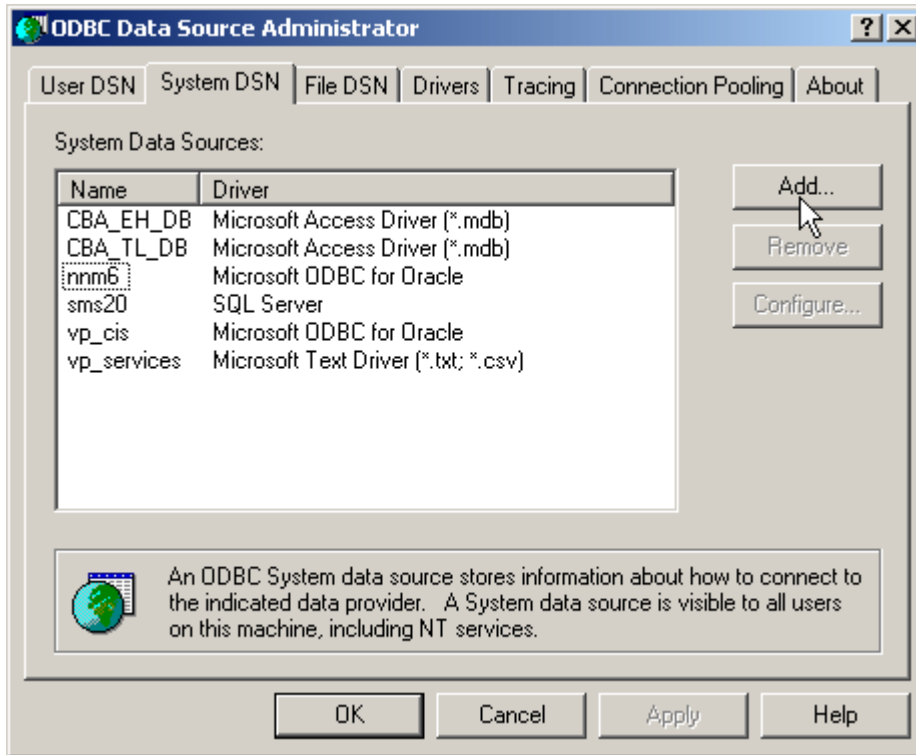
After you create the ODBC link and create or modify the configuration file you can export the data using the Data Exchange dialog box. Clear the option for importing data until after you have had time to verify that your export was successful. For more information see “Using a Task to Export Data From a Storage Device” on page 82.

### **Defining the ODBC Link**

An ODBC connection must be configured on the application server where the data you will import is located. For example, if the exported XML file is on your Service Desk application server, that is where you need to configure the ODBC link from. Service Desk uses System DSN for data exchange. Following is an example of how to set up an ODBC link for Microsoft Excel files. For other types of ODBC links see “Defining the ODBC link” on page 78.

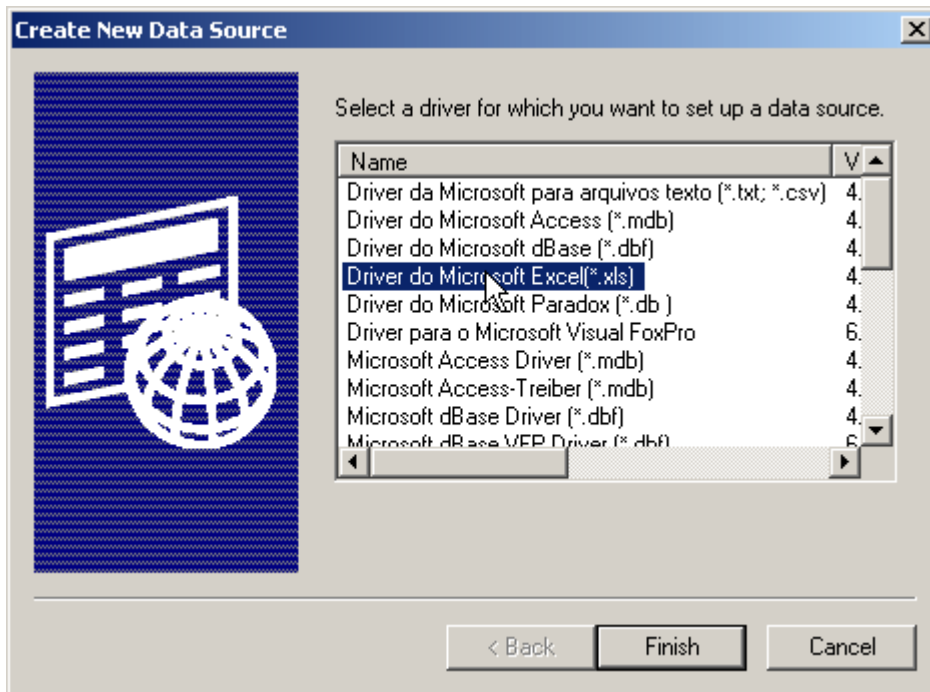
1. From the System DSN tab select Add to create a new ODBC link:

**Figure E-14 Add the Excel ODBC Driver**



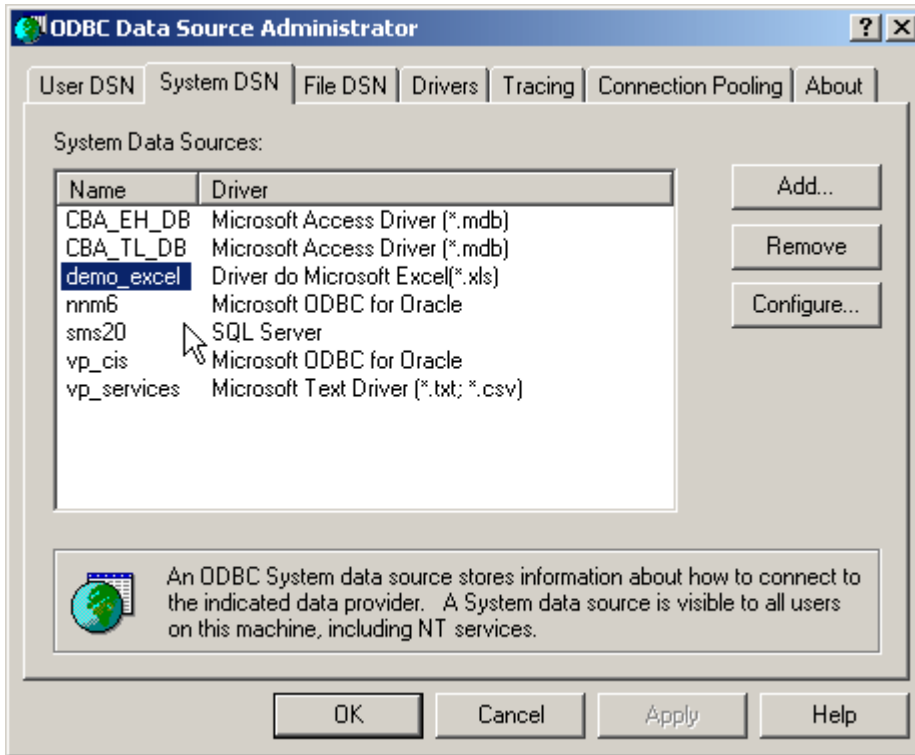
2. Select the Microsoft Excel Driver, it will have an \*.xls extension, then click Finish:

**Figure E-15**    **New Data Source**



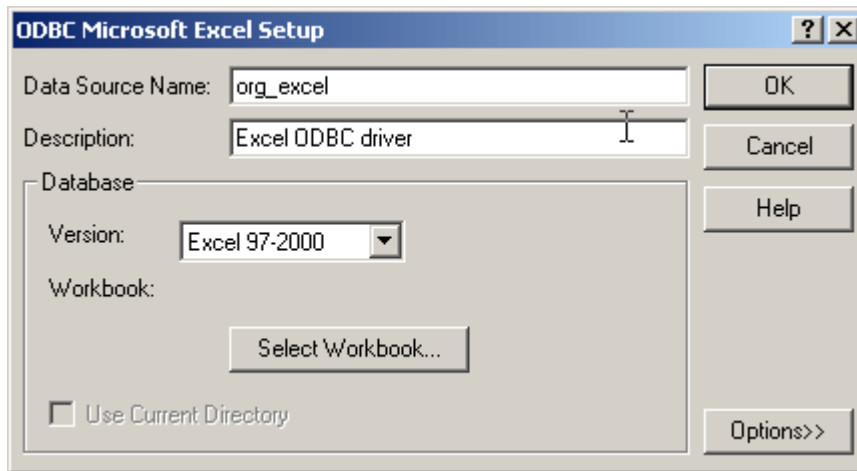


**Figure E-16**    **Select the New ODBC Connection**



3. Type the Data Source Name and a Description. The Data Source Name should match the DSN name in your configuration file, the description is for your own reference.

**Figure E-17** Excel ODBC\_DSN



4. Click **Select Workbook** to locate the data source that you will use. You will be able to use an explorer type tool to locate the workbook. Click **OK** to finish the procedure.

## **Configuring the Extractor and the Import Mapping**

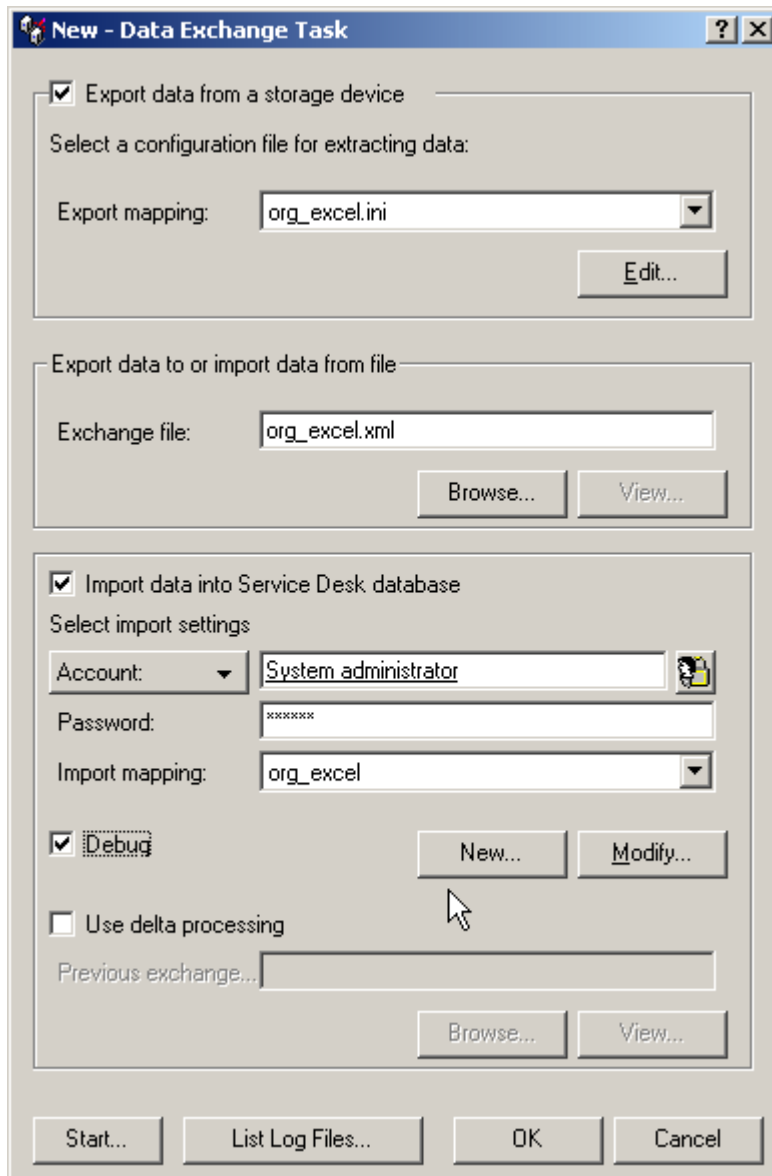
This section explains how the example `org_excel.ini` configuration file was created and the special attention given to exporting relations. The extractor is an executable extractor file that uses parameters to connect to the external source and extract data. After the data is exported the import mapping needs to be configured correctly for importing the relations.

Mapping data from the XML file to Service Desk items and fields can be a labor intensive job, if you cannot use one of the import mapping examples provided you can make the process easier by creating a well structured configuration file and making sure it works before going on to the import mapping. If you don't you might find yourself creating the import mapping more than once. For this example the import mapping is shown after each class in the configuration file is described, to better demonstrate how what is entered in the configuration file effects what needs to be entered in the import mapping. Normally you would create the entire configuration file and then make the import mapping based on the information in the XML file.

The export process can be started from the application user interface or

from the command line, see “Exporting Data” on page 82, or “Exporting From the Command Line” on page 84 for additional information. The configuration file can be opened from the Data Exchange dialog box, click `Edit` to edit or view the configuration file selected in the Export mapping field as shown below:

**Figure E-18**      **Org\_Excel Data Exchange Task**



In this example the following Classes will be configured for export by the

extractor:

<b>Class</b>	<b>Description</b>
SOFTWARE	This class is selected to export the different software items from the data source into the XML file.
ORG	This class will export the organization items from the data source into the XML file.
EMP	This class will export the employees and the employee attributes entered into the XML file: Email addresses, and employee organization.
PHONE	This class will export the employee telephone numbers from the data source to the XML file.
HARDWARE	This class will export the different hardware items from the data source into the XML file.
HWUSERS	This class will export the different hardware users from the data source into the XML file.
REL	This class will export the different relationships from the data source into the XML file.
WORKGROUP	This class will export the different workgroups from the data source into the XML file.
WORKGROUPEMP	This class will export the different workgroup members from the data source into the XML file.

### **Example E-1 Configuration File**

```
[DSN]
NAME=org_excel
USR=
PWD=

[SYSTEM]
LOG=TRUE
XML=TRUE
LOG_FILE=org_excel.log
OUTPUT_FILE=org_excel.txt
```

## Examples

### Importing Data With Relations

```
XML_OUTPUT_FILE=org_excel.xml  
APPLICATION_NAME=org_excel
```

```
[ CLASSES ]  
NAME=SOFTWARE , ORG , EMP , PHONE , HARDWARE , HWUSERS , REL , WORKGROUP , WORKGROUPEMP
```

```
[ SOFTWARE ]  
SOURCE=SOFTWARE  
ATT=[ SEARCHCODE3 ] , [ NAME3 ]  
COLUMNS=[ SEARCHCODE3 ] , [ NAME3 ]  
LOADTABLE=TRUE
```

```
[ ORG ]  
SOURCE=ORGANISATION  
ATT=[ SEARCHCODE2 ] , [ NAME ]  
COLUMNS=[ SEARCHCODE2 ] , [ NAME ]  
LOADTABLE=TRUE
```

```
[ EMP ]  
SOURCE=EMPLOYEE  
ATT=[ SEARCHCODE ] , [ FIRSTNAME ] , [ LASTNAME ] , [ EMAIL ] , [ ORG ]  
COLUMNS=[ SEARCHCODE ] , [ FIRSTNAME ] , [ LASTNAME ] , [ EMAIL ] , [ ORG ]  
LOADTABLE=TRUE
```

```
[ PHONE ]  
SOURCE=EMPLOYEE  
ATT=[ SEARCHCODE ] , [ PHONENUMBER ]  
COLUMNS=[ SEARCHCODE ] , [ NUMBER ] AS [ PHONENUMBER ]  
LOADTABLE=TRUE
```

```
[ HARDWARE ]  
SOURCE=HARDWARE  
ATT=[ SEARCHCODE5 ] , [ NAME5 ]  
COLUMNS=[ SEARCHCODE5 ] , [ NAME5 ]  
LOADTABLE=TRUE
```

```
[ HWUSERS ]  
SOURCE=HARDWARE  
ATT=[ SEARCHCODE5 ] , [ USER ]  
COLUMNS=[ SEARCHCODE5 ] , [ USER ]  
LOADTABLE=TRUE
```

```
[REL]
SOURCE=HARDWARE
ATT=[SEARCHCODE5],[SW]
COLUMNS=[SEARCHCODE5],[SW]
LOADTABLE=TRUE
```

```
[WORKGROUP]
SOURCE=WORKGROUP
ATT=[SEARCHCODE4],[NAME4]
COLUMNS=[SEARCHCODE4],[NAME4]
LOADTABLE=TRUE
```

```
[WORKGROUPEMP]
SOURCE=EMPLOYEE
ATT=[SEARCHCODE]
COLUMNS=[SEARCHCODE],[WG] AS [SEARCHCODE4]
LOADTABLE=TRUE
PARENT=WORKGROUP
PARENT_RELATION=[WG]=[SEARCHCODE4]
PARENT_RELATION_NAME=PARENT
```

### **Configuring the DSN Section**

The NAME line, in the DSN section of the configuration file, is the Data Source Name you entered when you created the ODBC link. Each configuration file needs to have its own ODBC link with a unique Data Source Name.

### **Configuring the SYSTEM Section**

When running Data Exchange from the user interface you only need to fill in the LOG, XML, and APPLICATION NAME fields. The fields LOG\_FILE, OUTPUT\_FILE, and XML\_OUTPUT\_FILE only need to be filled in if you run Data Exchange from a command line.

### **Configuring the CLASSES Section**

When developing the configuration file, the order that the classes are listed in the CLASSES section is important. The sections that follow should mirror the order in the CLASSES section. The extractor will take each class defined in the CLASSES section and make SQL statements. Sources, columns and classes are selected and relationships made according to the definitions in the configuration file, see “The Extraction

Process” on page 80 for additional information. When you later import the data the objects are imported in the order they were written. The order takes some thought when developing the configuration file for importing relations. If you look in the configuration file that follows you will see that WORKGROUP will be extracted before WORKGROUPEMP, because WORKGROUPEMP is the child relation of WORKGROUP.

---

**NOTE**

If a class is not in the CLASSES file it will not be imported, even if the class is described later in the configuration file. You will not get an error message if a class is missing, a missing class can only be detected by a manual check of the configuration file, or seeing that it was not exported to the XML file or imported into Service Desk.

---

### **Configuring the SOFTWARE Class**

In the data source the class called Software is stored in the table called SOFTWARE, from the excel spreadsheet it is a section defined as SOFTWARE in the Excel sheet. The table is the SOURCE in the configuration file. Within the SOURCE two attributes are selected for export, the attributes are put in the ATT field. In the COLUMNS field you need to specify where within the SOURCE those attributes, ATT items, are coming from. For this example class, the COLUMN names and ATT names match so no aliases are necessary.

You always need to include a SEARCHCODE. The SEARCHCODE is a unique key that must always be exported and mapped to the field `search code` in Service Desk for identifying the class. For Software the column SEARCHCODE3 contains the unique code name given to each software item, the NAME3 column contains the name the different software items were given.

The LOADTABLE field is used to specify how the data is stored while it is being extracted. In this case the LOADTABLE is set to TRUE so that the records will be cached in memory, because the class in this example does not have any parent-child relations that need to be searched for.

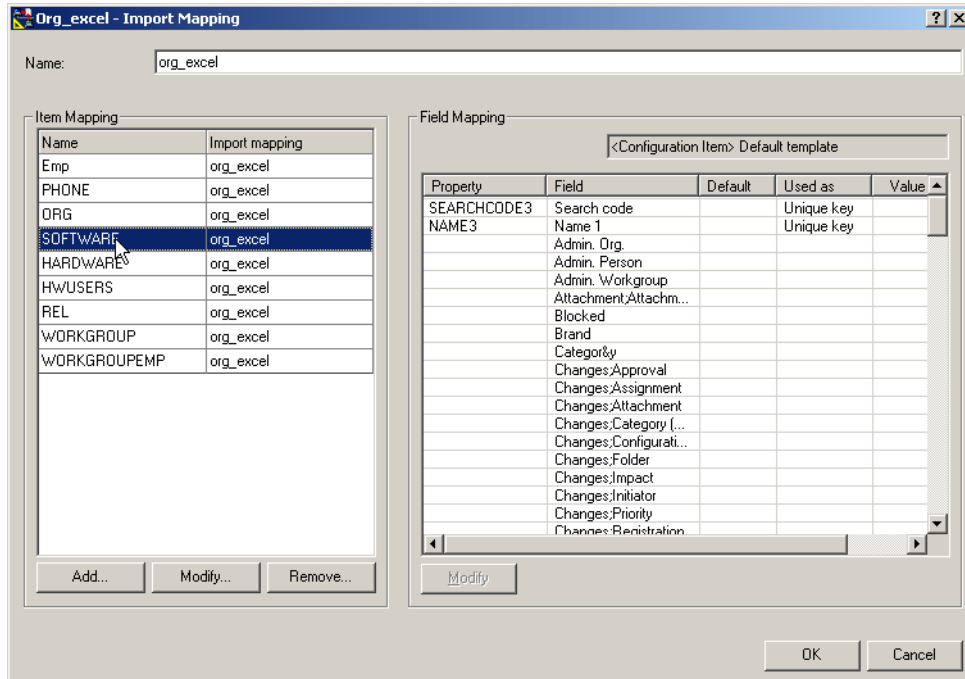
### **Import Mapping for the SOFTWARE Class**

In the import mapping the class name in brackets [SOFTWARE] is mapped to the item called `Configuration Item` in Service Desk. The default `template` is used for this class. The attributes in the ATT line are mapped to `Fields` that are available in the default template. In this



case [SEARCHCODE3] is mapped to the Search code field and [NAME3] is mapped to the Name1 field. Both attributes are unique, the mapping can be extended later by modifying the configuration file to include more attributes. The completed import mapping is displayed in the following dialog box:

**Figure E-19 Import Mapping - SOFTWARE**

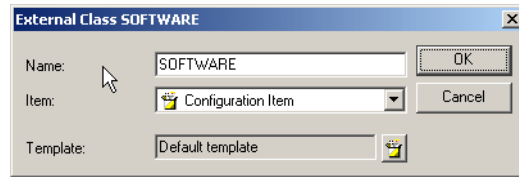


Scrolling down in the dialog box above will show the following additional fields:

Field	Default
Admin.Org	Invention Inc.
Max.Installations	1

The following dialog box shows the import mapping for the external class to the Configuration Item in Service Desk and the template selected:

**Figure E-20 Mapping External Class - SOFTWARE**



### Configuring the ORG Class

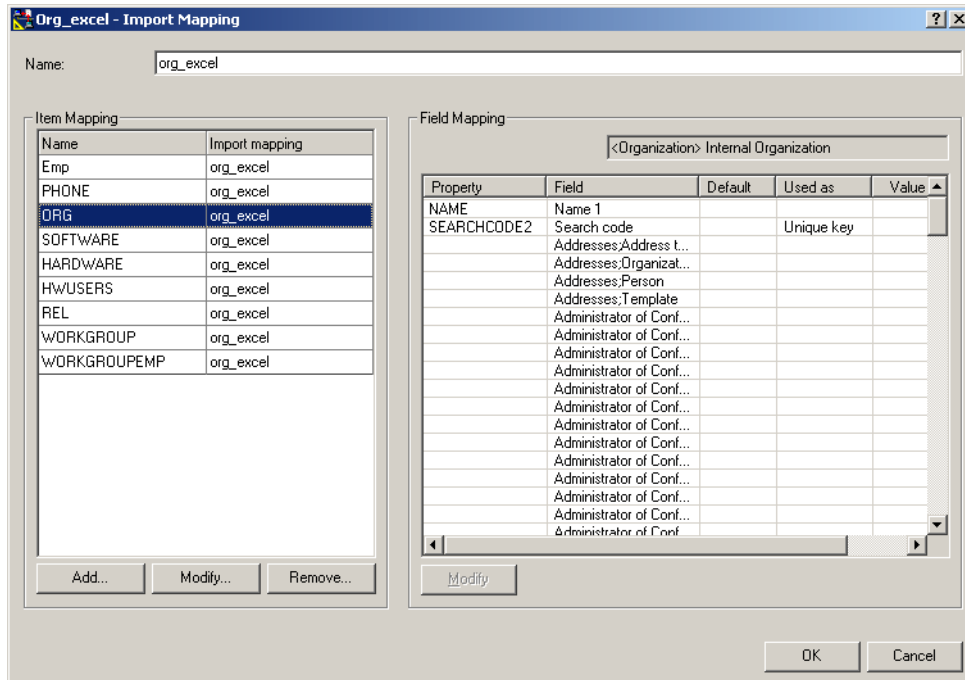
This class is imported in a similar manner as the class SOFTWARE. One major difference is that the SOURCE for the class has a different name. The class name is ORG and the SOURCE is the ORGANISATION table. This shows that you can give a class any name you want in the configuration file as long as the SOURCE is provided. Keep in mind that when the exported data is put into the XML file it will be sorted under the class name that you give it and that class name is what you will need to map to an item in Service Desk.

Another point to note about this class is that this class is intentionally put in the configuration file before the class EMP. The reason for that is because EMP will export ORG as an attribute.

### Import Mapping for the ORG Class

In the import mapping the class name in brackets [ORG] is mapped to the item called Organization in Service Desk. The template called Internal Org is used for this class. The attributes in the ATT line are mapped to Fields that are available in the Internal Org template. In this case [SEARCHCODE2] is mapped to the Search code field and [NAME] is mapped to the Name1 field. The mapping can be extended later by modifying the configuration file to include more attributes. The completed import mapping is displayed in the following dialog box:

**Figure E-21** Import Mapping - ORG

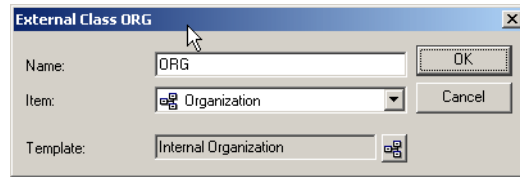


Scrolling down in the dialog box above will show the following additional fields:

Field	Default
Status	Active
Category	Organization

The following dialog box shows the how the external class was mapped to the Organization item in Service Desk and the template selected:

**Figure E-22** Mapping External Class - ORG



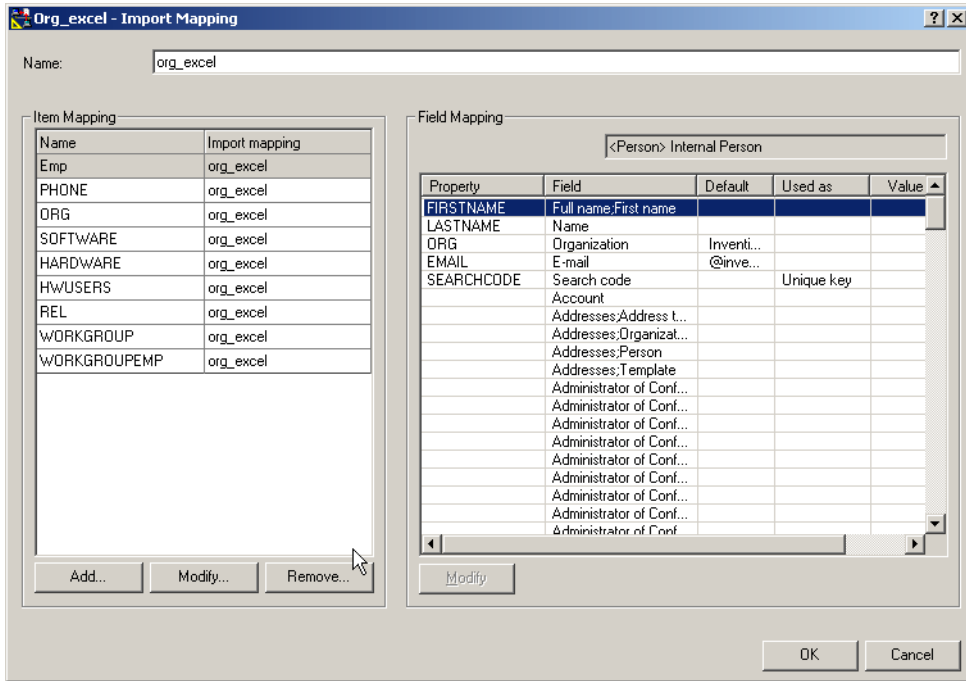
### Configuring the EMP Class

The class EMP is configured to be imported before the following class, PHONE. That is because phone numbers will be linked to employees.

### Import Mapping for the EMP Class

In the import mapping the class name in brackets [EMP] is mapped to the item called Person in Service Desk. The default template is used for this class. The attributes in the ATT line are mapped to Fields that are available in the default template. In this case [SEARCHCODE] is mapped to the Search code field, [ORG] is mapped to the Organization field, [LASTNAME] is mapped to Name, [FIRSTNAME] is mapped to Full name;First name, and [EMAIL] is mapped to E-mail. Search code is the unique key for identifying the class. The completed import mapping is displayed in the following dialog box:

**Figure E-23** Import Mapping - EMP

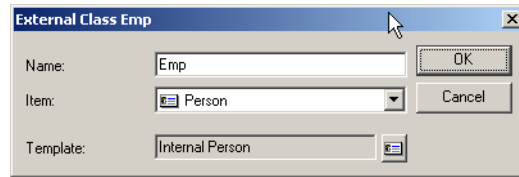


Scrolling down in the dialog box above will show the following additional fields:

Field	Default
Status	Active
Category	Employee

The following dialog box shows the how the external class was mapped to the Person item in Service Desk and the template selected, in this case the Internal Person template:

**Figure E-24 Mapping External Class - EMP**



You will need to create an item reference for the ORG property. To create the item reference:

1. Double-click the organization field, in the Import Mapping dialog box, to open the field mapping.
2. In the Field Mapping dialog box the External Property field should contain the ORG property.
3. In the A reference to Item field, use the drop-down arrow to enter the Name1 item.
4. Click OK to finish.

### Configuring the PHONE Class

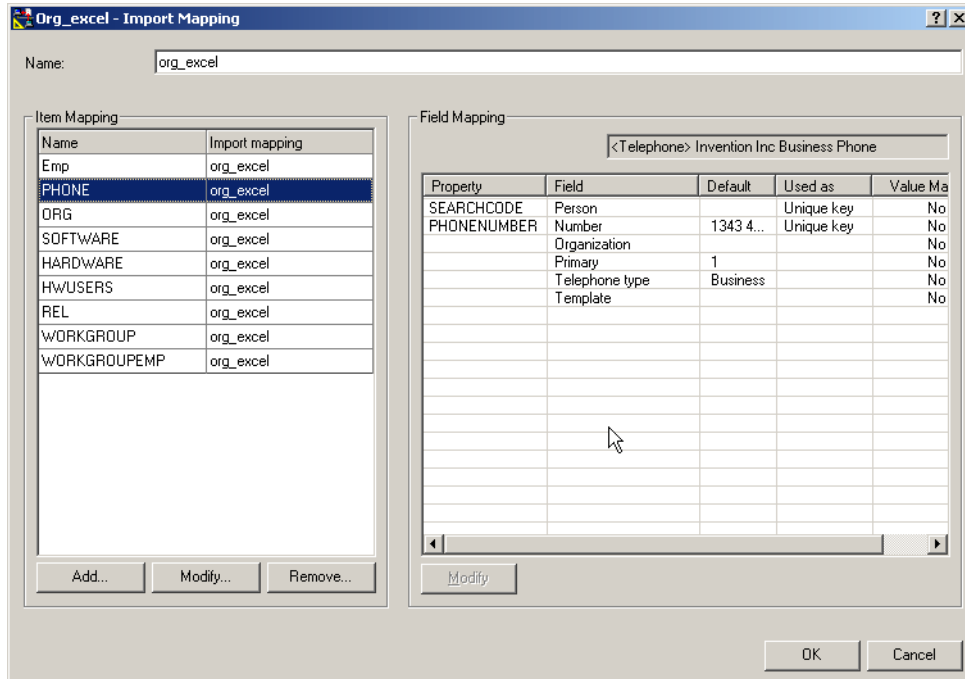
The PHONE class is taken from the EMPLOYEE SOURCE. This class is being exported as a separate class because later it will be imported to a different item with a different template. The EMP class will be imported to the Person item and the PHONE class will be imported to the Telephone item in Service Desk. When looking at the ATT and COLUMN names you will notice that they are different. ATT, attribute names are decided by you, just like class names. If they are different than the COLUMN names, if they have an alias, you need to include that with the AS statement in the COLUMN line. In the example the ATT called PHONENUMBER will be exported from the COLUMN called NUMBER and listed in the XML file as PHONENUMBER.

### Import Mapping for the PHONE Class

In the import mapping the class name in brackets [PHONE] is mapped to the item called Telephone in Service Desk. The phone template is used for this class. The attributes in the ATT line are mapped to fields that are available in the default template. In this case [SEARCHCODE] is mapped to the Person field and [PHONENUMBER] is mapped to the Number field. Both attributes are unique keys. Two default settings are included so that the number will be imported as the primary Business

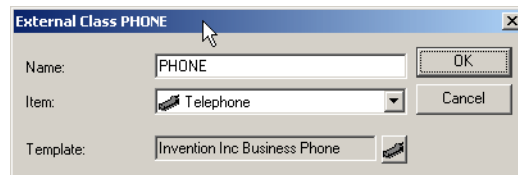
phone for that person. For information on importing multiple phone numbers, see the following example “Importing Multiple Telephone Numbers” on page 277. The import mapping for the PHONE class is displayed in the following dialog box:

**Figure E-25** Import Mapping - PHONE



The following dialog box shows the how the external class was mapped to the Telephone item in Service Desk and the template selected:

**Figure E-26** Mapping External Class - PHONE



You will need to create an item reference for the SEARCHCODE property. To create the item reference:

1. Double-click the `Person` field, in the import mapping dialog box, to open the field mapping dialog box.
2. In the Field Mapping dialog box the `External Property` field should contain the `SEARCHCODE` property.
3. In the `A` reference to `Item` field, use the drop-down arrow to enter the `Search code` item.
4. Click `OK` to finish.

### Configuring the HARDWARE Class

The `HARDWARE` class is intentionally listed for extraction before the class `HWUSERS` because it is the parent of that class.

---

#### NOTE

The classes; `HWUSERS` and `WORKGROUPEMP` are configured and imported differently in this example, yet the end result is the same. This demonstrates that there are multiple ways of configuring how items are imported into Service Desk. When configuring how relations will be imported into Service Desk it can be specified in the configuration file or in the import mapping. The process of doing it in the configuration file is simpler and provides better performance.

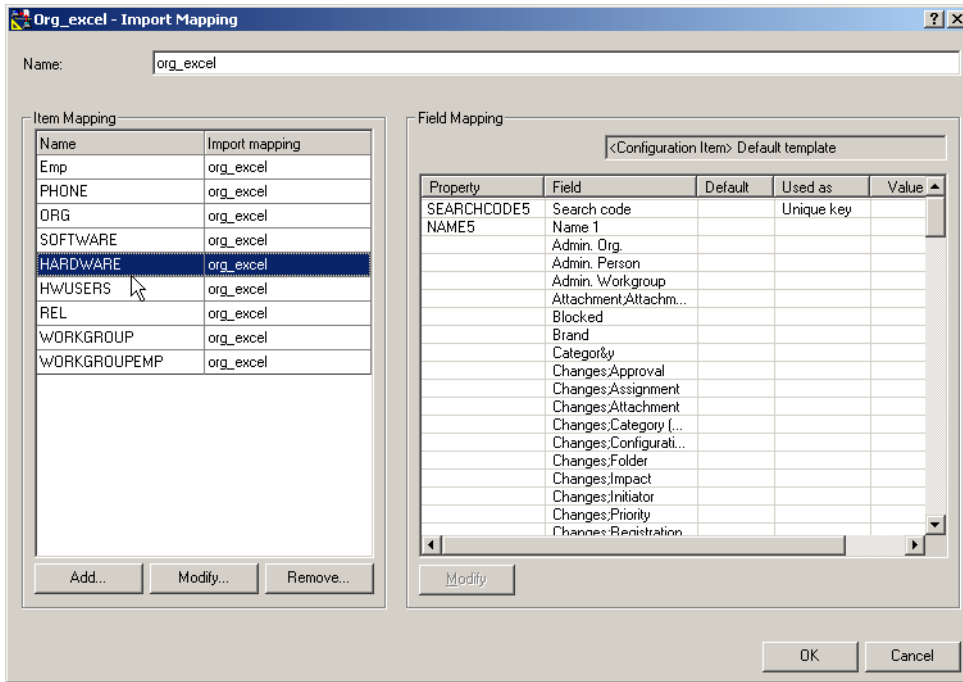
---

### Import Mapping for the HARDWARE Class

In the import mapping the class name in brackets [`HARDWARE`] is mapped to the item called `Configuration Item` in Service Desk. The default template is used for this class. The attributes in the `ATT` line are mapped to fields that are available in the default template. In this case [`SEARCHCODE5`] is mapped to the `Search code` field and [`NAME5`] is mapped to the `Name1` field. The completed import mapping is displayed in the following dialog box:



**Figure E-27 Import Mapping - HARDWARE**



Scrolling down in the dialog box above will show the following additional fields:

Field	Default
Admin.Org	Invention Inc
Max.Installations	1
Unique	1
Status	Production

The following dialog box shows the how the external class was mapped to the Configuration Item in Service Desk and the template selected:

**Figure E-28** Mapping External Class - HARDWARE



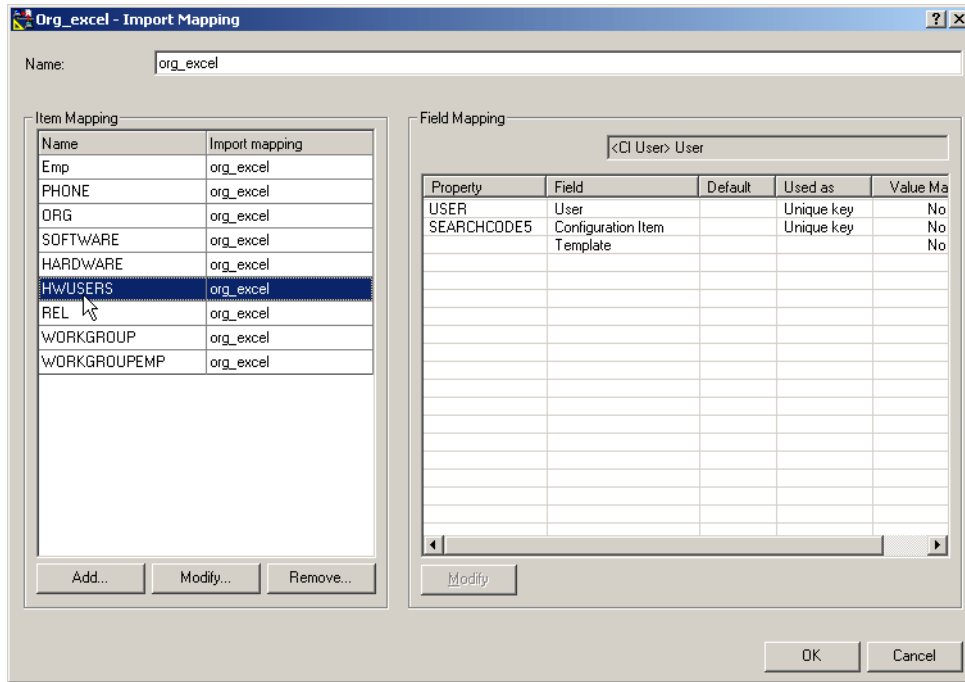
### Configuring the HWUSERS Class

The class HWUSERS is the child class of HARDWARE and was created separately so that it can be imported to a different item in Service Desk. HARDWARE is imported as a configuration item and HWUSERS is imported as CI User in Service Desk.

### Import Mapping for the HWUSERS Class

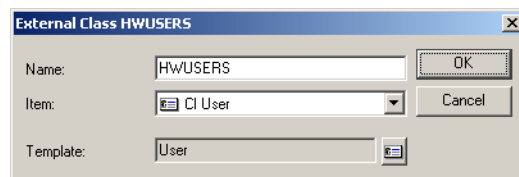
In the import mapping the class name in brackets [HWUSERS] is mapped to the item called CI User in Service Desk. The USER template is used for this class. The attributes in the ATT line are mapped to fields that are available in the USER template. In this case [SEARCHCODE5] is mapped to the Configuration item field, because this is a child of that item. The [USER] attribute is mapped to the User field. The completed import mapping is displayed in the following dialog box:

**Figure E-29** Import Mapping - HWUSERS



The following dialog box shows the how the external class was mapped to CI User in Service Desk and the template selected:

**Figure E-30** Mapping External Class - HWUSERS



You will need to create an item reference for the SEARCHCODE5 and the USER properties. To create the item references:

1. Double-click the Configuration Item field, in the import mapping dialog box, to open the field mapping dialog box.
2. In the Field Mapping dialog box the External Property field should contain the SEARCHCODE5 property.

3. In the `A` reference to `Item` field, use the drop-down arrow to enter the `Search code` item.
4. Click `OK` to return to the `Import Mapping` dialog box.
5. Double-click the `user` field to open the field mapping dialog box.
6. In the `Field Mapping` dialog box the `External Property` field should contain the `USER` property.
7. In the `A` reference to `Item` field, use the drop-down arrow to enter the `Search code` item.
8. Click `OK` to finish.

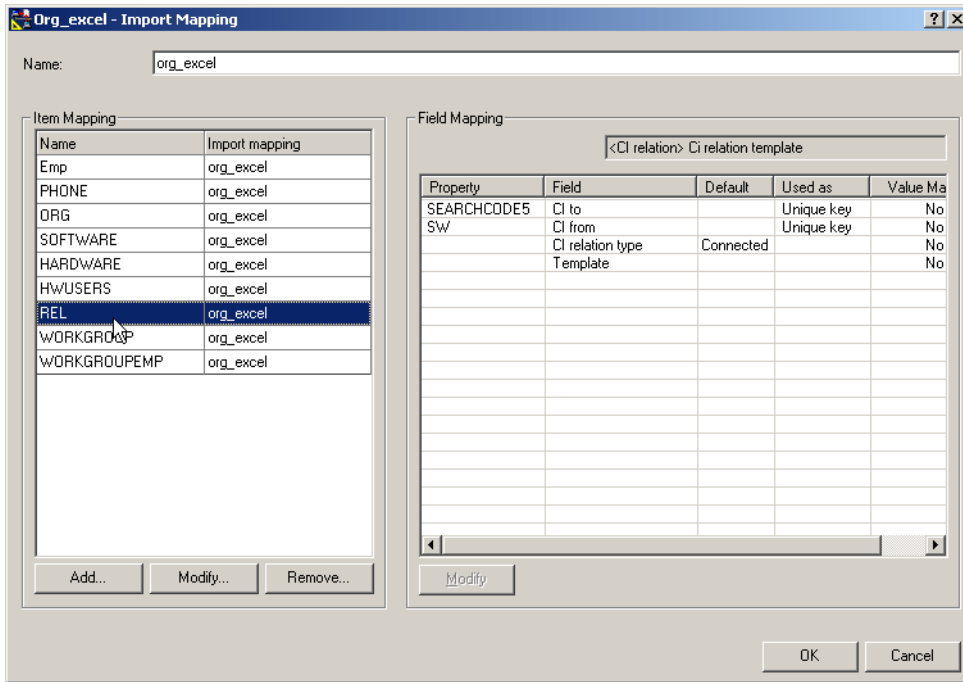
### Configuring the REL Class

This class is being used to define the relations in the `SOURCE` called `HARDWARE` for the attributes listed. This is a `1:Rel:1` relation. The Class `REL` is mapped to the `CI Relation` item that contains a number of different relations as fields. In this manner you can specify the relation of the classes that are linked to this class.

### Import Mapping for the REL Class

In the import mapping the class name in brackets `[REL]` is mapped to the item called `CI Relation` in `Service Desk`. The `CI Relation` template is used for this class. The attributes in the `ATT` line are mapped to fields that are available in the `CI relation` template. In this case `[SEARCHCODE5]` is mapped to the `CI to` field, and the attribute `SW`, which refers to software, is mapped to `CI from`. Any default values entered will be overwritten by the value mapping. If you want to give an attribute a default value you can add it by opening and modifying the template. For example, you can enter a default value to define the relationship, if you look at the field `CI relation type` you will see that it gives this class the default value `Installed Software`. It is also possible to create a value mapping to define the relations or to reference the relation table in `Service Desk`. The completed import mapping is displayed in the following dialog box:

**Figure E-31 Import Mapping - REL**

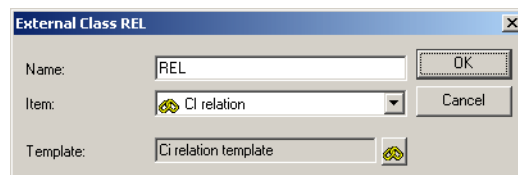


**NOTE**

The CI relation type field is mandatory and should be entered either as a default in the template used, or by the imported class.

The following dialog box shows how the external class is mapped to the CI Relation item in Service Desk and the template selected:

**Figure E-32 Mapping External Class - REL**



You will need to create an item reference for the SEARCHCODE5 and the SW properties. To create the item references:

## Importing Data With Relations

1. Double-click the `CI to` field, in the import mapping dialog box, to open the field mapping dialog box.
2. In the Field Mapping dialog box the `External Property` field should contain the `SEARCHCODE5` property.
3. In the `A reference to Item` field, use the drop-down arrow to enter the `Search code` item.
4. Select the `This field is used for key binding` check box.
5. Click OK to return to the Import Mapping dialog box.
6. Double-click the `CI from` field to open the field mapping dialog box.
7. In the Field Mapping dialog box the `External Property` field should contain the `SW` property.
8. In the `A reference to Item` field, use the drop-down arrow to enter the `Name1` item.
9. Select the `This item is used for key binding` check box.
10. Click OK to finish.

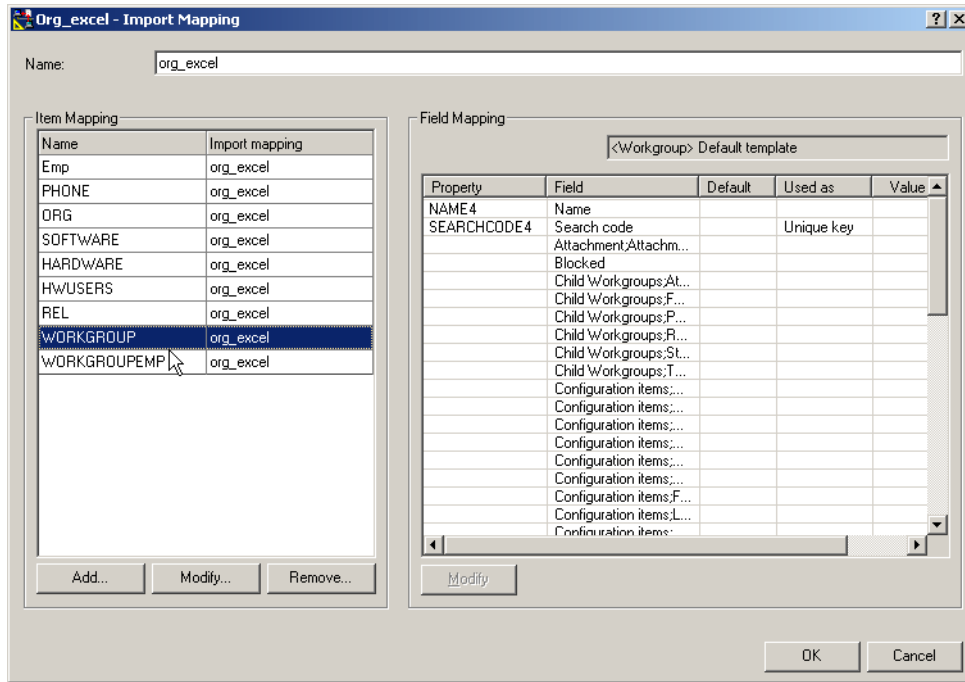
### Configuring the WORKGROUP Class

The class `WORKGROUP` is imported before `WORKGROUPEMP` because it is the parent of that class.

### Import Mapping for the WORKGROUP Class

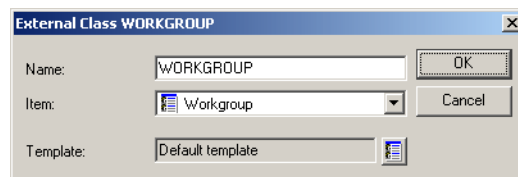
In the import mapping the class name in brackets [`WORKGROUP`] is mapped to the item called `Workgroup` in `Service Desk`. The default template is used for this class. The attributes in the `ATT` line are mapped to fields that are available in the default template. In this case [`SEARCHCODE4`] is mapped to the `Search code` field and [`NAME4`] is mapped to the `Name` field. The completed import mapping is displayed in the following dialog box:

**Figure E-33** Import Mapping - WORKGROUP



The following dialog box shows the how the external class was mapped to the Configuration Item in Service Desk and the template selected:

**Figure E-34** Mapping External Class - WORKGROUP



### Configuring the WORKGROUPEMP Class

This class demonstrates how to import parent-child relations. The goal of this section is to export employee search codes with their workgroup (WG) so that they can be imported into the Person item in Service Desk. The class WORKGROUPEMP is extracted from the SOURCE called EMPLOYEE. One attribute will be extracted and that is the

SEARCHCODE associated with the SOURCE called EMPLOYEE. In the COLUMN field SEARCHCODE is listed and an additional COLUMN called WG is listed as the alias for the column called SEARCHCODE4.

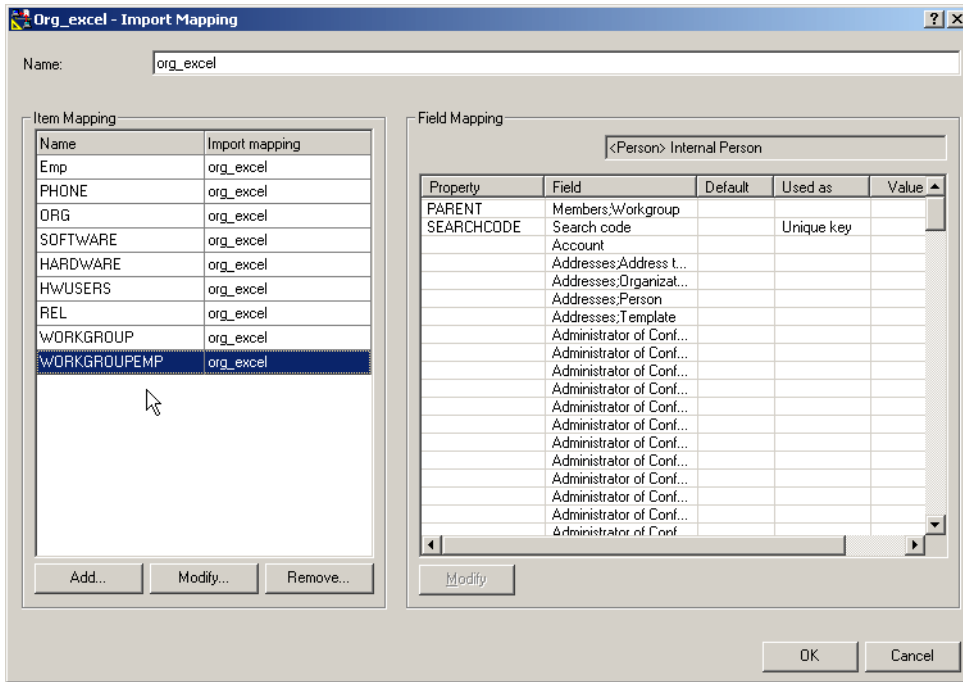
The additional COLUMN is listed to locate the PARENT relation for this class. The PARENT line specifies the parent of this class. This is the WORKGROUPEMP class and the parent is located in the WORKGROUP table. The next line PARENT\_RELATION defines the relationship that exists between this class and the parent, the column name mentioned first, WG, is the foreign key used to identify the COLUMN of the child. The column name that follows the equal sign is the primary key of the alias of the parent, in this case it is SEARCHCODE4. The final line PARENT\_RELATION\_NAME specifies the name to be used when searching for the parent of the relation, in this case PARENT is used. The default for the PARENT\_RELATION\_NAME is PARENT.

### **Import Mapping for the WORKGROUPEMP Class**

In the import mapping the class name in brackets [WORKGROUPEMP] is mapped to the item called *Person* in Service Desk, even though it came from the SOURCE called EMPLOYEE. Once the data is extracted it does not matter what the source was, what matters is what it is called in the XML file. The *Internal Person* template is used for this class. The attributes in the ATT line are mapped to fields that are available in the *Internal Person* template. In this case [SEARCHCODE] is mapped to the *Search code* field and the [PARENT] item is treated as an attribute itself and mapped to the parent *Members ; Workgroup* field. The completed import mapping is displayed in the following dialog box:



**Figure E-35 Import Mapping - WORKGROUPEMP**

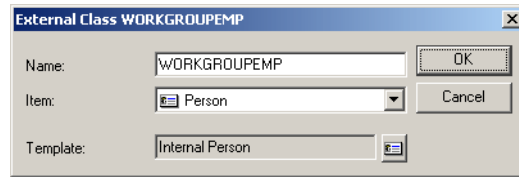


Scrolling down in the dialog box above will show the following additional fields:

Field	Default
Status	Active
Category	Employee

The following dialog box shows the how the external class was mapped to the Person item in Service Desk and the template selected:

**Figure E-36** Mapping External Class - WORKGROUPEMP



## Importing the Data

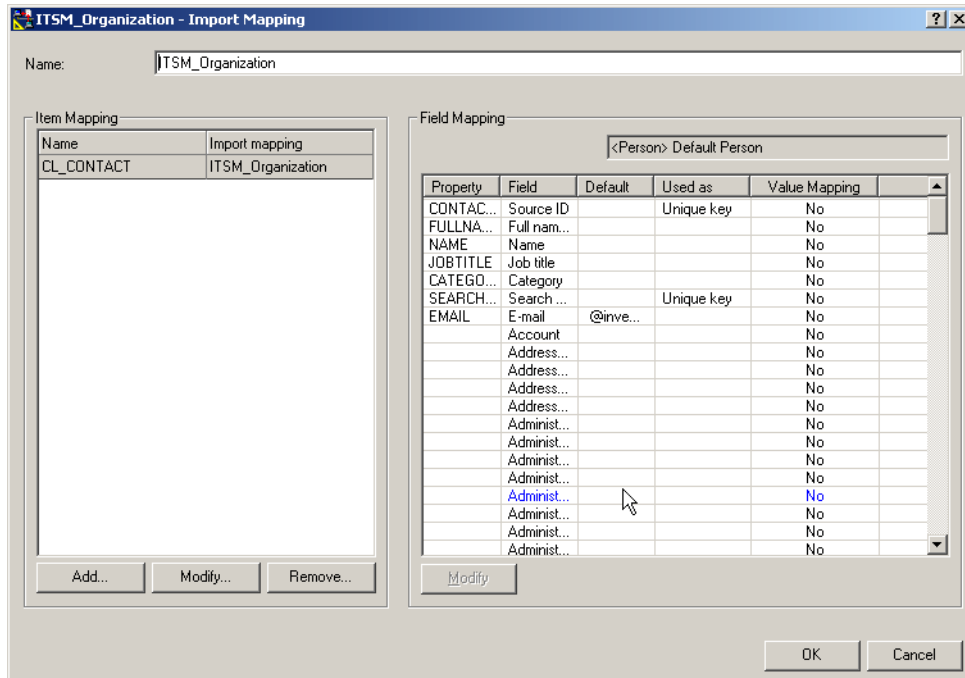
Do not import the data until after you have taken the time to verify that everything was exported correctly. You can run the task to import data from the Data Exchange user interface or from a command line, see “Using a Task to Import Data” on page 119.

## Importing Multiple Telephone Numbers

Importing more than one telephone number can be done by mapping telephone numbers to a person. If you want one telephone number to be shared by more than one person you can do this by adding a Person field to the Telephone template.

In this example multiple contact telephone numbers are mapped to contacts. The import mapping is shown in the following Import Mapping dialog box, where the class `CL_CONTACT` is mapped to the `Person` item in Service Desk. This class contains attributes that describe the contact, for example the name, and job title for each person. The attributes are mapped to Service Desk fields. The attribute `CONTACT_ID` is mapped to the `Source_ID` field in Service Desk and will be used as a search code, it is a unique key:

**Figure E-37** Import Mapping for `CL_CONTACT`

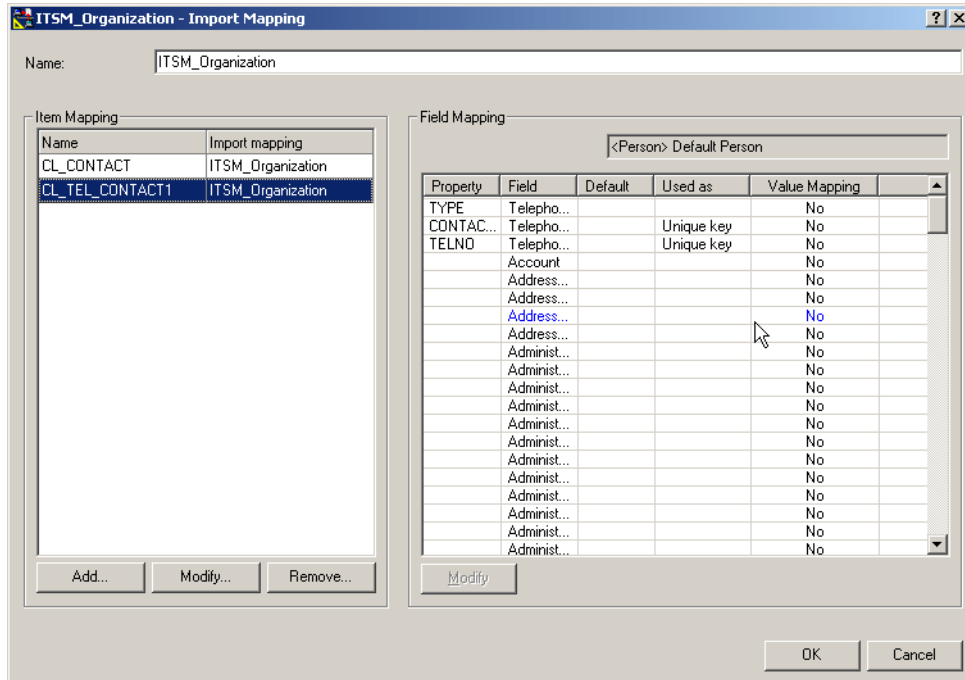


In the next dialog box, the class called `CL_TEL_CONTACT1` is mapped to

Examples  
Importing Multiple Telephone Numbers

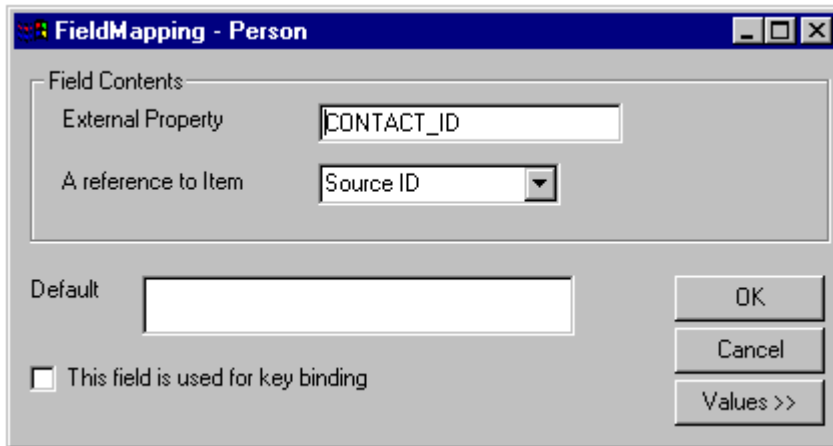
the Person item in Service Desk. The CL\_TEL\_CONTACT1 class contains attributes for telephone numbers:

**Figure E-38** Import Mapping CL\_TEL\_CONTACT1



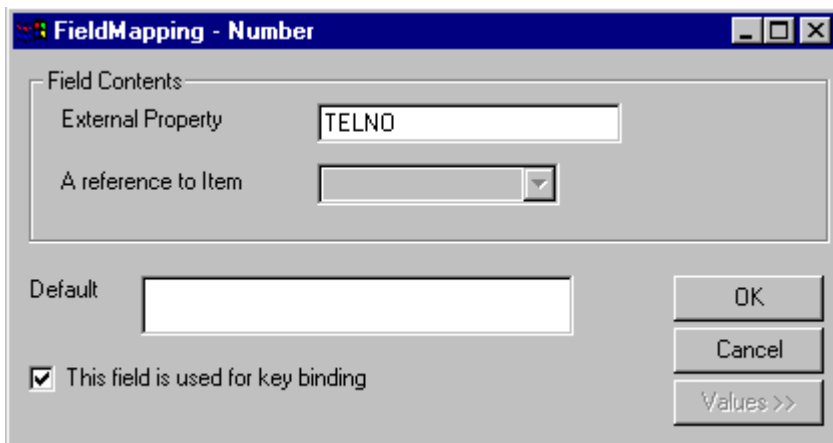
In the following dialog box, the attribute CONTACT\_ID is mapped to the Person field and uses Source ID as a search code to reference the Person item in Service Desk. This reference provides a link between the item containing the phone numbers and the item containing the people (contacts):

**Figure E-39**      **Field Mapping - Person Field**



The attribute `TELNO` is mapped to the Service Desk `Number` field, and set as a unique attribute with the key binding check box in the following dialog box:

**Figure E-40**      **Field Mapping - Number Field**



The third attribute called `TYPE` is mapped to the Service Desk field called `Telephone Type`. The field `Telephone type` contains a code table with values. Click `Values` to expand the Field Mapping dialog box for value mapping. The value `BUSINESS` for the External Property `TYPE` is mapped to the value `Business` in Service Desk. It is possible to enter additional

Examples  
Importing Multiple Telephone Numbers

types if you export more than one type of phone number, for example private and mobile phone numbers. The following dialog box shows the value mapping for the telephone type called BUSINESS:

**Figure E-41**      **Field Mapping - Telephone Type**

The dialog box titled "FieldMapping - Telephone type" contains the following elements:

- Field Contents:**
  - External Property: TYPE
  - A reference to Item: [Dropdown]
- Default:** [Text Field]
- This field is used for key binding
- Buttons: OK, Cancel, Values >>
- Value Mapping:**

Value for External Attribute	Value for Internal Attribute
BUSINESS	Business
- Buttons: Add To List, Remove
- Maps to: [Text Field] Maps to [Dropdown]

---

## Importing Data From an ASCII Text File

Microsoft's ODBC Text driver can be used to export data from an ASCII file into an XML file for importing. This can be a useful way of importing data that is not stored in an ordinary SQL database; Oracle, SQL Server or MS Access, for example.

**Step 1.** Configure the `SCHEMA.INI` file for the ODBC driver.

1. Place the ASCII text file you will import in the same directory as the `SCHEMA.INI` file. The following text will be imported for this example:

```
FirstName,LastName,Address,Searchcode
Irvine,Welsh,49 Bird road,IWELSH
Jonathan,Cape,50 Roller Avenue,JCAPE
```

2. The `SCHEMA.INI` file is used to define the ASCII text format. Modify the `SCHEMA.INI` file for the ODBC driver as follows:

```
[sample.txt]
ColNameHeader=True
Format=CSVDelimited
MaxScanRows=1
CharacterSet=OEM
Col1=FIRSTNAME Char Width 255
Col2=LASTNAME Char Width 255
Col3=ADDRESS Char Width 255
Col4=SEARCHCODE Char Width 255
```

---

### NOTE

The Help files for the Microsoft Text driver contain detailed information on how to define your text file with the `SCHEMA.INI` file. From the Microsoft driver help, the topics "Defining Text Format" and "Schema.ini file" are particularly useful.

---

**Step 2.** Setup the ODBC Text driver.

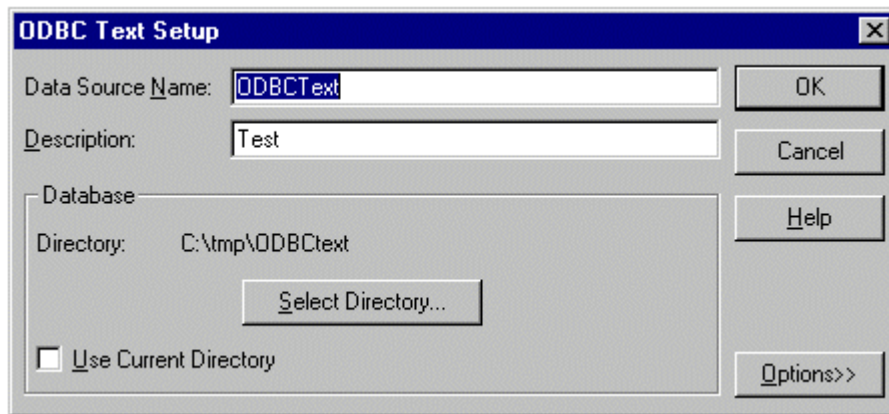
1. Start the ODBC Data Sources from the Windows Control Panel.
2. Select the `System DSN` tab and click `Add`.

## Examples

### Importing Data From an ASCII Text File

3. Select the Microsoft Text Driver, then click Finish.
4. Enter the Data Source Name.
5. Enter a Description of the file.
6. Clear the Use Current Directory check box and click Select Directory. The directory selected needs to be the same directory that the SCHEMA.ini file and the ODBCText file are located. The ODBC Text Setup dialog box should be set up similar to the following dialog box:

**Figure E-42 ODBC Text Setup**



### Step 3. Modify the configurable extractor (.ini) file.

1. Copy the configurable extractor `sd_event.ini` in Service Desk and save it with a new name. For example, `ODBCText.ini`.
2. Modify the configurable .ini file as follows:

```
[ DSN ]
NAME=ODBCText
USR=
PWD=

[ SYSTEM ]
LOG=TRUE
XML=TRUE
LOG_FILE=\ODBCText.log
OUTPUT_FILE=\ODBCText.txt
```



```
XML_OUTPUT_FILE=\ODBCtext.xml
APPLICATION_NAME=ODBCtext
ENCODING=ISO-8859-1

[CLASSES]
NAME=SAMPLE

[SAMPLE]
SOURCE=sample.txt
ATT=[LASTNAME],[FIRSTNAME],[ADDRESS],[SEARCHCODE]
COLUMNS=[LASTNAME],[FIRSTNAME],[ADDRESS],[SEARCHCODE]
LOADTABLE=TRUE
```

**Step 4.** Export the ASCII text file data.

1. Open the Data Exchange dialog box and select the Export data from a storage device check box.
2. In the Export Mapping field enter the name of the .ini file you modified. In this example it is ODBCtext.ini.
3. In the Exchange file field enter the name you specified for the XML file in the ODBCtext.ini file. In this example it is ODBCtext.xml.
4. Click OK to export the data into an XML file.

---

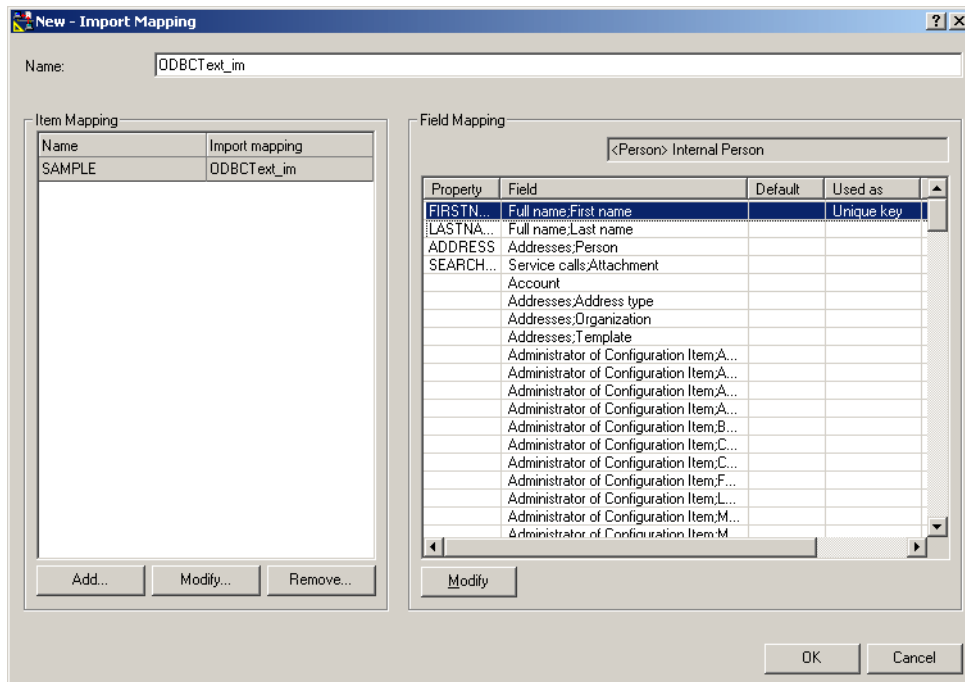
**NOTE**

You can also export the ASCII text file from a command line with:  
sd\_export -f odbctext.ini -x odbctext.xml -l odbctext.log

---

- Step 5.** Create an import mapping for the data. The following dialog box shows the import mapping for this example:

**Figure E-43** ODBC Text file - Import Mapping



**Step 6.** Import the data.

1. Open the Data Exchange dialog box.
2. In the Exchange file field, enter the location and name of the ODBCText.xml file.
3. Select the Import data into Service Desk database check box.
4. Enter a Service Desk account and password created for integration purposes.
5. In the Import mapping field use the drop-down arrow to find the import mapping you created for importing the Excel file. In this example it is ODBCText\_IM:
6. Select the Debug check box and click OK to import the data.
7. Check ODBCText\_IM\_imp.log to verify that the import was

successful.

---

**NOTE**

You can also import the XML file from a command line with: `sd_import odbctext.xml username/password odbctext_im Y odbctext.log c\temp\data_exchange.`

---

Examples  
Importing Data From an ASCII Text File

---

# Glossary

## A

**action** An operation carried out as a result of the activation of a rule and the successful evaluation of a rule or conditions within the rule.

**agent** A program or process running on a remote device or computer system that responds to management requests, performs management operations, and/or sends event notification.

**API** Application programming interface. An interface that enables programmatic access to an application.

**attribute** An object characteristic or property that describes the current state of the object.

**attribute value** A value assigned to one of the properties (attributes) that are associated with an object.

## B

**batch importing** The process of importing a group (batch) of data from an external data source into the Service Desk database.

## C

**CIM-XML** Common interface model extensible markup language.

**condition** A rule element consisting of a condition type and a set of parameter values to which a source value is compared to determine if the condition holds true or not, (eg. "IP Address is 15.2.112\*").

**configuration item** An item belonging to the technical infrastructure of an organization. A configuration item may consist of other configuration items, and may be part of other configuration items. For example, a PC, an application program, a network, a work space with desks and chairs. Configuration items are stored in the application database.

**class** *see object*

## D

**database rule** *see rule*

**daemon** A software process that runs continuously in the background and provides services upon request.

**data exchange file** A term used to refer to the files that are exported during the data exchange process. The data exchange process creates three data exchange files: a text file an XML file and a log file.

**DTD** Document type definition. A DTD is a set of syntax rules for mark-up tags and their

---

interpretation within an XML or HTML file. A DTD provides specific information about the tags used in the document; the order those tags should appear, which tags can appear inside other ones, and which tags have attributes, for example. A DTD defines the relationship between document elements.

## E

**event** An event is an unsolicited notification such as an SNMP trap, node down notification or TL1 event, generated by an agent or process in a managed object or by a user action. Events usually indicate a change in the state of a managed object or cause an action to occur.

**event communicator** Includes the `sd_event` program and the Service Desk agent, which are necessary for sending and receiving events.

**event queue** The process of putting multiple events in order so that they can be executed one by one.

**event storm** When hundreds of events occur at the same time it is referred to as an event storm. Queuing tools are used to organize the events so that they do not flood the system all at one time.

**extractor** A program used for

exporting 'extracting' information from a data source. The extractor is configurable through the use of an \*.ini file.

## H

**HTML** Hypertext markup language. A standard generalized markup language (SGML) document type definition (DTD) that provides a collection of platform-independent styles (indicated by markup tags) to define the various components of a World Wide Web (WWW) document.

**HTTP** Hypertext transfer protocol. The protocol that World Wide Web clients and servers use to communicate.

## I

**integration account** A Service Desk account created specifically for the purposes of integration. Integration accounts normally are not given access to the user interface.

**item** *See object*

## K

**key binding** The process of assigning at least one attribute the role of identification for an item. A typical key binding attribute is a search code or a serial number.

---

**key value** *See key binding*

## M

**mapping** The process of matching external classes, properties and values with Service Desk items, attributes and values. Mapping is done so that exported information can be formatted in such a way that it can be correctly imported into Service Desk.

**message** A structured readable notification that is generated as a result of an event, the evaluation of one or more events relative to specified conditions, or a change in application, system, network, or service events.

**message annotation** Message annotations can be written for each message, and often describe the actions that were taken to solve the problem. An annotation can be created and reviewed by the operator or administrator for any message.

**modus** The mode being used to make a change in a database. For example, inserting a record, deleting a record or updating a record.

## O

**object** A managed logical or physical resource, or a group of such physical resources that exist in a managed environment.

Example of objects are a network, a computer, an interface, and a printer.

**ODBC** Open database connectivity. Software that enables a program to communicate with a particular RDBMS. ODBC permits database independence by providing an interface to a set of drivers for the RDBMS' supported by OpenView.

## outbound service events

Events that are sent from the Service Desk application.

## P

**property** A characteristic or attribute of a class, object, or item.

## Q

**queue** A waiting line in which unsatisfied requests (events) are placed until a resource becomes available to execute them.

## R

**rule** A rule is the combination of one or more actions and the set of conditions that determine when the action will take place.

**rule manager** Maintains rules and is responsible for pulling event information from the event queue to determine if a rule exists that applies to the event.

---

**rule manager agent** *see agent*

## **S**

**service event** *see event*

**SNMP** Simple network management protocol. A protocol running above TCP/IP used to communicate network management information.

**SNMP trap** An unconfirmed event, generated by an SNMP agent in response to some internal state change or fault condition, which conforms to the protocol specified.

## **X**

**XML** Extensible markup language. Similiar to HTML except that it provides more semantic information. The file format is used to represent data, and for describing the data structure. The tags used make it possible to indicate what kind of data each tag contains, rather than indicating only how something should look.



**A**

- acceptance filter
  - ManageX, 195
- accounts, integration, 147
- aliases, 57
- ATT, 62, 63
- attributes
  - import mapping, 95
- auditing, 163

**C**

- case sensitivity, 170
- change log, 132
- CIM-XML, viewing, 90
- classes
  - configurable extraction wizard, 52
  - import mapping, 95
- columns, 59, 60
- COM objects
  - ManageX, 201
- command syntax, 144
  - service events, 151
- configuration
  - extractor, 45
  - importing, 95
  - keywords, 47
- configuration file
  - errors, 170

**D**

- Data Exchange, 72
- data exchange
  - exporting data, 83
  - exporting overview, 36
  - files, 86
  - import mapping overview, 37
  - importing data, 119
  - importing overview, 38
  - overview, 33
  - scheduling, 125
- database rules
  - ManageX, 194
- database structure, checking, 43
- database user, 54
- debug mode, 164
- delta processing, 131
- document conventions, 26
- DSN, 170
  - configurable extraction wizard, 52
- duplicate columns, 62

**E**

- encoding, 56
- equal sign, 170
- error log, 156
- errors
  - file locations, 165
  - scalable importing, 142
  - troubleshooting, 166
- event storms, 158
- example service events. *See* service events
- examples
  - exported XML file, 87
  - LDAP integration, 203
  - ManageX, 189
  - queuing, 202
  - Radia integration, 217
- export files, 86
- exporting
  - command syntax, 144
  - data, 45
  - new task, 85
  - procedures, 83
  - process, 81
  - what to export, 43
- extractors
  - configuration, 45
  - keywords, 47

- F**
  - free text, 61

- G**
  - grouping tasks, 126

- I**
  - import mapping
    - classes, 95
    - relations, 96
    - templates, 95
    - values, 113
  - importing
    - command syntax, 144
    - LDAP directory, 215
    - tasks, 119
  - importing mapping
    - attributes, 95
  - ini file, 53
  - integration accounts, 147
  - ISO-8859-1, 56

---

---

# Index

## J

Join syntax, 65

## K

key binding, 95  
key values, mapping, 95  
keywords, 47

## L

LDAP  
  configuration, 207  
  configuration file, 207  
  installation, 206  
Lightweight Directory Access Protocol (LDAP), 203  
load balancing, 135  
log file, 55  
log files, 164

## M

ManageX  
  acceptance filter, 195  
  COM objects, 201  
  database rules, 194  
  multiple servers, 193  
  overview, 189  
  string substitutes, 199  
multiple servers  
  ManageX, 193

## N

Network Node Manager  
  command line, 178  
  ODBC, 179  
  variables, 182

## O

ODBC (open database connectivity), 79  
ODBC data source  
  establish link, 79  
  extraction configuration wizard, 52, 53  
  Network Node Manager, 179  
  Radia example, 219  
operator, 68  
Oracle database, 66

## P

parent relation, 52, 62

password, 54

## Q

queuing events, 158  
queuing example, 202

## R

Radia integration  
  about, 217  
  configuration, 219  
  installation, 218  
  opening inventory browser, 231  
real values, 62  
reconciliation, 130  
related publications, 22  
relating tasks, 126  
relations, import mapping, 96  
required fields, 96  
resending service events, 156

## S

scalable importing, 135  
  errors, 142  
  task group, 141  
scheduling tasks, 125  
sd\_event.exe, 151  
service events  
  command line, 151  
  NNM example, 177  
  queuing, 158  
  resending, 156  
  switches, 151  
SQL, 51, 61  
SQL statements, 81  
string substitutes  
  ManageX, 199  
switches, service events, 151

## T

task group, 126  
tasks  
  exporting, 85  
  scheduling, 125  
templates, import mapping, 95  
test connection, 54  
text editor, 53  
troubleshooting  
  debug mode, 164  
  log files, 164  
  program files, 165

viewing XML files, 90  
typographic conventions, 26

**V**

value mapping, 113  
variables  
    Network Node Manager, 182  
viewing, XML files, 90

**X**

XML, 53, 55, 72  
XML (extensible markup language)  
    about, 86  
    CIM-XML, 86  
    example XML file, 87  
    export files, 86  
    viewing export files, 90