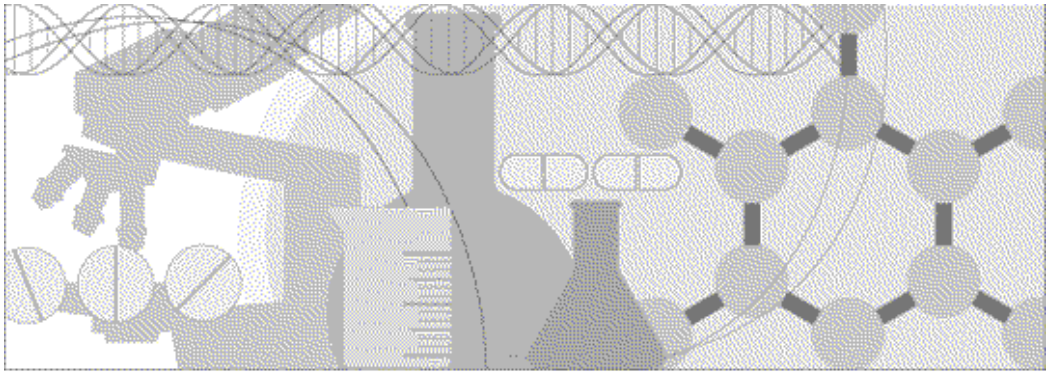
InForm Utilities Guide

InForm™

Release 4.6 SP3



ORACLE

Part number: DC-INF46-004-030

Copyright and Trademark Notice

Phase Forward is a provider of integrated enterprise-level software products, services, and hosted solutions for use in the clinical trial component of its customers' global research and development initiatives. Phase Forward is headquartered in Waltham, Massachusetts with offices in the United Kingdom, Australia, France, India, and Japan. Additional information about Phase Forward is available at www.phaseforward.com.

Phase Forward
77 Fourth Avenue
Waltham, MA 02451

Third-party Trademarks

Restricted Rights Legend

Export Notice

Phase Forward products and services may be subject to export controls under applicable export control regulations, including without limitation, the United States Export Administration Regulations, and you agree to comply with any such regulations. You agree not to knowingly export or re-export any Software, Documentation, technical data or know-how, directly, or indirectly, into Iran, Iraq, Libya, Cuba, North Korea, Sudan, and Syria, or to any of its nationals or to any other countries or individuals of concern to the United States government.

Contents

About this guide vii

Overview of this guide viii
Audience viii
Related information ix
Documentation ix
Training xi
If you need assistance xii

Chapter 1 InForm overview 1

About the InForm application 2
InForm system configuration 3

Chapter 2 About InForm utilities 5

Overview of InForm utilities 6

Chapter 3 Using the PFConsole utility 7

Overview of the PFConsole utility 8
Running the PFConsole utility 9
Example 9
Using the PFConsole utility within a script 10

Chapter 4 Using the InForm Data Import utility 11

Overview of the InForm Data Import utility 12
Parts of the InForm Data Import utility 12
Import methods 12
Special considerations 13
Importing a data and map file 15
Creating a data and map file 15
Specifying and editing map files 17
Specifying a submission type 19
Specifying an input field type 20
Building an item path 21
Indicating that data contains multiple selection items 23
Checking for duplicate information within itemsets 24
Importing information into an unscheduled visit 24
Specifying a data type 25
Mapping strings and child controls 25
Navigating the map file 27
Inserting or deleting an import field 28
Checking the map against the import file 28
Running the import using the data and map import file 29
Saving the map file 31
Checking the error file 32
Importing an XML file 33
XML import files 33
Creating an XML file for data import 33
Screening 36

Enrolling 36
Adding new patient clinical data 37
Updating existing patient clinical data 42
Transferring a patient record 47
Running the import using the XML import file 50
Importing an XML file 54
Importing coded items 55
Creating an autocode import file 55
Running the import using an autocode import file 55
Date and time validation 60
Additional InForm Data Import attributes 61
Running the InForm Data Import from the command line 62
Enhancing your data import 64
Log off of the InForm application server 64
Stop the WWW Publishing Service 64
Run the Oracle update statistics script 64
Change the home page 64
Organize by patient 64

Chapter 5 Using the MedML Installer utility 65

Overview of the MedML Installer utility 66
Setting up a trial with the MedML Installer utility 67
Validation checks 68
Selection value checks 68
Coding mapping checks 68
Launching the MedML Installer utility 70
About the MedML Installer utility window 71
Running the MedML Installer utility 73
Running the MedML Installer utility for the first time 73
Updating a trial that is already in progress 73
Removing XML files from the build 74
About MedML Installer utility output messages 75
Running the MedML Installer utility from the command line 76
Command line parameters 76

Chapter 6 Using the InForm Data Export 79

Overview of the InForm Data Export utility 80
InForm Data Export output options 80
Running the InForm Data Export utility 81
Exporting coded controls 83
Required elements 83
Running the export for AutoCode items 84
Exporting data into a CDD 87
Overview 87
Moving data to a CDD 87
Running the export for CDD data 88
Exporting Name Value Pairs 90
Output file format 91
Output file formatfor associated forms 91
Example 91
Exporting data in Oracle Clinical format 93
Overview 93
Transferring data to the Oracle Clinical upload application 93
Running the export in Oracle Clinical format 93
Running the InForm Data Export from the command line 98

Example 100
Oracle Clinical fields 101
Format of output files 102

Chapter 7 Using the InForm Performance Monitor utility 103

Overview 104
Starting the InForm Performance Monitor utility 105
Capturing performance statistics 106
Viewing messages from specific subsystems 107
Selecting InForm servers 108
Performance Monitor output options 109
Managing the InForm Performance Monitor data 110
Examples of using the InForm Performance Monitor utility 111
Testing rule script efficiency 111
Reviewing SQL query performance 113

Chapter 8 Using the InForm Report Folder Maintenance utility 117

Overview 118
Folder structure for multiple trials or sponsors 119
Setting up the initial folder structure 119
Folder structure for multiple trials 120
Folder structure for multiple sponsors, multiple trials 122
Setting up a folder structure for multiple trials or sponsors 124
Setting up reporting packages 124
Creating new folders for multiple trials or sponsors 125
Copying report folders 127

Appendix A Sample Data Import XML 129

Overview 130
Importing screening and enrollment data 131
Importing new patient clinical data 132
Updating existing patient clinical data 133
Importing new itemset data 134
Editing an existing itemset 136
Deleting data from an itemset 137
Undeleting data from an itemset 138
Adding data to an unscheduled visit 139
Transferring patient records 140

Index 141

About this guide

In this preface

Overview of this guide viii
Related information ix
If you need assistance xii

Overview of this guide

The *Utilities Guide* provides information about and step-by-step instructions for using the following utilities:

- PFConsole utility
- MedML Installer utility
- InForm Data Import utility
- InForm Data Export utility
- InForm Performance Monitor utility
- InForm Report Folder Maintenance utility

Audience

This guide is for trial designers and system administrators who need to move data into and out of the InForm database.

Related information

Documentation

All documentation is available from the Phase Forward Download Center.

Item	Description
Release Notes	The *Release Notes* document describes enhancements introduced and problems fixed in the current release, upgrade considerations, release history, and other late-breaking information.
Known Issues	The *Known Issues* document provides detailed information about the known issues in this release, along with workarounds, if available. **Note:** The most current list of known issues is available on the Phase Forward Extranet. To sign in to the Extranet, go to www.phaseforward.com and click **Customer Login**. Enter your email address and password, and navigate to the **Known Issues** section. Select a product, and then enter your search criteria.
Installation and Configuration	The *Installation and Configuration* guide describes how to install the software and configure the environment for the InForm application and Cognos 8 Business Intelligence. This document is also available from the Documentation CD.
Setting Up a Trial with InForm Architect and MedML Guide	The *Setting Up a Trial with InForm Architect and MedML Guide* describes how to design and implement trials in the InForm application using the InForm Architect application. This document is also available from the Documentation CD.
Step by Step for CRCs and CRAs	The *Step by Step for CRCs and CRAs Guide* describes how to use the InForm application to: • Screen and enroll patients. • Enter, update, and monitor clinical data. • Enter and respond to queries. • Run trial management reports and clinical data listings. This document is also available from the Documentation CD and the user interface.

Item	Description
Reporting and Analysis Guide	The *Reporting and Analysis Guide* provides an overview of the Reporting and Analysis module. It includes a brief overview of the Reporting and Analysis interface, illustrates how to access the Ad Hoc Reporting feature, and describes the study management and clinical data packages available for reporting. It also provides detailed descriptions of each standard report that is included with your installation. This document is also available from the Documentation CD and the user interface.
Utilities Guide	The *Utilities Guide* provides information about and step-by-step instructions for using the following utilities: • PFConsole utility • MedML Installer utility • InForm Data Import utility • InForm Data Export utility • InForm Performance Monitor utility • InForm Report Folder Maintenance utility This document is also available from the Documentation CD.
Reporting Database Schema	The *Reporting Database Schema* Guide describes the InForm reporting database schema. This document is also available from the Documentation CD.
Portal Administration Guide	The *Portal Administration Guide* provides step-by-step instructions for setting up the InForm Portal software, and configuring and managing the InForm Portal application. This document is also available from the Documentation CD.
Online Help	The online Help describes how to use and administer the InForm application. This document is available only from the user interface.
InForm Architect online Help	The InForm Architect online Help describes how to design and implement trials in the InForm application using the InForm Architect application. This document is available only from the user interface.
MedML Installer utility online Help	The MedML Installer utility online Help provides information about, and step-by-step instructions for using, the MedML Installer utility, which is used to load XML that defines study components into the InForm database. This guide also provides reference information for the MedML elements and scripting objects that are used to import and export data to and from the InForm application, as well as sample data import XML. This document is available only from the user interface.

Item	Description
InForm Data Export utility online Help	The InForm Data Export utility online Help provides information about and step-by-step instructions for using the InForm Data Export utility, which is used to export data from the InForm application to the following output formats: • AutoCode. • Customer-defined database (CDD). • Name value pairs. • Oracle Clinical. This document is available only from the user interface.
InForm Data Import utility online Help	The InForm Data Import utility online Help provides information about and step-by-step instructions for using the InForm Data Import utility, which is used to import data into the InForm application. This document is available only from the user interface.

Training

For information about training offerings, please see the course catalog on our website at http://www.phaseforward.com/support/training/ or contact Phase Forward Educational Services at training@phaseforward.com.

If you need assistance

If you are a Phase Forward customer with a maintenance agreement, you can contact the Global Support Center for assistance with product issues.

Your maintenance agreement indicates the type of support you are eligible to receive and describes how to contact Phase Forward. Additionally, the Phase Forward website lists the toll-free support number for your product, location, and support level:

```
http://www.phaseforward.com/support/
```

In the event that our toll-free telephone service is interrupted, please use either of the following methods to contact the Global Support Center:

- Email

  customer.support@phaseforward.com

- Telephone

In the US:	781-902-4900
Outside the US:	+44 (0) 1628 640794

Phase Forward also provides assistance with User Management, Site Assessment, and Provisioning. Please refer to you Master Services Agreement and individual Statement of Work to determine if you are eligible to use these services.

CHAPTER 1

InForm overview

In this chapter

About the InForm application 2
InForm system configuration 3

About the InForm application

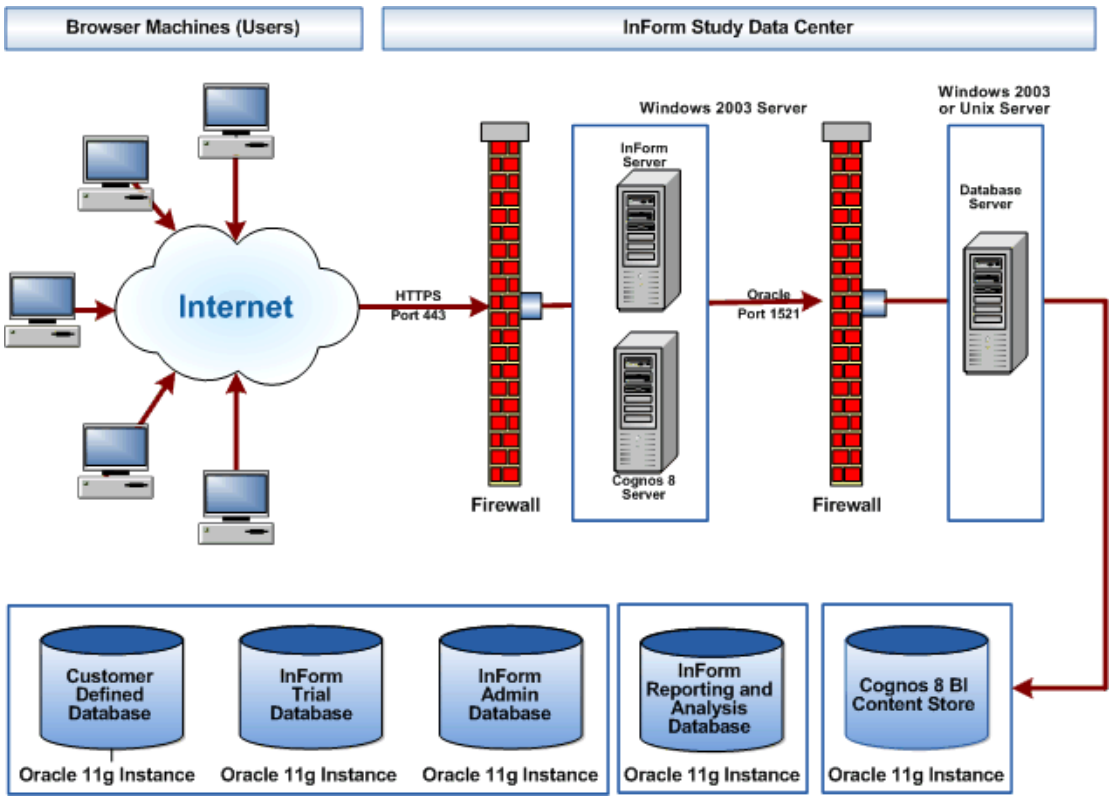
The InForm application is a data collection and trial management application that uses a secure web browser to provide access to clinical trial data and management of the clinical trial process.

Embedded within the InForm user interface is the Reporting and Analysis module, which is a reporting application that was developed by the Cognos Corporation, and that has been customized and integrated into the InForm application by Phase Forward. The Reporting and Analysis module provides a library of configurable reports, predefined reports, and ad hoc reporting and charting tools. Both clinical and operational data are available in real time from a database that can be accessed using the Internet.

Note: You can install the InForm software without the Reporting and Analysis module if you want to use the data collection features of the InForm application without the reporting features.

InForm system configuration

The InForm system configuration consists of the browser computers, application server, and database server.



CHAPTER 2

About InForm utilities

In this chapter

Overview of InForm utilities 6

Overview of InForm utilities

Utility	Description
PFConsole utility	Enables you to run the InForm Data Import utility, InForm Data Export utility, and MedML Installer utility from a command line. For more information, see *Using the PFConsole utility* (on page 7).
MedML Installer utility	Installs the metadata definition of your trial into the InForm application. For more information, see *Using the MedML Installer utility* (on page 65).
InForm Data Import utility	Imports patient data into the InForm database. The following options are available: • MedML format. • CSV format, with which you can optionally run rules. For more information, see *Using the InForm Data Import utility* (on page 11).
InForm Data Export utility	Exports patient data from the InForm database into the following formats: • **Customer-Defined Database (CDD)**—Exports data into a CDD. • **Name Value**—Creates a comma-delimited file that consists of data path names and data values. For more information, see *Using the InForm Data Export utility* (on page 79).
InForm Performance Monitor utility	Provides statistics about several types of activities that help with performance tuning during the process of developing and implementing a trial. For more information, see *Using the InForm Performance Monitor utility* (on page 103).

CHAPTER 3

Using the PFConsole utility

In this chapter

Overview of the PFConsole utility 8

Running the PFConsole utility 9

Using the PFConsole utility within a script 10

Overview of the PFConsole utility

The PFConsole utility consolidates all the activities of each tool within one window from which you can run:

- The InForm Data Import.
- The InForm Data Export.
- The MedML Installer utility.

Note: **Phase Forward strongly recommends that you use the PFConsole utility to run these utilities from the command line. Running them individually in the command-line window is not recommended.**

Running the PFConsole utility

Use the following command to open the PFConsole utility and run an InForm utility.

```
pfconsole <application> -autorun <parameters>
```

where:

- *pfconsole* starts the PFConsole utility.
- <*application*> is one of the following:
  - *pfmminst* for the MedML Installer utility.
  - *pfimport* for the InForm Data Import.
  - *pfexport* for the InForm Data Export.
- *-autorun* is a required parameter of pfconsole.
- <*parameters*> are the variables for the individual utility.

For specific parameters for each utility, see:

- ***Running the MedML Installer utility from the command line*** (on page 76).
- ***Running the InForm Data Import from the command line*** (on page 62).
- ***Running the InForm Data Export from the command line*** (on page 98).

Example

```
pfconsole pfmminst -trial pfst46 -verbose -autorun -outfile text.log -xml
filename.xml
```

Using the PFConsole utility within a script

If you are running several imports or exports, you can generate a script that will run the utilities in batch mode. To pause the script until the application completes and then moves to the next command, use the start command:

`start /wait <pfconsole command line>`
where:

<*pfconsole command line*> is the string of commands you specified when you ran the PFConsole utility.

For more information, see ***Running the PFConsole utility*** (on page 9).

CHAPTER 4

Using the InForm Data Import utility

In this chapter

Overview of the InForm Data Import utility 12

Importing a data and map file 15

Importing an XML file 33

Importing an XML file 54

Importing coded items 55

Date and time validation 60

Additional InForm Data Import attributes 61

Running the InForm Data Import from the command line 62

Enhancing your data import 64

Overview of the InForm Data Import utility

Parts of the InForm Data Import utility

Part	Description
Data file	A pipe-delimited file that contains the following: • Line feed characters and carriage returns between lines. • Data to load into the trial database.
Map file	A file that contains all the mapping information that is necessary to import information from the data file to the InForm database.
XML file	A file that contains information that can be directly imported into the trial database. For more information, see *Creating a data and map file* (on page 15) and *Adding new patient clinical data* (on page 37).
Rules	Information against which the XML file is checked to ensure that it complies with the trial standards. You use rules to: • Avoid collisions during data entry. • Check the validity of data before it is committed to the trial database. **Note:** Running rules might cause the import to run slowly.
InForm application server	The server on which the InForm software is running.
Trial database	The database in which the data for the trial resides.

Import methods

The following data loading methods are available in the InForm Data Import utility:

- **Importing a data file**—Uses a map file to define mappings between the imported data and the InForm database tables. The InForm Data Import utility then loads the data from a pipe-delimited (\|) file into the InForm database.

  When you use this method, the data is processed by the InForm application server, which runs edit checks to validate the data before writing to the database.

- **Importing an XML file**—Loads data from an XML file into the InForm database or transfers selected patient records from one site to another.

  When you use this method, you can run rules during the import. If you run rules, the application processes the data as if you were entering it online; it runs edit checks and generates queries on data that fails the checks.

- **Importing an AutoCode XML file**—Similar to the MedML import option. Use this option

for autocoding. For more information, see *Exporting coded controls* (on page 83).

Note: The InForm Data Import utility does not support importing RegDocs or Visit Reports data.

Special considerations

- **Importing new patient data**—You must use the XML file option to import data for a subject who has not gone through the screening and enrollment process. After importing the data, you must stop and restart the trial to view the data online with the InForm application.

  You cannot use the data and map file option to import screening and enrollment data. With the InForm Unplugged application, you can import screening and enrollment data only to a site server.

  Note: You cannot import new data to a form with a **Frozen** or **Locked** status in the InForm application.

- **Importing comments**—To import form-level comments, you must use the XML file option.

  To import item-level comments, you must use either the data and map file option or the XML file option.

- **Importing calculated controls**—When you import calculated controls, the data type definition of the import field must be text control. If integer or floating number is used, the match is not recognized and data is not updated. Instead, the data is added as a new row.

- **Importing units**—When you submit unit data, the units must be associated with the previous field in the data and map file.

- **Importing unscheduled visit data**—To import data to unscheduled visits, use either the XML file or data and map file option.

- **Deleting and undeleting itemsets**—To delete or reinstate itemset data, you must use the XML file option.

- **Importing on multiple servers**—If your trial is running on multiple servers, run the InForm Data Import utility on only one server at a time to eliminate the possibility of importing duplicate data. Once the data is imported, use synchronization to distribute it to other servers.

- **Editing repeating forms**—To create or edit repeating form instances, you must create the MAP file using a text editor, not the InForm Data Import user interface. Follow these guidelines for editing a map file:

  - Use NOFORMNEW in the first line of the MAP file to instruct the InForm Data Import utility not to create a new repeating form instance.

  - Use the new !formmatch! element in the MAP file to specify an item (not within an itemset) that will be compared to other repeating form instances.

    You can use multiple !formmatch! elements. If all such elements match some existing repeating form instance, then that form instance will be updated. If there is no match (or no !formmatch! element) then a new repeating form instance will be created, as

illustrated in the following sample code:

```
FORMNOVISITNEWNOFORMNEW|
!cd!Site|
!cd!Patient|
!visitmatch!0.UnschVisit.DOV.DOV.0.DOV.DOV!dtdatetime!|
!formmatch!0.UnschVisit.HH.DH.0.DURATIONGROUP.DURTAIONGROUP.YRD
URATION!dtstring!|
!formmatch!0.UnschVisit.HH.DH.0.DURATIONGROUP.DURTAIONGROUP.MTH
DURATION!dtstring!|
0.UnschVisit.HH.DH.0.previousgroup.previousgroup!DTSTRING!|
```

- When you create a repeating form, include regular items instead of itemsets to uniquely identify the form. If you cannot uniquely identify the form, then you can enter data only once to the form, and the next data entry will go to a new form instance.

Importing a data and map file

Creating a data and map file

The data and map file that used in the direct import method must be a text file. You can use any text editor that creates plain text files to create it, or develop a conversion tool that automatically formats your raw data.

You can direct the data in the file to target controls on more than one CRF; however, you must use separate files for each itemset into which you are importing rows of data, and data that you import into CRF itemsets must be in a separate file from data that you import into regular CRF items.

The import file must have the following characteristics.

- Each import row must include either of the following:
  - A field that specifies the patient number and initials in the following format:

    patient_number (patient_initials)

    If it is possible that more than one patient could have the same patient number and initials, you must also include a field for the site mnemonic of the patient.
  - The database ID of the patient.
- All import fields must be separated by a pipe character (|).
- Do not include double quotation marks (") in the data file.

Additionally, some types of data must be presented in specific ways. The following sections describe how to set up the following fields and controls for import:

- *Date fields* (on page 15).
- *Time fields* (on page 16).
- *DateTime fields* (on page 16).
- *Nested Controls* (on page 16).
- *Checkboxes and multiple-selection drop-down lists* (on page 16).
- *Units* (on page 16).
- *Item comments* (on page 17).

Date fields

Dates must consist of three fields in month|day|year format, using a 4-digit year. Observe these formatting considerations:

- If a date is missing a component, include a null field, as in the following example:

```
month||year
```

- If a component of a date is unknown, use the keyword UNK, as in the following example:

```
month|UNK|year
```

Time fields

Times must consist of three fields in hour|minute|second format, using a 24-hour clock. Observe these formatting considerations:

- If a time is missing a component, include a null field, as in the following example:

```
hour|minute|
```

- If a component of a time is unknown, use the keyword UNK, as in the following example:

```
hour|UNK|UNK
```

DateTime fields

Dates and times must consist of six fields in month|day|year|hour|minute|second format, using a 4-digit year and a 24-hour clock. Observe these formatting considerations:

- If a datetime data item is missing a component, include a null field.
- If a component of a datetime item is unknown, use the keyword UNK.

Example:

```
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA|Clinical Chemistry:|AG
Ratio|XGR|Nov|UNK|1998|0.8-2||2.0|N|||
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA|Clinical
Chemistry:|ALAT(SGPT)|XGP|Nov|UNK|1998|0-48|U/L|42|N|||
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA|Clinical
Chemistry:|Albumin|XAL|Nov|UNK|1998|3.2-5|G/DL|4.3|N|||
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA|Clinical Chemistry:|Alkaline
Phosphatase|XLK|Nov|UNK|1998|20-125|U/L|63|N|||
```

Nested Controls

Nested controls must consist of a field for each control separated by a pipe character (|). Only one of these fields will contain information for each row of data.

Example:

```
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA|hematology|||
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA||urinalysis||
PF|001 (CJB)|Sep|23|1975|1|1|215210|LABDATA|||toxicology|
```

Checkboxes and multiple-selection drop-down lists

Checkboxes and multiple-selection drop-down lists must consist of a field with a list of zero or more controls separated by a comma.

Example:

```
PF|001 (CJB)|Sep|23|1975|LABDATA|Sinus Tachycardia, Premature Ectopic
Junctional Beats, Sinus Bradycardia, Premature Ectopic Atrial Beats|
```

Units

A field that contains units must immediately follow the value that it describes.

The example shows weight in pounds. The number value of the weight (177) is contained in one field, and the unit value (pounds) is contained in the following field.

```
PF|001 (CJB)|Sep|23|1975|70|in|177|lb|
```

Item comments

Item comments must consist of a field that contains the text comment for the item.

Example:

```
PF|001 (CJB)|Sep|23|1975|70|177|weight measured with shoes and socks
```

Specifying and editing map files

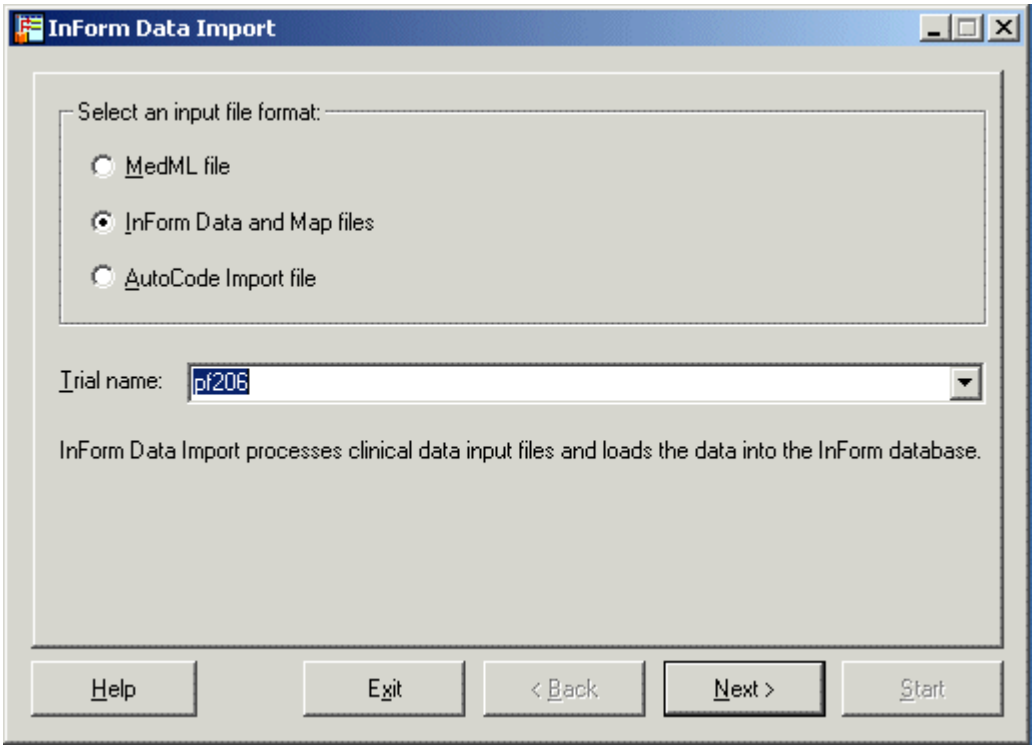
To import a data file, do one of the following:

- Specify an existing map file.
- Define a map file.

To specify and edit an existing import map file:

1 Select **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm Data Import**.

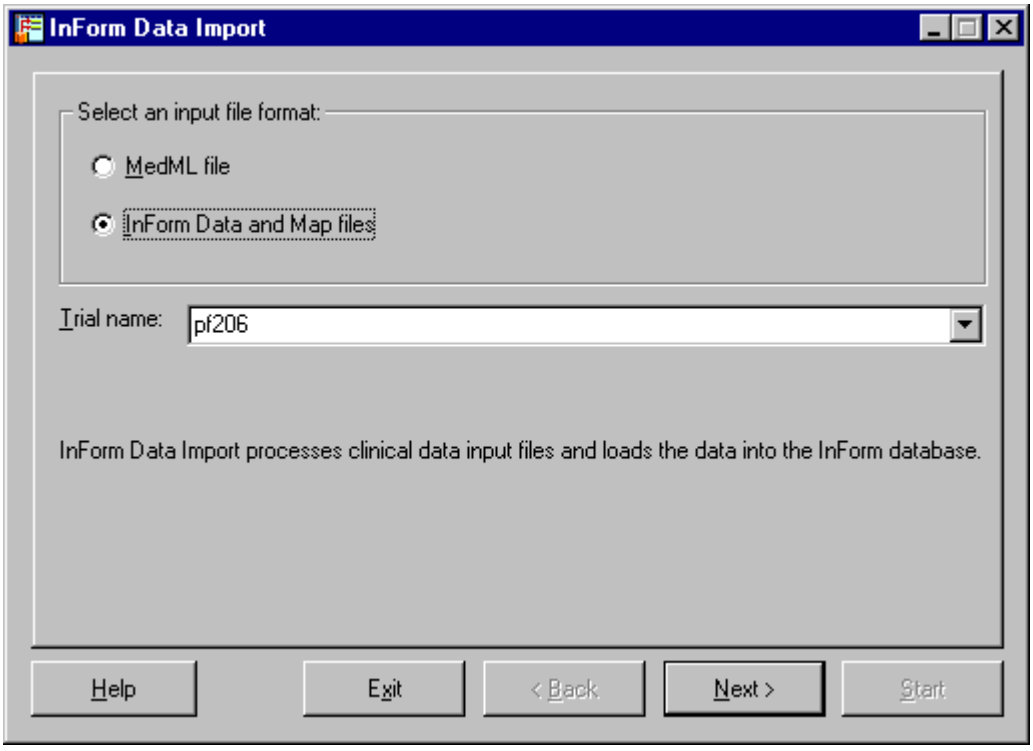
The InForm Data Import main window appears.



2 Select **InForm Data and Map files**.

3 In the **Trial Name** field, type or select the name of the trial into which to import the files.

4 Click **Next**.



5 Type the file name of the data file to import.

6 Type the file name of the map file to import.

Note: After you enter a file name and open the map file editor, the file is created, regardless of whether you click **Start** or **Stop**. This file is stored in the same location from which pfimport.exe was opened.

7 Click **Edit Map File**.

To create or edit a map file:

- Specify whether the data is targeted for CRF items or CRF itemsets. For more information, see *Specifying a submission type* (on page 19).
- Specify the input field type. For more information, see *Specifying an input field type* (on page 20).
- Specify the definition of each map file, one field at a time. For more information, see *Building an item path* (on page 21).
- Indicate whether the data contains multiple selection items. For more information, see *Indicating data contains multiple selection items* (on page 23).
- Specify the data type. For more information, see *Specifying a data type* (on page 24).
- Specify mappings between the values in your import file and the values defined for the target data fields in the InForm database. For more information, see *Mapping strings*

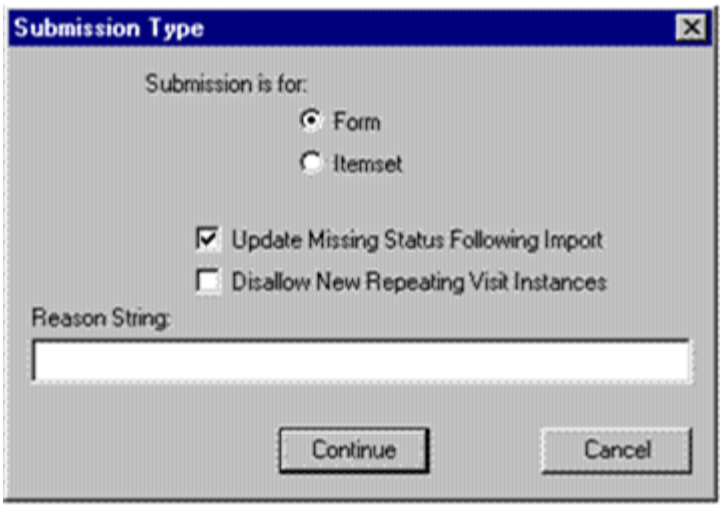
and child controls (on page 25).

- Save the map file. For more information, see *Saving the map file* (on page 31).

8 After you have saved the map file, exit the map file editor to return to the InForm Data Import window to import the data and map file. For more information, see *Running the import using the data and map import file* (on page 29).

Specifying a submission type

Use the Submission Type window to specify a submission type for the import file.



To specify a submission type:

1 Select one of the following options:

- **Form**—If all of the data in the import file is targeted for regular CRF items.
- **Itemset**—If all of the data in the import file is targeted for CRF itemsets. Optionally, to specify that you want to import only data that updates existing itemsets, select **Disallow New Itemset Rows**.

2 If you want to automatically update the traffic lights on a form when the import is complete and the trial is restarted, select **Update Missing Status Following Import**.

3 If you want to import data that only updates existing unscheduled visits, select **Disallow New Repeating Visit Instances**.

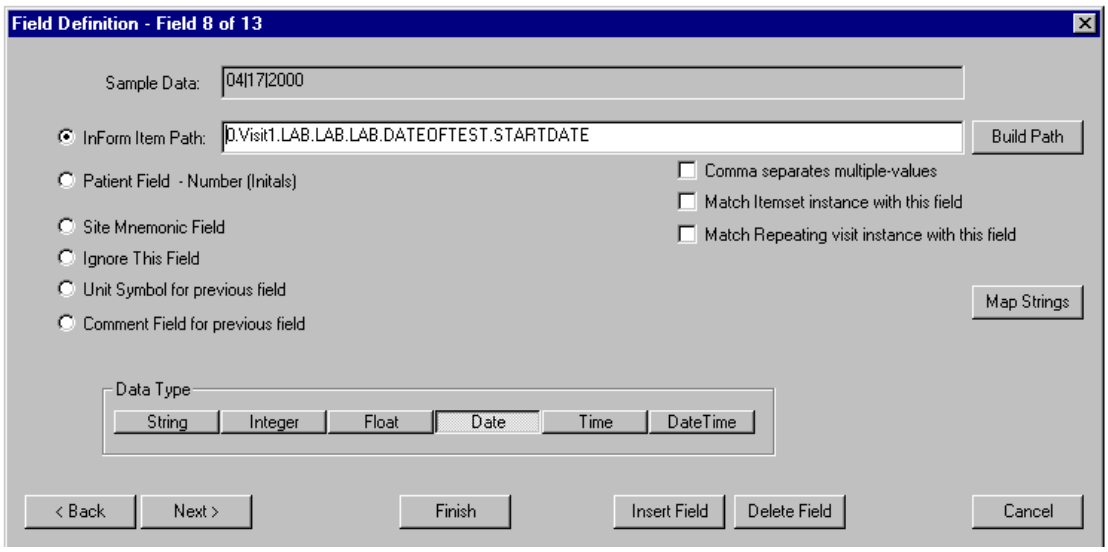
4 In the **Reason String** field, type the text that should appear in the **Reason for Change** section on the Data Value(s) screen if data is updated in the data load. The default reason is Lab Import.

5 Click **Continue**.

The Field Definition dialog box appears. Follow the procedure in *Specifying an input field type* (on page 20) to define map fields.

Specifying an input field type

Use the Field Definition dialog to define each map field one at a time. Each map field corresponds to a data item in the import file.



In the Field Definition dialog box, select one of the following field types:

- **InForm Item Path**—Contains data that is targeted to a data item on a CRF. When you select this option, you must provide additional information about the import field. For more information, see *Building an item path* (on page 21).
- **Patient Field – Number (Initials)**—Contains patient identification information in either patient_number (patient_initials) format or in the form of a patient database ID. If your import data identifies each patient by the patient database ID, select the **Field Contains Known Patient ID** checkbox.

  When you select this option, the definition of the map field is complete, and you can move to another map field definition or save the map file and exit the map file editor. For more information, see *Navigating the map file* (on page 27).
- **Site Mnemonic Field**—Contains the mnemonic of the site where the patient is enrolled. When you select this option, the definition of the map field is complete, and you can move to another map field definition or save the map file and exit the map file editor. For more information, see *Navigating the map file* (on page 27).
- **Ignore This Field—**Indicates that you do not want the field to be imported. When you select this option, the definition of the map field is complete, and you can move to another map field definition or save the map file and exit the map file editor. For more information, see *Navigating the map file* (on page 27).
- **Unit Symbol for previous field**—Contains the symbol for the units that apply to the previous field in the import and map files. The symbol of a unit is the text that identifies the unit on the CRF, as defined in the SYMBOL attribute of the UNIT definition in the appropriate XML file. When you select this option, the definition of the map field is complete, and you can move to another map field definition or save the map file and exit the

map file editor. For more information, see *Navigating the map file* (on page 27).

- **Comment Field for previous field**—Contains the comment text that is associated with a form or itemset. When you select this option, the definition of the map field is complete, and you can move to another map field definition or save the map file and exit the map file editor. For more information, see *Navigating the map file* (on page 27).

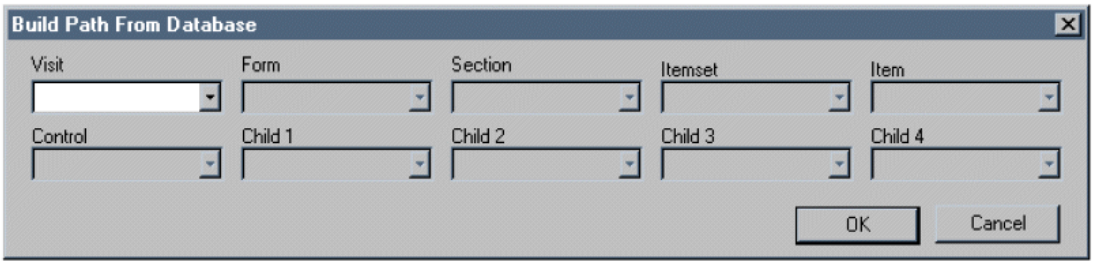
Building an item path

If you selected **InForm Item Path** as the item type, you must specify the RefName path for the target CRF data item for the field in the import file.

To build an item path, select each RefName from the drop-down lists in the Build Path from Database dialog, or type the path in the **InForm Item Path** text box.

Using the Build Path from Database dialog box

To specify an item path, click **Build Path**. The Build Path from Database dialog appears.



To build an item path:

1 From the **Visit** drop-down list, select the RefName of the target visit.

2 From the **Form** drop-down list, select the RefName of the target CRF.

3 From the **Section** drop-down list, select the RefName of the target section.

4 Optionally, from the **Itemset** drop-down list, select the itemset RefName to load the import data into an itemset.

5 From the **Item** drop-down list, select the RefName of the target item.

> **Note: You must create a separate map field definition for each item in an itemset.**

6 From the **Control** and **Child Control** drop-down lists, select the RefName of the target group and child controls.

7 Click **OK**.

Entering an item path explicitly

To specify an item path, use the following item path:

```
0.Visit.Form.Section.Itemset.Item[.control[.control…]]
```

Each component of the item path is the RefName used to define an element:

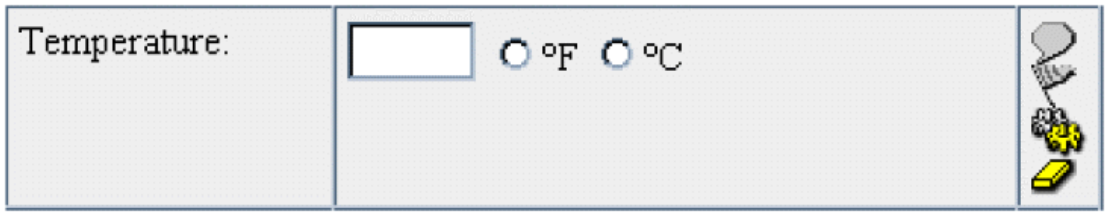
- **0**—Indicates the current patient.
- **Visit**—RefName of the visit, as specified in the XML file that contains the visit definition.
- **Form**—RefName of the CRF or other form, as specified in the XML file that contains the form definition.
- **Section**—RefName of the section, as specified in the XML file that contains the section definition.
- **Itemset**—RefName of the itemset, as specified in the XML file that contains the itemset definition. If the import data for which you are creating a map field definition is a regular CRF item, not an itemset, type 0.
- **Item**—RefName of the item, as specified in the XML file that contains the item definition. Create a separate map field for each item in an itemset.
- **Control**—RefName of the control, as specified in the XML file that contains the control definition. To access an element of a group control, refer to each parent control in which the child element is nested. For example, to address one of two text controls within a group control, type the RefName of the group control followed by the RefName of the text control, and separate the names with periods, as follows: GroupControlRefName.TextControlRefName.

Examples

- This example shows the item path for the TEMPTEXT field in the TEMPTEXT item in the VS section of the VSL form in a visit called VISIT1. Note the substitution of 0 for the Itemset RefName.

```
0.Visit1.VSL.VS.0.TEMPTEXT.TEMPTEXT
```

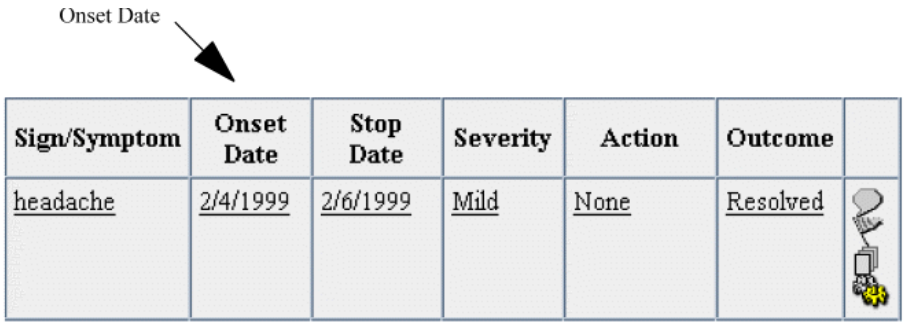
The online control appears as follows:



- This example shows the item path for an item in an itemset. The control is a date control called ONSETDATE, in the ONSETDATE item of the SS2GROUP itemset in the SECTION2 section of the SS form in VISIT1.

```
0.Visit1.SS.SECTION2.SS2GROUP.ONSETDATE.ONSETDATE
```

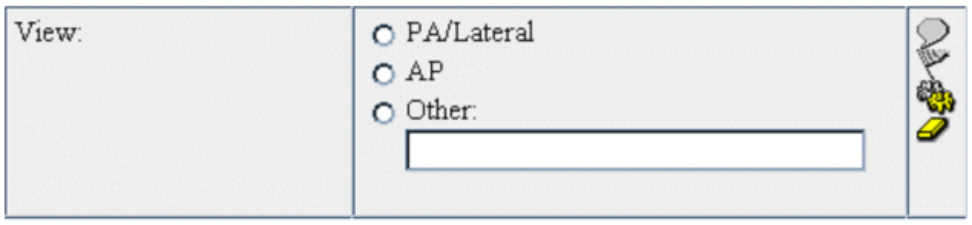
The online control appears as follows:



- This example shows the item path for a text control nested within the VIEW radio control item in the CHESTXRAY section of the ECG form in VISIT1. The item is identified by the VIEW RefName; the radio control is identified by VIEWRADIO, and the text control is identified by OTHERTEXT.

```
0.Visit1.ECG.CHESTXRAY.0.VIEW.VIEWRADIO.OTHERTEXT
```

The online control appears as follows:



Indicating that data contains multiple selection items

To import data to a multiple-selection control, a checkbox group, or a multiple-selection drop-down list:

1 In the import file, include all applicable selections in a single field. Separate each selection with a pipe (|).

For example, if you want the cigarettes and cigars checkboxes to be selected in a list containing cigarettes, cigars, and Not Done, and the values defined for those selections are "cigarettes," "cigars," and "ND," the import file should contain a field with the following value:

```
|cigarettes,cigars|
```

2 Define a map field as an InForm Item Path field that references the parent control for the checkboxes or the drop-down list.

3 In the Field Definition dialog, select the **Comma Separates Multiple Values** checkbox.

Checking for duplicate information within itemsets

The InForm Data Import utility can determine whether data already exists in the database by comparing the data itemset in which you are mapping to existing itemsets in the database.

To use this feature, in the Field Definition dialog, select **Match Itemset instance with this field** for each control.

For example, if you are importing lab information and the patient name, the data, and the type of test match data are already in the database, this might indicate that the data is a duplicate. The InForm Data Import utility recognizes this as duplicate data and does not add a second instance of the data in the database.

1 In the Submission Type window, select **Itemset**, then click **Continue**.

The Field Definition window appears.

2 Select the **Item Path** for the data object you want to match.

3 Select **Match Itemset instance with this field**.

4 Repeat these steps for any other data items you want to compare.

Importing information into an unscheduled visit

To import information into an unscheduled visit:

1 In the Submission Type window, select **Itemset**.

2 Click **Continue**.

The Field Definition window appears.

3 Select the **Item Path** for the date of visit with which you want to match a repeating visit.

4 Select **Match Repeating visit instance with this field**.

Repeat these steps for any other unscheduled visits to import.

Specifying a data type

You must specify a data type for each import field definition that you create as an InForm Item Path field.

To specify the data type of an import field:

- Click the appropriate button in the **Data Type** group.

  The InForm Data Import utility issues an error if any of the following exist:

  - An invalid integer field. An invalid integer field cannot be fully converted to an integer value. For example, 123$ is an invalid integer field.
  - A string identified as a floating number that does not meet the specification for the CRF control for allowed number of digits before and after the decimal point.
  - A string identified as a text control that is not within the defined size range for the CRF text control.

Mapping strings and child controls

Use the InForm Data Import utility to map field values in your import file to the database definitions of CRF controls, which have predefined values. Additionally, the mapping feature generates mappings in compound controls between individual child controls and their database ID paths.

Mapping strings

When the target of an input data field is a control for which an online user selects a predefined value, the value of the import field must be the same as the value of the selected control as defined in the database. These values are case-sensitive.

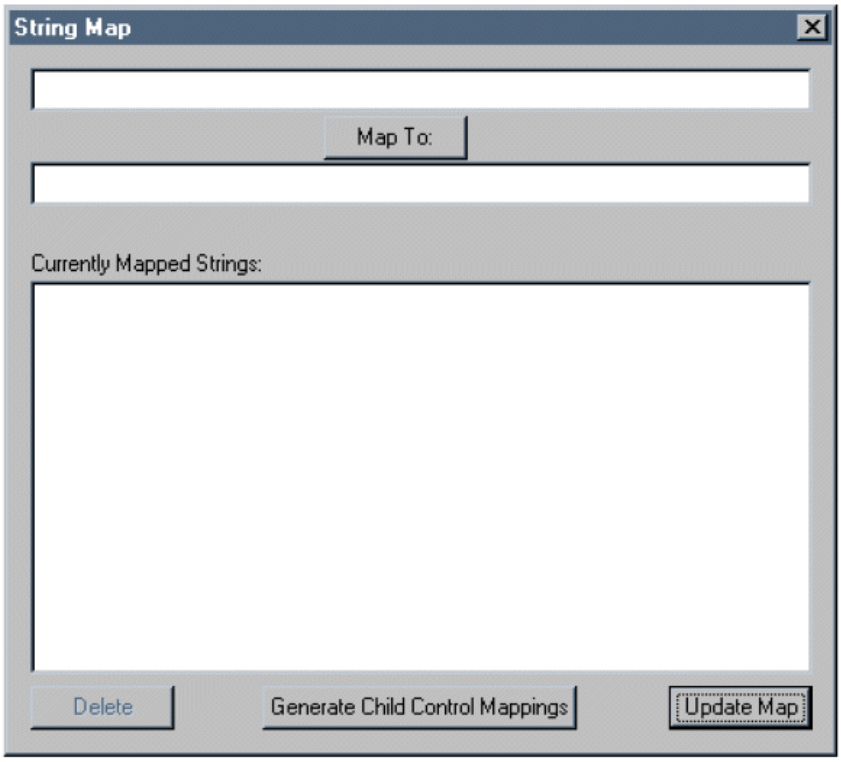
If your import file does not match the defined database values, you can convert your file to match them.

Alternatively, you can use the string mapping feature of the InForm Data Import utility to specify mappings between the values in your import file and the values defined for the target data fields in the InForm database.

To use this feature while creating the definition of a map file field:

1 In the Field Definition dialog box, click **Map Strings**.

The String Map dialog box appears.



2 In the top field, type a possible value of the control as it appears in the import file.

3 In the next field, type the value of the control as it is defined in the database. This definition is specified, generally with a VALUE attribute, in the XML file that is used to load form and data item definitions into the database.

4 Click **Map To**. The utility transfers the pair of values to the **Currently Mapped Strings** field. For example, if you typed *Yes* as a value that appears in your file and *Y* as the defined control value, the **Currently Mapped Strings** field shows the mapping as *Yes maps to Y*.

5 Repeat the mapping definition for each possible combination of values that the field can have in your import file and in the database definition of the control.

6 Click **Update Map**.

Mapping child controls

When the target control is nested within another control (for example, a field within a list of radio buttons), you must create separate map fields for the group control and for each child control within the group. Similarly, your import file must contain fields for the group control and for each possible child control selection.

To assign a specific value to the group control selection, the InForm Data Import utility maps child control names to their database ID paths.

To generate child control mappings for a group control map field:

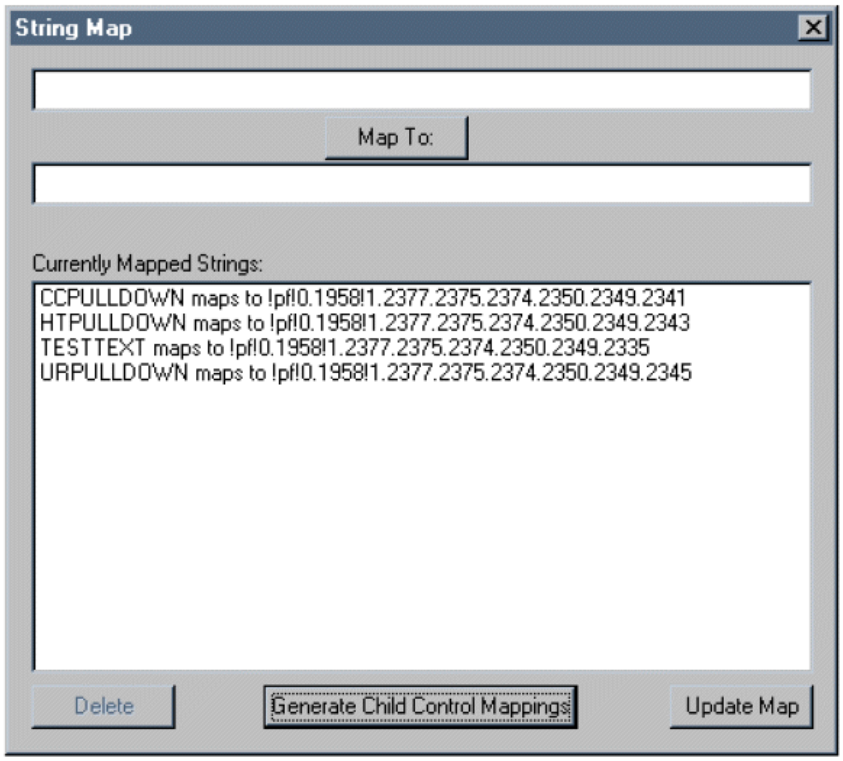
1 In the Field Definition dialog box, click **Map Strings**.

The String Map dialog box appears.

2 Click **Generate Child Control Mappings**.

The **Currently Mapped Strings** field shows the mappings between child control RefNames and their database IDs. In the import file field that corresponds to the map field that defines the group control, type the child control RefName for which you are providing data.

3 Click **Update Map**.



Navigating the map file

As you create the fields in a map file, the field definitions are strung together in a sequence in which you can move back and forth.

Use the control buttons at the bottom of the Field Definition dialog box to do the following:

- To view a field that occurs earlier in the map file, click **Back**.
- To view a field that occurs later in the map file, click **Next**.
- To create a new field definition, advance to the last field in the map file and click **Create Next**.

Note: You cannot navigate away from the field until you create or delete the new field definition by clicking **Finish** or **Delete**.

Inserting or deleting an import field

To insert or delete an import field definition in the map:

1 In the InForm Data Import dialog, in the **Field Map File** field, type the name of the file.

2 Click **Edit Map File**.

3 Click **Insert Field**.

> **Note:** The InForm Data Import utility inserts the new field definition immediately before the field that is displayed.

The InForm Data Import utility clears the data entry fields on the Field Definition dialog.

4 Type the definition for the new field.

5 Click **Next**, **Back**, or **Finish**, as appropriate.

The InForm Data Import utility inserts the new field definition immediately before the field that was displayed when you clicked **Insert**.

6 To save the map definition, click **Finish**.

> **Note:** You cannot navigate away from the field until you create or delete the new field definition by clicking **Finish** or **Delete Field**.

To delete an import field definition:

1 Click **Next** or **Back** to find the field definition to delete.

2 Click **Delete Field**.

3 To save the map definition, click **Finish**.

> **Note:** If you change the definitions in a map file and click **Cancel** (instead of **Finish**), the InForm Data Import utility saves an empty map file with the path and filename that you specified in the **Map** field.

Checking the map against the import file

Use the InForm Data Import map file editor to review the map field definitions against the actual import data that you will be processing.

To check the map file against the first line of the import file:

1 In the InForm Data Import dialog, in the **Map** field, type the name of the map file to check, or click **Browse** and locate the file.

2 In the **Data** field, type the name of the import file, or click **Browse**.

3 Click **Edit Map File**.

4 In the Submission Type window, click **Continue**.

5 In the Field Definition dialog, navigate through the field definitions and compare the

definition of each file with the data that appears in the **Sample Data** field.

Running the import using the data and map import file

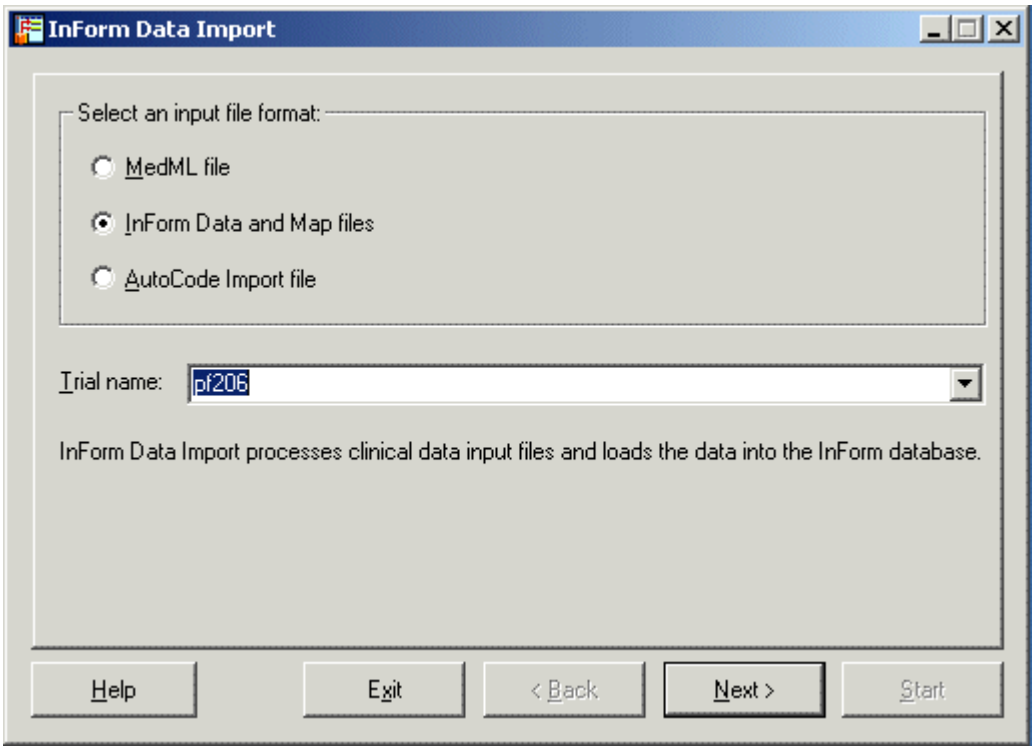
When your map definition is complete and the import file and map file are synchronized, you can import data.

Note: To import data into the database, the server must be running before you start the InForm Data Import. To see the effect of imported data on the patient status icons, you must stop and restart the server after importing.

To import InForm data and map files into the InForm database:

1 Double-click the **PFImport.exe** file located in the **\bin** directory of the InForm installation.

The InForm Data Import appears.



2 Select **InForm Data and Map files**.

3 In the **Trial Name** field, select or type the name of the trial into which you want to import the InForm data and map files.

4 Click **Next**.

5 In the **Data** field, type the full path name of the data file you want to import, or click **Browse**.

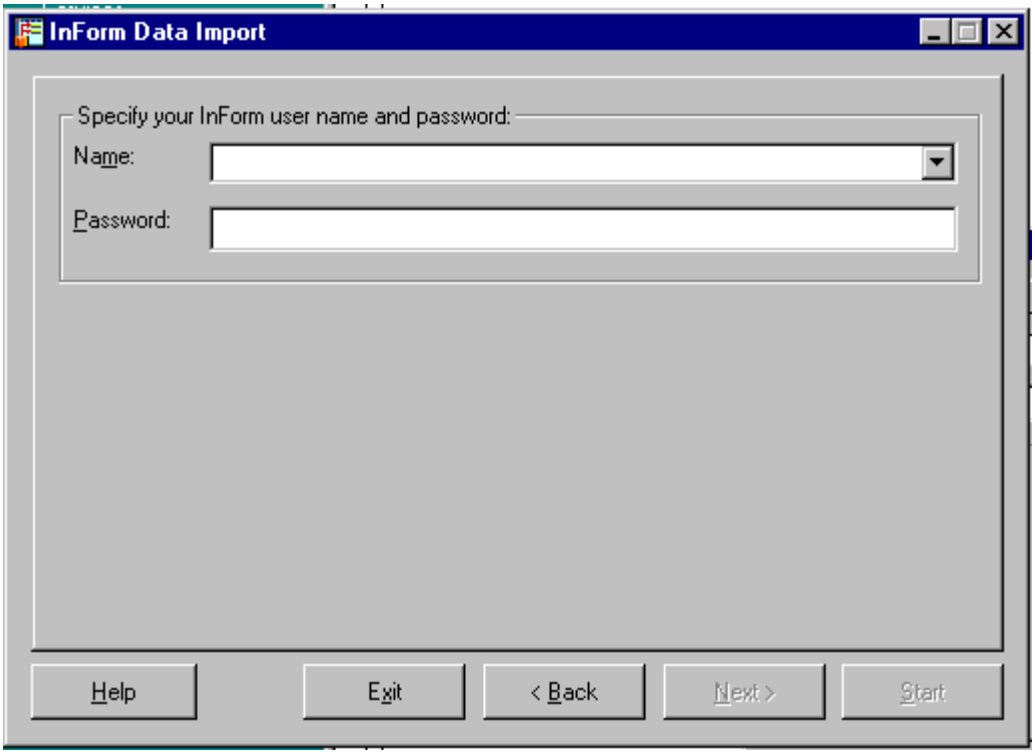
6 In the **Map** field, type the full path name of the map file you want to import, or click **Browse**.

7 To create or edit the map file, click **Edit Map File**. For more information, see *Specifying and*

editing map files (on page 17).

8 Select **Next**.

A dialog appears and requests your InForm name and password.

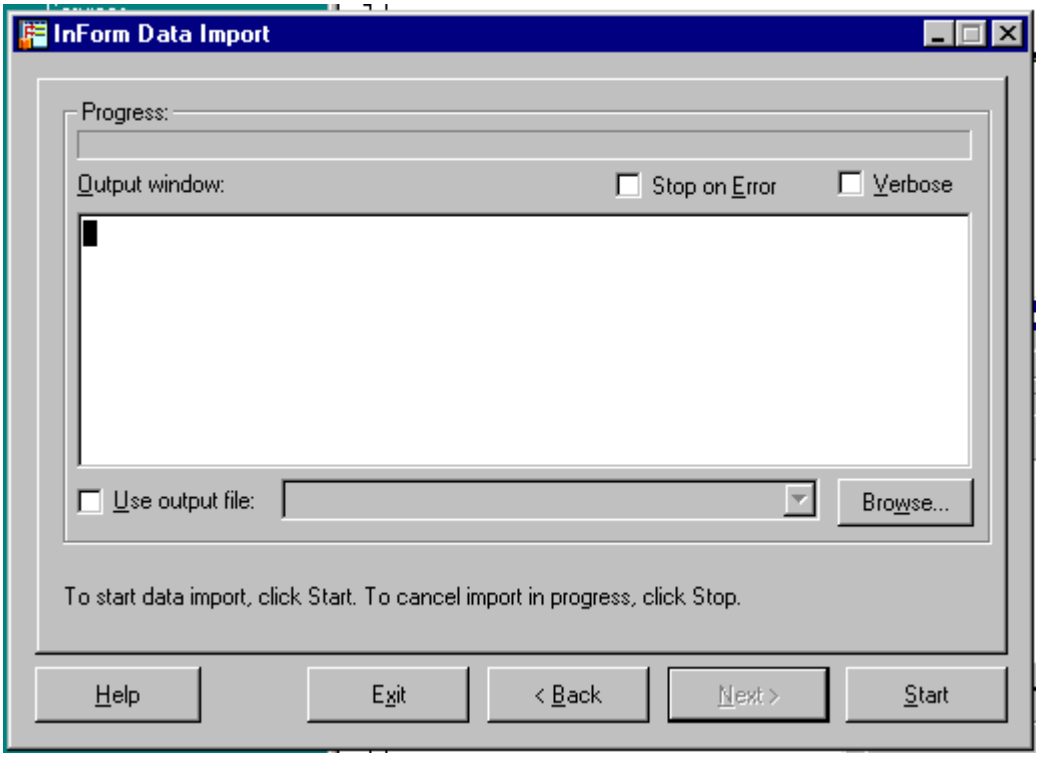


9 In the **Name** field, type the name of an InForm user who has the appropriate rights for the data you are importing:

To import data that matches this InForm system activity	User needs these rights
Add patient clinical data	Enter Data into a CRF
Update patient clinical data	Edit Data on a CRF

10 In the **Password** field, type the user password, then click **Next**.

The Summary window appears.



11

12 Optionally, select any of the following:

- **Stop on Error**—To instruct the InForm Data Import to stop if it encounters an error.
- **Verbose**—To instruct the InForm Data Import to generate detailed messages as it processes the file.
- **Use output file**—Specify the filename to save the output file as a text file.

13 Click **Start**.

The InForm Data Import processes the import file, writes messages to the message area and the output file, if specified, and adds or updates data in the database.

14 Close the InForm Data Import. To view the data that you imported, you must stop and restart the trial.

Saving the map file

To save the map definitions in the specified file:

1 In the **Field Map File** field, click **Finish**.

Note: If you change definitions in a map file and click **Cancel** (instead of **Finish**), an empty map file is saved with the path and filename you specified in the Map field.

Checking the error file

As the InForm Data Import utility processes the import file, it creates an error file if it is unable to import any of the import file rows.

To check the error file:

1 Open the output file with the filename that you specified, in the directory that you specified.

> **Note**: If you did not specify a directory and filename for the output file, the error file is saved in the same directory in which the InForm Data Import utility executable, PFImport.exe, is stored. After the import is complete, check for the presence of an ERR file and review the file for errors.

Importing an XML file

To import an XML file:

1 Create an XML file that contains the patient data to add or update. For more information, see *Creating an XML file for data import* (on page 33).

2 Run the import with the MedML file option. For more information, see *Running the import using the XML import file* (on page 49).

XML import files

The import file for the MedML file option is an XML file that contains elements that specify the type of processing to perform during the import and the destinations and values of the import data. The file can contain tags for the following types of import actions:

- Screening and enrolling a patient.
- Adding new patient data.
- Updating existing patient data.
- Transferring a patient from one site to another.

To create the import file, use any text editor that creates plain text files. For more information, see *Appendix A: Sample Data Import XML* (on page 129).

Creating an XML file for data import

To create an XML file for data imports:

1 Create a first line that contains the XML version number. The version string must be lower case:

```
<?xml version="1.0"?>
```

2 Add an opening and closing element that tells the InForm Data Import utility what type of processing to perform:

```
<CLINICALDATA>
</CLINICALDATA>
```

3 Between the opening and closing elements, add opening and closing elements for each activity for the InForm Data Import utility to perform. Use one set of activity elements for each patient for whom to import data. For example, to import screening data for a new patient, use the following elements:

```
<SCREEN>
</SCREEN>
```

4 In the opening element for the import activity, add the attributes required for that activity type, and any optional attributes. For example, the SCREEN element requires a SITEMNEMONIC or SITENAME attribute to specify the patient site by mnemonic or by name. If the patient site mnemonic is PF, you would insert the SITEMNEMONIC element as follows:

```
<SCREEN SITEMNEMONIC="PF">
```

Note: If you are using the InForm Data Import utility to transfer patient records between sites, go to the final step. A patient record transfer import file does not use the DATA element.

5 Between the opening and closing elements that specify the import activity, insert a DATA element for each form control:

```
<DATA/>
```

6 The DATA tag has the following required attributes:

- **TAG**—A database path that identifies the target data item control, and that is made up of RefNames in the following order:

  Section.Itemset.Item[.control[.control…]]

- **Section**—RefName of the section, as specified in the XML file that contains the section definition.

- **Itemset**—RefName of the itemset, as specified in the XML file that contains the itemset definition. If the target data item is a regular CRF item, not an itemset, type 0.

- **Item**—RefName of the item, as specified in the XML file that contains the item definition. Create a separate DATA element for each item in an itemset.

- **Control**—RefName of the control, as specified in the XML file that contains the control definition. To access an element of a group control, refer to each parent control in which the child element is nested. For example, to address one of two text controls within a group control, type the RefName of the group control followed by the RefName of the text control, and separate the names with periods, as follows:

  GroupControlRefName.TextControlRefName.

- One of the following:

  - **VALUE**—The value of the data to import into the control. Enclose the value in double quotes.

    Note: Because double quotes are used to delimit the value of an attribute, you can not include double quotes as part of the value text. If you need to include double quotes as part of the value text, use the XML entity reference ".

  - **CHILDSELECTED**—The RefName of the selected child control, if the child control is nested within a compound control. For example, use the CHILDSELECTED attribute to indicate which radio control to select if the radio control includes two drop-down lists.

  - **MONTH**, **DAY**, **YEAR**, **HOUR**, **MINUTE**, **SECOND**—The value of each applicable part of a datetime control.

  - **UNIT**—Unit type of the selected control, when a unit definition is part of the target control.

  - **COMMENT**—The text of an item-level comment.

  - **REASONINCOMPLETE**—The reason the item is incomplete. When you specify this attribute, do not include a VALUE or any datetime control attributes in the

DATA tag.

- **NOMULTIVALUE**—Indicates that a data value containing a comma (,) is a single value. Without this attribute, only the data preceding a comma is stored as a value. Data that occurs after a comma is assumed to be a separate value in a multi-value DATA tag.

For existing patient data only:

- **CLEARVALUE**—Clears the existing value for the specified control. Available options are TRUE or FALSE.

7 Save the file.

Examples

The following example shows a DATA element that is used to import the initials of patient AAA in the PF site to the patientinitials field in the screening form:

```
<SCREEN SITEMNEMONIC="PF">
   <DATA TAG="screen.0.patientinitials.patientinitials"
   VALUE="AAA"/>
</SCREEN>
```

The following example shows the use of the CHILDSELECTED attribute of the DATA element to indicate that, in the RACE radio control, the RACEPULLDOWN radio button is being selected. The second DATA element gives the selected value within the drop-down list.

```
<PATIENTDATA PATIENTINITIALS="AAA" SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1" FORMREFNAME="DEM" COMMENT>
   <DATA TAG="DEM.0.RACE.RACEGROUP"
   CHILDSELECTED="RACEPULLDOWN"/>
   <DATA TAG="DEM.0.RACE.RACEGROUP.RACEPULLDOWN"
   VALUE="Asian"/>
</PATIENTDATA>
```

The following example shows a DATA element used to specify a date to be imported into an enrollment form:

```
<ENROLL PATIENTINITIALS="AAA" SITEMNEMONIC="PF"
PATIENTNUMBER="BK1" ENROLL="TRUE">
   <DATA TAG="consent.0.consentdate.date"
   MONTH="1" DAY="6" YEAR="1999"/>
</ENROLL>
```

The following example shows the use of the UNIT attribute to specify that the unit in which height is being measured is inches.

```
<PATIENTDATA PATIENTINITIALS="AAA" SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1" FORMREFNAME="DEM">
   <DATA TAG="DEM.0.HEIGHT.HEIGHTTEXT"
   VALUE="67" UNIT="Inches"/>
</PATIENTDATA>
```

The following example shows the use of the NOMULTIVALUE attribute to specify that a data value containing a comma (,) is a single value.

```
<PATIENTDATA PATIENTINITIALS="AAA" SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1" FORMREFNAME="DEM">
   <DATA TAG="DEM.0.HEIGHT.ID"
   VALUE="67,11" NOMULTIVALUE=""/>
</PATIENTDATA>
```

Screening

To import screening data for a patient at a site, use the SCREEN element with the following required attribute:

Attribute	Definition
SITEMNEMONIC	Specifies the mnemonic of the site at which the patient is being screened. Either SITEMNEMONIC or SITENAME is required.

Example

The following sample file illustrates the elements used to screen patient AAA at site PF.

```
<?xml version="1.0"?>
<CLINICALDATA>
<!-- Screen Patient -->
<SCREEN SITEMNEMONIC="PF">
   <DATA TAG="screen.0.patientinitials.patientinitials"
      VALUE="AAA"/>
   <DATA TAG="screen.0.eligible.eligible" VALUE="yes"/>
   <DATA TAG="screen.0.datescreened.date" MONTH="1" DAY="6"
      YEAR="1999"/>
   <DATA TAG="screen.0.dob.dob" MONTH="11" DAY="11"
      YEAR="1959"/>
</SCREEN>
</CLINICALDATA>
```

Enrolling

To import enrollment data for a patient at a site, use the ENROLL element with the following required attributes:

Attribute	Definition
PATIENTINITIALS	Specifies the initials of the patient that is being enrolled.
SITEMNEMONIC	Specifies the mnemonic of the site at which the patient is being screened. Either SITEMNEMONIC or SITENAME is required.
DUPLICATEORDER	Number that specifies the order in which patients who have the same patient initials and were enrolled in the same site were screened.
PATIENTNUMBER	Specifies the patient number of the patient being enrolled.
ENROLL	TRUE or FALSE, indicating whether to enroll the patient.

Example 1

The following example illustrates the elements used to enroll patient XYZ at site PF. This example assumes that patient XYZ has previously been screened.

```
<?xml version="1.0"?>
```

```
<CLINICALDATA>

<!-- Screen Patient -->
<SCREEN SITEMNEMONIC="PF">
<DATA TAG="screen.0.patientinitials.patientinitials" VALUE="XYZ"/>
<DATA TAG="screen.0.eligible.eligible" VALUE="yes"/>
<DATA TAG="screen.0.datescreened.date" MONTH="1" DAY="6" YEAR="1999"/>
<DATA TAG="screen.0.dob.dob" MONTH="11" DAY="11" YEAR="1959"/>
</SCREEN>

<!-- Enroll Patient -->
<ENROLL PATIENTINITIALS="XYZ" SITEMNEMONIC="PF" PATIENTNUMBER="BK-XYZ"
ENROLL=”TRUE”>
<DATA TAG="consent.0.consentdate.date" MONTH="1" DAY="6" YEAR="1999"/>
<DATA TAG="consent.0.patientnumber.patientnumber" VALUE="28"/>
<DATA TAG="inclusion.0.age_inc.yesno" VALUE="1"/>
<DATA TAG="inclusion.0.hyper_inc.yesno" VALUE="1"/>
<DATA TAG="inclusion.0.understand_inc.yesno" VALUE="1"/>
<DATA TAG="inclusion.0.agree_inc.yesno" VALUE="1"/>
<DATA TAG="exclusion.0.secondary_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.malignant_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.allergyhistory_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.myocardial_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.monitor_ex.yesno" VALUE="0"/>
</ENROLL>
</CLINICALDATA>
```

Example 2

The following example illustrates the use of the DUPLICATEORDER attribute to specify the order in which patients with duplicate patient initials should be enrolled. Patients John R. Doe and Jane R. Doe (JRD) are screened at site PF; first John, then Jane.

```
<!-- This is John R. Doe-->
<ENROLL PATIENTINITIALS=”JRD” SITEMNEMONIC=”PF” DUPLICATEORDER=”1”
PATIENTNUMBER=”BK-JRD” ENROLL=”TRUE”>
</ENROLL>
<!-- This is Jane R. Doe-->
<ENROLL PATIENTINITIALS=”JRD” SITEMNEMONIC=”PF” DUPLICATEORDER=”2”
PATIENTNUMBER=”BK-JRD” ENROLL=”TRUE”>
</ENROLL>
```

Adding new patient clinical data

To add new clinical data to an existing patient at a site, use the PATIENTDATA element with the following attributes:

Attribute	Definition
PATIENTINITIALS	Specifies the initials of the patient that is being enrolled. Either PATIENTINITIALS or PATIENTNUMBER is required.
PATIENTNUMBER	Specifies the patient number of the patient being enrolled. Either PATIENTINITIALS or PATIENTNUMBER is required.
SITEMNEMONIC	Specifies the mnemonic of the site at which the patient is being screened. Either SITEMNEMONIC or SITENAME is required. Either SITEMNEMONIC or SITENAME is required.

Attribute	Definition
SITENAME	Specifies the name of the site at which the patient is enrolled. Either SITEMNEMONIC or SITENAME is required.
FORMSETREFNAME	Specifies the RefName of the visit to which you are importing data.
FORMSETINDEX	Indicates to which visit instance to add the data. Either FORMSETINDEX or FORMSETINDEXORDER is required.
FORMSETINDEXORDER	Indicates to which visit instance to add the data. Either FORMSETINDEX or FORMSETINDEXORDER is required.
FORMREFNAME	Specifies the RefName of the CRF to which you are importing data.
REASONINCOMPLETE	This attribute may apply to either the form or item level. The value is one of the values in the radio group control. This attribute will be ignored if the form or item is complete. To add an incompletion reason at the form level, do not include a DATA tag in the PATIENTDATA group.
FORMINDEX	(Optional) If not present, then a new repeating form instance will be created (if FORMREFNAME is a repeating form). If present, the value indicates to which form instance the new data will be added.
ASSOCIATION	Creates or deletes an association instance.

Adding data to an itemset

To add data to an itemset on a form, use the following additional attributes:

Attribute	Definition
SECTIONNAME	Specifies the RefName of the section in which the itemset occurs. Either SECTIONNAME or SECTIONID is required.
SECTIONID	Specifies the ID of the section in which the itemset occurs. Either SECTIONNAME or SECTIONID is required.
ITEMSETNAME	Specifies the RefName of the itemset definition.

Attribute	Definition
ITEMSETINDEX	Number of the itemset line to update. If the ITEMSETINDEX section is blank or has a value of 0, the InForm Data Import utility adds a new itemset line. If ITEMSETINDEX is a number other than 0, the InForm Data Import utility updates the specified line. Note that existing itemset data is not changed, but missing data is filled in. For more information, see *Example 2: Creating an itemset row* (on page 41).

Adding data to an unscheduled visit

To add data to an unscheduled visit, use the following additional attributes:

Attribute	Definition
NEWUNSCHEDVISIT	Indicates whether the data is being added to a new unscheduled visit. Values are TRUE or FALSE (the default). Use this attribute on the Visit Date form, a predefined form with the FORMREFNAME of DOV. For more information, see *Example 3: Adding data to an unscheduled visit* (on page 41).
FORMSETINDEX	Number that specifies the unscheduled visit to which to add the data. The number corresponds to the order in which the unscheduled visit was added to the study.

Importing autocoded data

When you use the InForm Data Import utility to import data that was exported using the autocode feature of the InForm application and coded by an external coding application, the InForm Data Import utility sets the following attribute to TRUE:

Attribute	Definition
AUTOCODEIMPORT	Indicates whether data has been exported using the autocode feature of the InForm application and is being imported after being coded by an external coding application. A value of TRUE indicates that the InForm application must determine whether signatures on the form that is receiving the data must be invalidated. This determination is based on: • Visibility of the coded data. • Setting of the INVALIDATIONLEVEL attribute of the SIGNCRF MedML tag that associates a form with a signature group. For more information, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

Adding a comment

You can add a comment to an item, an item within an itemset, or a form by using the COMMENT attribute:

- To add a comment to an item, include the COMMENT attribute along with the text of the comment in the DATA element.
- To add a comment to a form, include the COMMENT attribute along with the text of the comment in a PATIENTDATA element that does not include SECTIONNAME and ITEMSETNAME attributes.

Example 1: Adding data to a form

The following XML fragment shows elements used to add data to the DEM form for patient AAA at site PF.

```
<?xml version="1.0"?>
<CLINICALDATA>
<PATIENTDATA
   PATIENTINITIALS="VOL" SITEMNEMONIC="PF"
   FORMSETREFNAME="Visit1"
   FORMREFNAME="DEM" COMMENT=This form was edited by Joe>
   <DATA TAG="DEM.0.GENDER.GENDERRADIO" VALUE="1"/>
   <DATA TAG="DEM.0.DEMDOB.dob" MONTH="2" DAY="14"
      YEAR="1961" COMMENT=As reported by patient/>
   <DATA TAG="DEM.0.RACE.RACEGROUP"
      CHILDSELECTED="RACETEXT"/>
   <DATA TAG="DEM.0.RACE.RACEGROUP.RACETEXT"
      VALUE="African American"/>
   <DATA TAG="DEM.0.HEIGHT.HEIGHTTEXT" VALUE="67"
      UNIT="Inches"/>
   <DATA TAG="DEM.0.WEIGHT.WEIGHTTEXT" VALUE="150"
```

```
      UNIT="Pound"/>
   <DATA TAG="DEM.0.FRAME.FRAMEPULLDOWN" VALUE="1"/>
   <DATA TAG="SH.0.SMOKE.SMOKERADIO" VALUE="Y"/>
   <DATA TAG="SH.0.EVERSMOKED.SMOKERADIO" VALUE="N"/>
   <DATA TAG="SH.0.WHATSMOKED.SMOKECHECKBOX"
      VALUE="cigarette,pipe"/>
   <DATA TAG="SH.0.HOWMUCHSMOKED.SMOKERADIO2"
      CHILDSELECTED="NUMTEXT"/>
   <DATA TAG="SH.0.HOWMUCHSMOKED.SMOKERADIO2.NUMTEXT"
      VALUE="10"/>
   <DATA TAG="SH.0.YRSSMOKED.SMOKERADIO2"
      VALUE="NDElement"/>
</PATIENTDATA>
</CLINICALDATA>
```

Example 2: Creating an itemset row

The following XML fragment shows the elements used to create a new itemset row on the Hypertension History form.

```
<PATIENTDATA
   PATIENTINITIALS="AAA" SITEMNEMONIC="PF"
   FORMSETREFNAME="Visit1" FORMREFNAME="HH"
   SECTIONNAME="PT" ITEMSETNAME="PT">
   <DATA TAG="PT.PT.THERAPYTEXT.THERAPYTEXT"
      VALUE="aspirin"/>
   <DATA TAG="PT.PT.DOSAGETEXT.DOSAGETEXT"
      VALUE="1 tablet"/>
</PATIENTDATA>
```

The following XML fragment shows the elements used to finish entering items for an existing itemset on the Hypertension History form.

```
<PATIENTDATA
   PATIENTINITIALS="AAA" SITEMNEMONIC="PF"
   FORMSETREFNAME="Visit1" FORMREFNAME="HH"
   SECTIONNAME="PT" ITEMSETNAME="PT" ITEMSETINDEX="1">
   <DATA TAG="PT.PT.DOSEDATE.DOSEDATE" MONTH="12" DAY="23"
      YEAR="1998"/>
   <DATA TAG="PT.PT.DISCPULLDOWN.DISCPULLDOWN"
      VALUE="Not Effective"/>
</PATIENTDATA>
```

Example 3: Adding data to an unscheduled visit

The following XML fragment shows the elements used to add data to the DOV form and Vital Signs form in the first and second unscheduled visits containing those forms.

```
<?xml version="1.0"?>
<CLINICALDATA>
<!-- DOV form -->
<PATIENTDATA
    PATIENTINITIALS="ABC" SITEMNEMONIC="PF"
    FORMSETREFNAME="UnschVisit"FORMREFNAME="DOV"
    NEWUNSCHEDVISIT="TRUE">
    <DATA TAG="DOV.0.DOV.DOV" MONTH="2" DAY="1"
        YEAR="1999"/>
</PATIENTDATA>
<!-- VitalSigns form -->
<PATIENTDATA
    PATIENTINITIALS="ABC" SITEMNEMONIC="PF" FORMSETINDEX="1"
    FORMSETREFNAME="UnschVisit" FORMREFNAME="VS">
    <DATA TAG="VS.0.DATEASSESS.COMMONDATE" MONTH="3" DAY="1"
        YEAR="1999"/>
    <DATA TAG="VS.0.WEIGHT.PFWT_TC" VALUE="150"
        UNIT="Pound"/>
    <DATA TAG="VS.0.TEMPTEXT.TEMPTEXT" VALUE="98.7"
        UNIT="Fahrenheit"/>
    <DATA TAG="VS.0.BPREADING.BPREADINGGROUP.SYSTEXT"
        VALUE="130"/>
    <DATA TAG="VS.0.BPREADING.BPREADINGGROUP.DIASTEXT"
```

```
        VALUE="85"/>
</PATIENTDATA>
<!-- DOV form -->
<PATIENTDATA
    PATIENTINITIALS="ABC" SITEMNEMONIC="PF"
    FORMSETREFNAME="UnschVisit" FORMREFNAME="DOV"
    NEWUNSCHEDVISIT="TRUE">
    <DATA TAG="DOV.0.DOV.DOV" MONTH="2" DAY="2"
        YEAR="1999"/>
</PATIENTDATA>
<!-- VitalSigns form -->
<PATIENTDATA
    PATIENTINITIALS="ABC" SITEMNEMONIC="PF" FORMSETINDEX="2"
    FORMSETREFNAME="UnschVisit" FORMREFNAME="VS">
    <DATA TAG="VS.0.DATEASSESS.COMMONDATE" MONTH="3" DAY="2"
        YEAR="1999"/>
    <DATA TAG="VS.0.WEIGHT.PFWT_TC" VALUE="150"
        UNIT="Pound"/>
    <DATA TAG="VS.0.TEMPTEXT.TEMPTEXT" VALUE="98.7"
        UNIT="Fahrenheit"/>
    <DATA TAG="VS.0.BPREADING.BPREADINGGROUP.SYSTEXT"
        VALUE="130"/>
    <DATA TAG="VS.0.BPREADING.BPREADINGGROUP.DIASTEXT"
        VALUE="85"/>
</PATIENTDATA>
```

Example 4: Specifying REASONINCOMPLETE

The following XML fragment shows an addition to the Pulse Rhythm item on the Vital Signs (VS) form.

```
<PATIENTDATA
   PATIENTINITIALS="A3" SITEMNEMONIC="PF"
   FORMSETREFNAME="DV1" FORMREFNAME="VS"
   REASONOTHER="test reason">
   <DATA TAG="VS.0.PULSERHYTHM.PULSERHYTHMRADIO"
      COMMENT=irregular REASONINCOMPLETE="NAElement"/>
</PATIENTDATA>
```

Example 5: Creating a new association instance

The following XML fragment shows the creation of a new association instance.

```
<PATIENTDATA
   PATIENTINITIALS="pjb" SITEMNEMONIC="PF"
   FORMSETREFNAME="Visit6" FORMREFNAME="VS"
   FORMINDEX=”2”
   REASONPULLDOWN=”3”
   <ASSOCIATION FORMSETREFNAME=”Visit1” FORMREFNAME=”DEM”
      FORMINDEX=”4”/>
</PATIENTDATA>
```

Updating existing patient clinical data

To modify existing clinical data for a patient at a site, use the EDITPATIENTDATA element with the following attributes:

Attribute	Definition
PATIENTINITIALS	Specifies the initials of the patient that is being enrolled. Either PATIENTINITIALS or PATIENTNUMBER is required.

Attribute	Definition
PATIENTNUMBER	Specifies the patient number of the patient being enrolled. Either PATIENTINITIALS or PATIENTNUMBER is required.
SITEMNEMONIC	Specifies the mnemonic of the site at which the patient is being screened. Either SITEMNEMONIC or SITENAME is required. Either SITEMNEMONIC or SITENAME is required.
SITENAME	Specifies the name of the site at which the patient is enrolled. Either SITEMNEMONIC or SITENAME is required.
FORMSETREFNAME	Specifies the RefName of the visit to which you are importing data.
FORMREFNAME	Specifies the RefName of the CRF to which you are importing data.
REASONPULLDOWN	Specifies the value of a predefined reason for change, as listed in the Reason for Change drop-down list on the Data Value(s) form.
REASONOTHER	Specifies the text of a reason for change, as entered in the Other Reason for Change field on the Data Value(s) form.
REASONINCOMPLETE	This attribute may apply to either the form or item level. The value is one of the values in the radio group control. This attribute will be ignored if the form or item is complete.
CLEARCRF	When true, indicates to ignore all following <Data Tags>.
FORMINDEX	Required for repeating forms. Indicates that a repeating form be updated. Value corresponds to the form instance to update.
ASSOCIATION	Creates or deletes an association instance.
ACTION	ADD or REMOVE; adds or removes an association.

Editing data in an itemset

To edit data in an itemset on a form, use the following additional attributes:

Attribute	Definition
SECTIONNAME	Specifies the RefName of the section in which the itemset occurs.
ITEMSETNAME	Specifies the RefName of the itemset definition.

Attribute	Definition
ITEMSETINDEX	Number of the itemset line to update. If the ITEMSETINDEX section is blank or has a value of 0, the InForm Data Import utility adds a new itemset line. If ITEMSETINDEX is a number other than 0, the InForm Data Import utility updates the specified line. Note that existing itemset data is not changed, but missing data is filled in. For more information, see *Example 2: Creating an itemset row* (on page 41).

Example: Modifying an itemset row

The following XML file fragment shows a modification to the fifth row of itemset data on the Adverse Event (AE) form.

```
<EDITPATIENTDATA
   PATIENTINITIALS="EDT" SITEMNEMONIC="PF"
   FORMSETREFNAME="CommonCRF" FORMREFNAME="AE"
   SECTIONNAME="AE" ITEMSETNAME="AE"
   ITEMSETINDEX="5" REASONOTHER="additional description">
   <DATA TAG="AE.AE.AEDESC.AEDESCTEXT"
      VALUE="Temp spike"/>
</EDITPATIENTDATA>
```

Updating autocoded data

When you use the InForm Data Import utility to update data that was exported using the autocode feature of the InForm application and coded by an external coding application, the InForm Data Import utility sets the following attribute to TRUE:

Attribute	Definition
AUTOCODEIMPORT	Indicates whether data has been exported using the autocode feature of the InForm application and is being imported after being coded by an external coding application. A value of TRUE indicates that the InForm application must determine whether signatures on the form that is receiving the data must be invalidated. This determination is based on: • Visibility of the coded data. • Setting of the INVALIDATIONLEVEL attribute of the SIGNCRF MedML tag that associates a form with a signature group. For more information, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

Editing a comment

You can edit a comment on an item, an item within an itemset, or a form by using the COMMENT attribute:

- To edit a comment on an item, include the COMMENT attribute along with the text of the comment in the DATA element.
- To edit a comment on an itemset, include the COMMENT attribute along with the text of the comment in an EDITPATIENTDATA element that includes the SECTIONNAME and ITEMSETNAME attributes, which signal that the EDITPATIENTDATA element is updating an itemset.
- To edit a comment on a form, include the COMMENT attribute along with the text of the comment in an EDITPATIENTDATA element that does not include SECTIONNAME and ITEMSETNAME attributes.

Example 1: Changing a data value

The following sample file illustrates a change to the value of the Height item on the DEM form. The Reason for Change is a string other than the predefined Reason for Change strings on the Data Value(s) form. Note that you must enter the UNIT value even if you are not modifying it, or it will be removed.

```
<?xml version="1.0"?>
<CLINICALDATA>
<!-- Demographics form -->
<EDITPATIENTDATA
   PATIENTINITIALS="XYZ"
   SITEMNEMONIC="PF"
   FORMSETREFNAME="Visit1"
   FORMREFNAME="DEM"
   REASONOTHER="received new data">
   <DATA TAG="DEM.0.HEIGHT.PFHT_TC" VALUE="75"
      UNIT="Inches" />
</EDITPATIENTDATA>
</CLINICALDATA>
```

Example 2: Clearing a data value

The following XML file fragment shows the use of the CLEARVALUE attribute of the DATA element. When you change the selection of a compound radio control, you must clear the value of the original child control.

```
<EDITPATIENTDATA
   PATIENTINITIALS="PE1" SITEMNEMONIC="PF"
   FORMSETREFNAME="Visit1" FORMREFNAME="DEM"
   REASONPULLDOWN="New Information">
   <DATA TAG="RACEGROUP.RACETEXT" CLEARVALUE="TRUE"/>
</EDITPATIENTDATA>
```

Example 3: Specifying REASONINCOMPLETE

The following XML file fragment shows a modification to the Pulse Rhythm item on the Vital Signs (VS) form.

```
<EDITPATIENTDATA
   PATIENTINITIALS="A3" SITEMNEMONIC="PF"
   FORMSETREFNAME="DV1" FORMREFNAME="VS"
   REASONOTHER="test reason">
   <DATA TAG="VS.0.PULSERYTHM.PULSERHYTHMRADIO"
      COMMENT=me too REASONINCOMPLETE="NAElement"/>
</EDITPATIENTDATA>
```

Example 4: Clearing a CRF

The following XML file fragment shows the clearing of a CFR.

```
<EDITPATIENTDATA
   PATIENTINITIALS="DD" SITEMNEMONIC="PF"
   FORMSETREFNAME="VISIT7" FORMREFNAME="PREIM"
   REASONOTHER="clear CFR test">
   CLEARCRF="TRUE"
   <DATA TAG="DEM.0.DTV.DTV_DC" MONTH="4" DAY="22"
      YEAR="2000"
</EDITPATIENTDATA>
```

Example 5: Deleting an association

The following XML file fragment shows the command to delete an association.

```
<EDITPATIENTDATA
   PATIENTINITIALS="DD" SITEMNEMONIC="PF"
   FORMSETREFNAME="VISIT7" FORMREFNAME="PREIM"
   FORMINDEX=”2”
   REASONPULLDOWN="Transcription Error">
   <ASSOCIATION FORMSETREFNAME=”Visit6” FORMREFNAME=”PE” FORMINDEX=”3”
   ACTION=”REMOVE”
</EDITPATIENTDATA>
```

Example 6: Deleting a repeating form

The following XML file fragment shows a command to delete a repeating form.

```
<?xml version="1.0"?>
<CLINICALDATA xmlns="PhaseForward/ImportXML/Inform4">
<EDITPATIENTDATA
     PATIENTINITIALS="XYZ"
     SITEMNEMONIC="PF"
     FORMSETREFNAME="vstAECM"
     FORMREFNAME=" LAE1 "
     FORMINDEX="1"
  <!-- Form Status action:
       DELETE to delete the form
       UNDELETE to restore a deleted form
   -->
     FORMSTATUS="DELETE"
     REASONOTHER="Deleting crf" />
</CLINICALDATA>
```

Example 7: Restoring a repeating form

The following XML file fragment shows a command to restore (undelete) a repeating form.

```
<?xml version="1.0"?>
<CLINICALDATA xmlns="PhaseForward/ImportXML/Inform4">
<EDITPATIENTDATA
   PATIENTINITIALS="XYZ"
   SITEMNEMONIC="PF"
   FORMSETREFNAME=" vstAECM "
   FORMREFNAME=" LAE1"
   FORMINDEX="1"
   <!-- Form Status action:
        DELETE to delete the form
        UNDELETE to restore a deleted form
    -->
   FORMSTATUS="UNDELETE"
   REASONOTHER="Undeleting CRF" />
</CLINICALDATA>
```

Transferring a patient record

You can use the InForm user interface to transfer patient information one patient at a time. You can also transfer patients in bulk using the InForm Data Import utility.

You can transfer patients who:

- Change permanent address before completing the trial.
- Have multiple residences throughout the course of the trial.
- Were initially assigned to the wrong sites or to an investigator who is no longer with the trial.

Whether you transfer patients individually, or transfer them in bulk, keep in mind that:

- The InForm application only allows you to transfer patients from one site to another if the trial version at the destination site is the same or greater than the trial version at the current patient site.

  To see the trial version for a site, click **Admin > Sites**.
- You can transfer only patients who are fully enrolled; you cannot transfer a patient who is screened but not enrolled or a patient who has failed enrollment.
- You cannot use the patient record transfer feature if you are also using the site filtering feature of the InForm Unplugged application.

To transfer a patient record from one site to another, use the following tags:

- PATIENTSITECHANGE
- NEWSITE
- CURRENTSITE

PATIENTSITECHANGE

The PATIENTSITECHANGE element identifies each patient to transfer. Use one pair of opening and closing PATIENTSITECHANGE elements for each patient that you want to transfer. The PATIENTSITECHANGE element surrounds all information about a single patient, the current site, and the destination site.

The PATIENTSITECHANGE element has one required attribute:

Attribute	Definition
REASON	A textual description of the reason for the patient transfer.

Within the PATIENTSITECHANGE element, include these elements:

- NEWSITE
- CURRENTSITE

NEWSITE

The NEWSITE element provides information about the destination site for the patient. Use the NEWSITE element within the opening and closing PATIENTSITECHANGE elements.

The NEWSITE element requires one attribute to identify the destination site. You can use any one of these attributes:

Attribute	Definition
SITEMNEMONIC	Mnemonic for the destination site.
SITENAME	Site name for the destination site.

The NEWSITE tag has one optional attribute:

Attribute	Definition
PATIENTNUMBER	The patient number for the patient at the destination site. Use this attribute only if you need to change the patient number from the one that exists at the current site because a duplicate exists at the destination site.

CURRENTSITE

The CURRENTSITE element provides information about the current, or originating, site for the patient. Use the CURRENTSITE element within the opening and closing PATIENTSITECHANGE elements.

The CURRENTSITE element requires two attributes: one to identify the current site, and one to identify the patient who is to be transferred.

Use any one of the following attributes to identify the current site:

Attribute	Definition
SITEMNEMONIC	Mnemonic for the destination site.
SITENAME	Site name for the destination site.

Use any one of the following attributes to identify the patient to be transferred:

Attribute	Definition
PATIENTINITIALS	The initials of the patient being transferred. If you use this attribute, and you know that more than one patient at the current site has the specified initials, use the optional DUPLICATEORDER attribute to indicate which patient to transfer.

Attribute	Definition
PATIENTNUMBER	The patient number of the patient being transferred, as it exists on the current site.

Use the following attribute with the CURRENTSITE element if more than one patient at the site has the initials specified in the PATIENTINITIALS element:

Attribute	Definition
DUPLICATEORDER	Specifies which patient should be transferred, if more than one patient at the site has the same initials. When multiple patients have the same initials, the InForm application checks the screening numbers of those patients and transfers the patient whose screening number is in the order specified by the DUPLICATEORDER tag. Use this attribute only if you have used PATIENTINITIALS to identify the patient.

Example

This example shows the elements that are used in a MedML file to transfer several patients. Note that each of the pairs of PATIENTSITECHANGE elements defines current and destination site information for one patient.

```
<?xml version="1.0"?>
<CLINICALDATA>
    <PATIENTSITECHANGE REASON="new patient address">
        <NEWSITE SITEMNEMONIC="MCLEAN"/>
        <CURRENTSITE SITEMNEMONIC="PF" PATIENTNUMBER="1003"/>
    </PATIENTSITECHANGE>
<!----For patient 1001 the file specifies a new patient number because a patient already exists at the McLean Hospital site with the same patient number.>

    <PATIENTSITECHANGE REASON="new patient address">
        <NEWSITE SITENAME="McLean Hospital" PATIENTNUMBER="1002"/>
        <CURRENTSITE SITEMNEMONIC="PF" PATIENTNUMBER="1001"/>
    </PATIENTSITECHANGE>
<!----The DUPLICATEORDER tag indicates that patient DDD is the second patient with those initials to be screened at the Phase Forward site.>

    <PATIENTSITECHANGE REASON="new patient address">
        <NEWSITE SITEMNEMONIC="MCLEAN"/>
        <CURRENTSITE SITENAME="Phase Forward" PATIENTINITIALS="DDD"
            DUPLICATEORDER="2"/>
    </PATIENTSITECHANGE>
</CLINICALDATA>
```

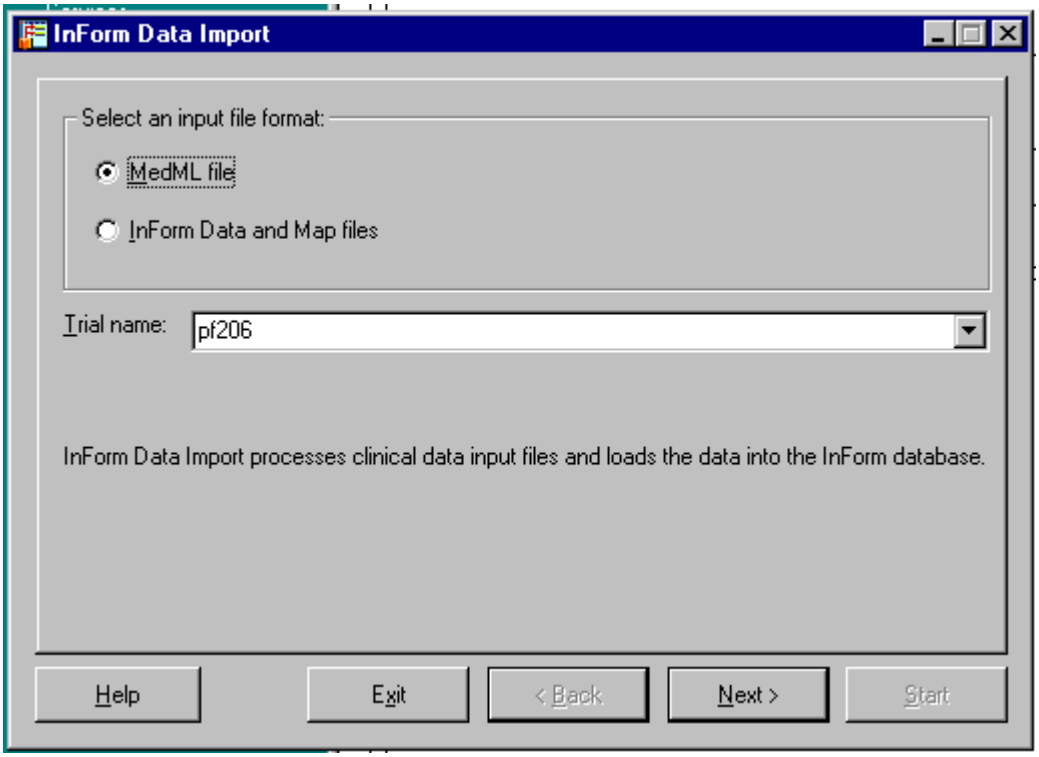
Running the import using the XML import file

To import an XML file into the database and submit it through the InForm application server:

Note: To import data into the database, the server must be running before you start the InForm Data Import utility.

1 Click **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm Data Import**.

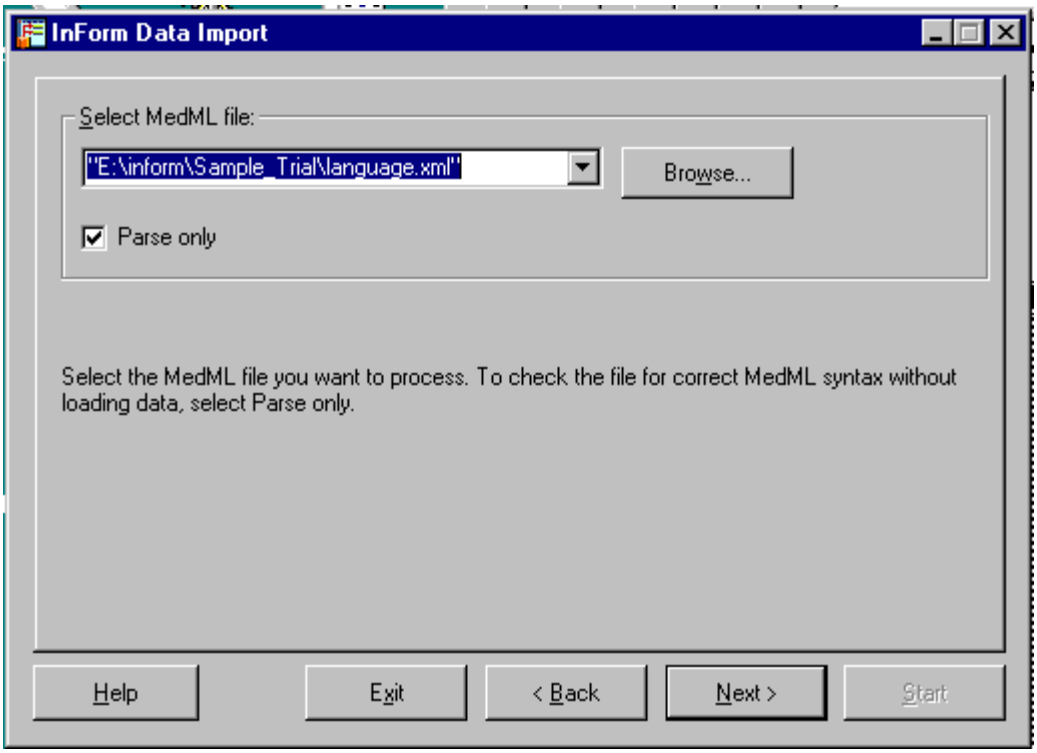
The InForm Data Import main window appears.



2 Select the **MedML file** radio button.

3 In the **Trial Name** field, type or select the name of the trial into which to import the MedML file. The last 10 trials you accessed are stored in the drop-down list.

4 Click **Next**.



5 In the **Select MedML file** field, type the full path name of the XML file to import, or click **Browse** and locate the file. Any of the following options are valid:

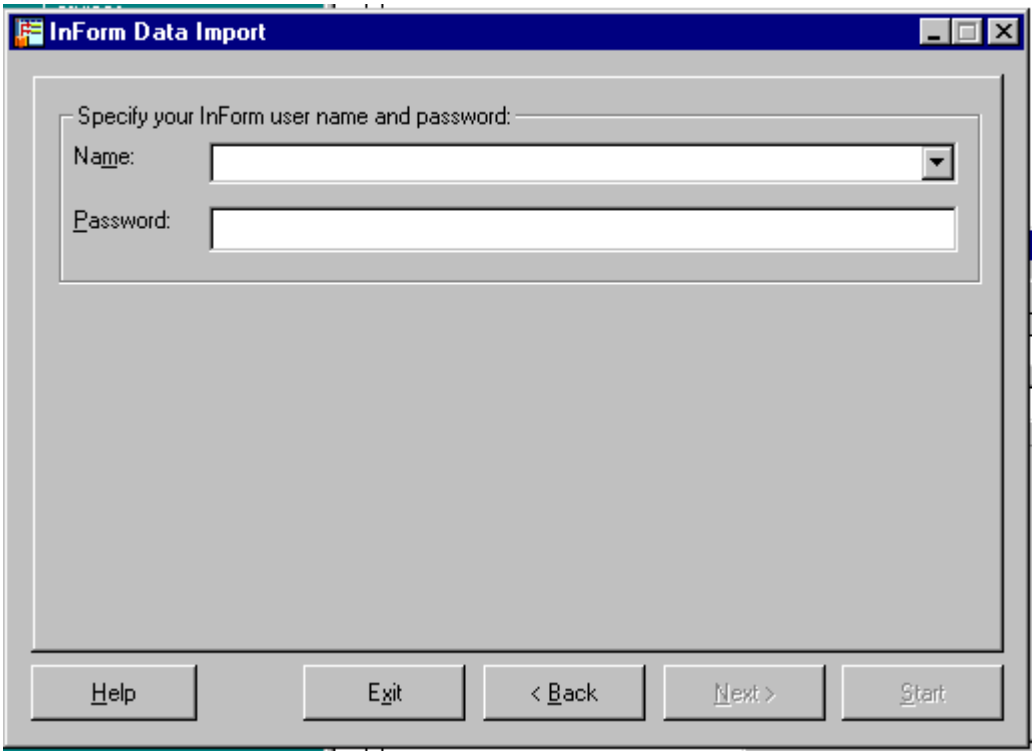
- The name of one XML file.
- The name of multiple XML files, each separated by a space.
- The name of a response file that contains multiple XML files, one per row, in the following format:

  @filename

6 Select **Parse Only** to run the import without actually importing data into the database. Use this function to test the syntax of your MedML file before you import it into the database.

7 Click **Next**.

A dialog appears and requests your InForm name and password.

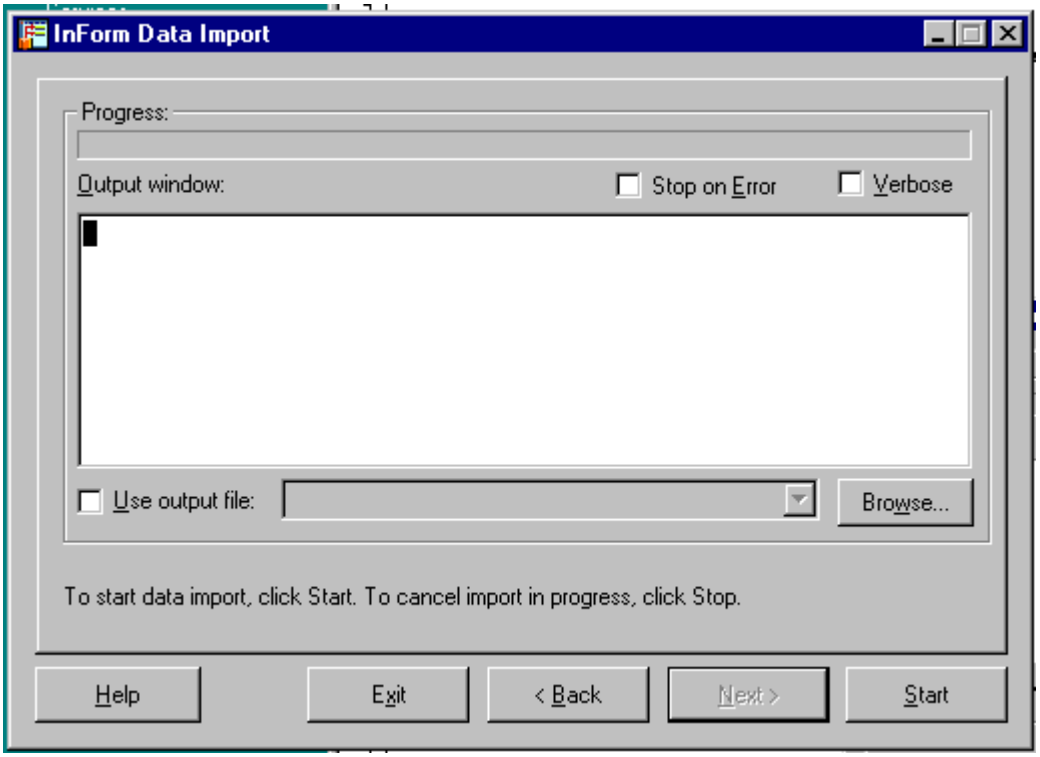


8 In the **Name** field, type the name of an InForm user who has the appropriate rights for the data you are importing:

To import data that matches this InForm system activity	User needs these rights
Add patient clinical data	Enter Data into a CRF
Update patient clinical data	Edit Data on a CRF

9 In the **Password** field, type the user password, then click **Next**.

The Summary window appears.



Note: If you selected **Parse Only**, you do not need to specify a name and password.

10 Optionally, select any of the following:

- **Stop on Error**—To instruct the InForm Data Import to stop if it encounters an error.
- **Verbose**—To instruct the InForm Data Import to generate detailed messages as it processes the file.
- **Use output file**—Specify the filename to save the output file as a text file.

11 Click **Start**.

The InForm Data Import processes the import file, writes messages to the message area and the output file, if specified, and adds or updates data in the database.

12 Close the InForm Data Import. To view the data that you imported, you must stop and restart the trial.

Importing an XML file

To import an XML file:

1 Create an XML file that contains the patient data to add or update. For more information, see Creating an XML import file.

2 Run the import with the MedML file option. For more information, see ***Running the import using the XML import file*** (on page 49).

Importing coded items

Creating an autocode import file

The import file that is used in the autocoding process must originate from the InForm Data Export utility, and must have been run previously on the same server. The source and destination must be the same machine.

This example is a sample autocode file format that was generated by the InForm Data Export utility:

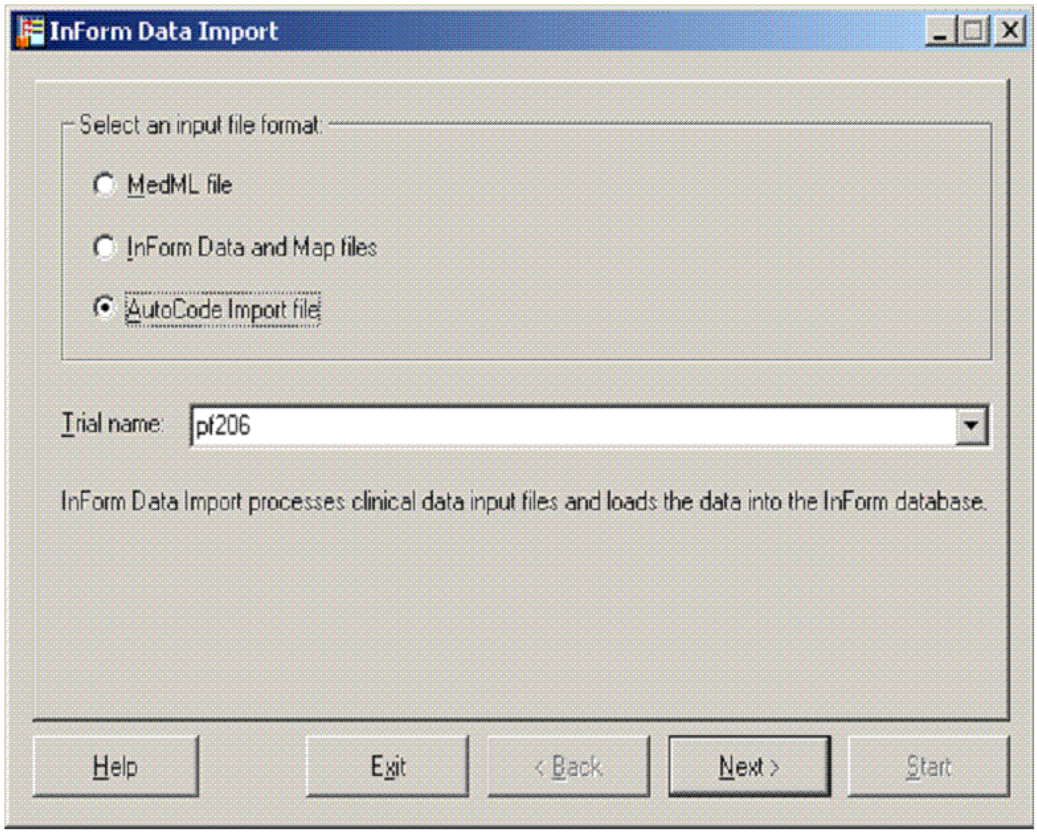
```
<AUTOCODEDATA>

<AUTOCODESET CODESETTYPE="AE">

<AUTOCODE VERBATIM="Headache" CODEVALUE="HED">
<CODEMAP
CODEMAPKEY="840C35C7-BD2D-4C3A-AA4E-EF396CDCD44B"
    SOURCEID="FDBFA029-B175-47D4-B16D-D6618E90D8A0"/>
</AUTOCODE>

<AUTOCODE VERBATIM="Vomiting" CODEVALUE="VMT">
  <CODEMAP
     CODEMAPKEY="5C05A861-91F5-4DDE-87C3-CB04634A7520"
     SOURCEID="B18EF874-8503-4058-B364-49BE41744003"/>
</AUTOCODE>

</AUTOCODESET>
</AUTOCODEDATA>
```

Running the import using an autocode import file

To run the import:

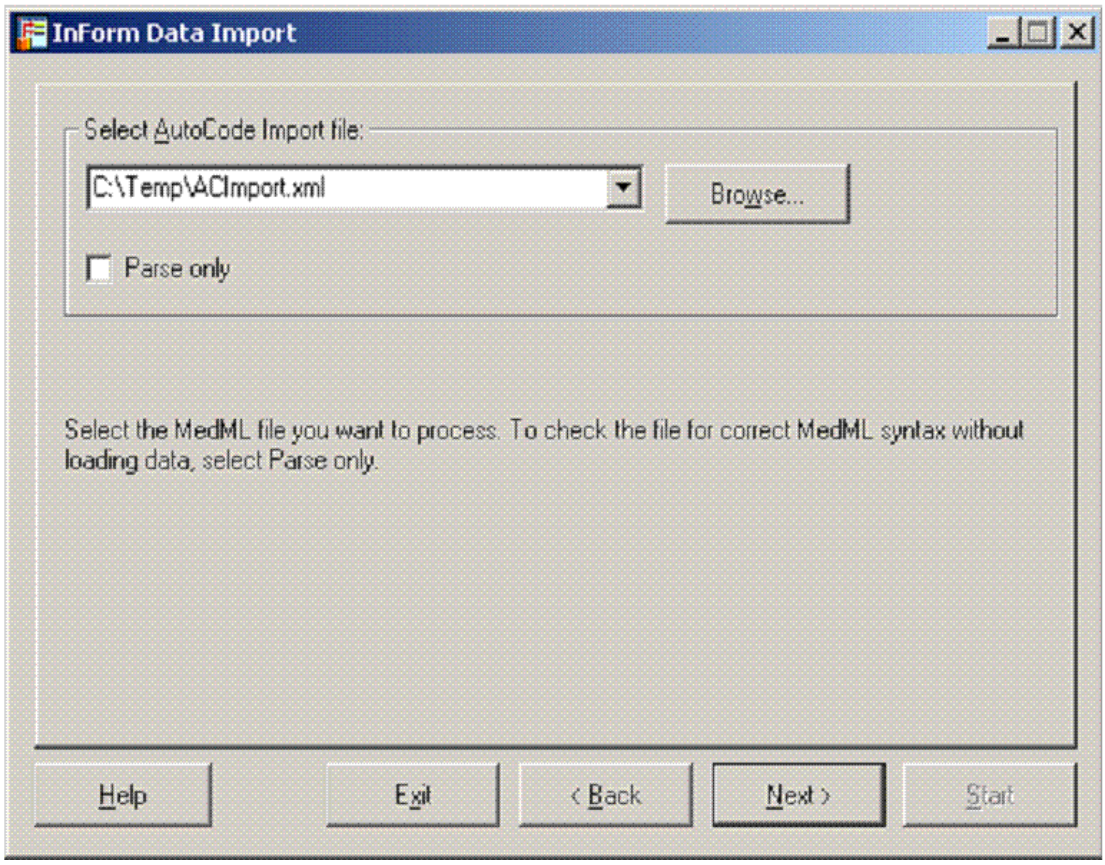
1 Select **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm Data Import.**

2 When the InForm Data Import window appears, select the **AutoCode Import file** radio button.



3 Select a trial.

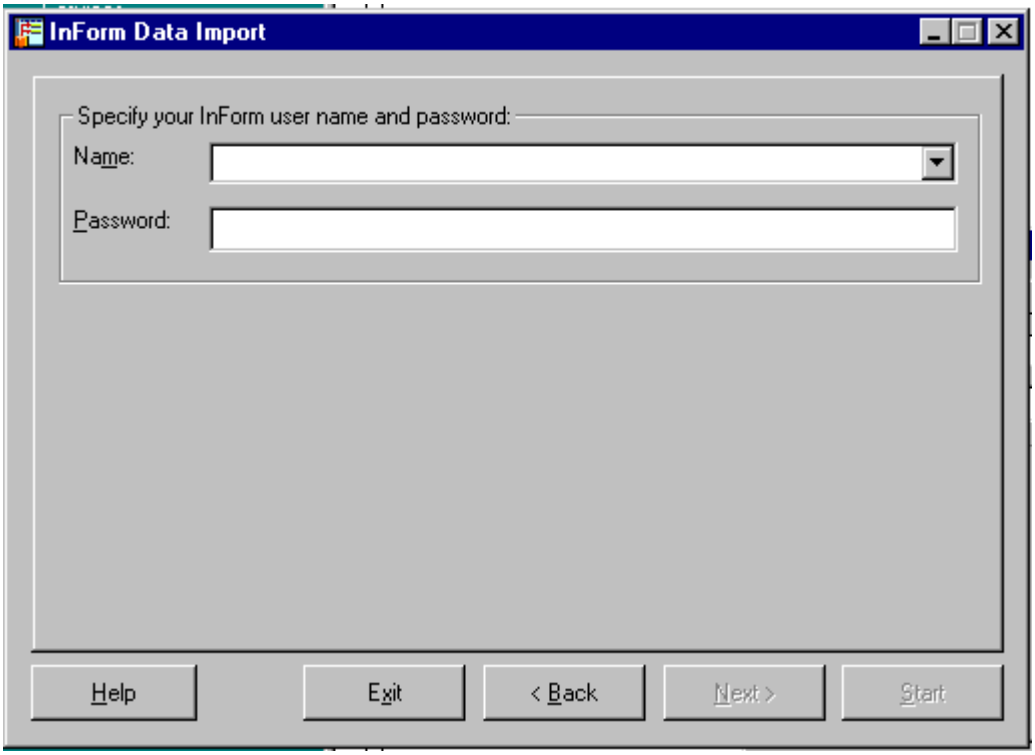
4 Click **Next**.



5 In the **Select AutoCode Import file** field, type the full path name of the autocode XML file to import, or click **Browse** and locate an existing autocode XML file.

Note: This input file must originate from the InForm Data Export utility from the same source and destination machine.

A dialog appears and requests your InForm name and password.

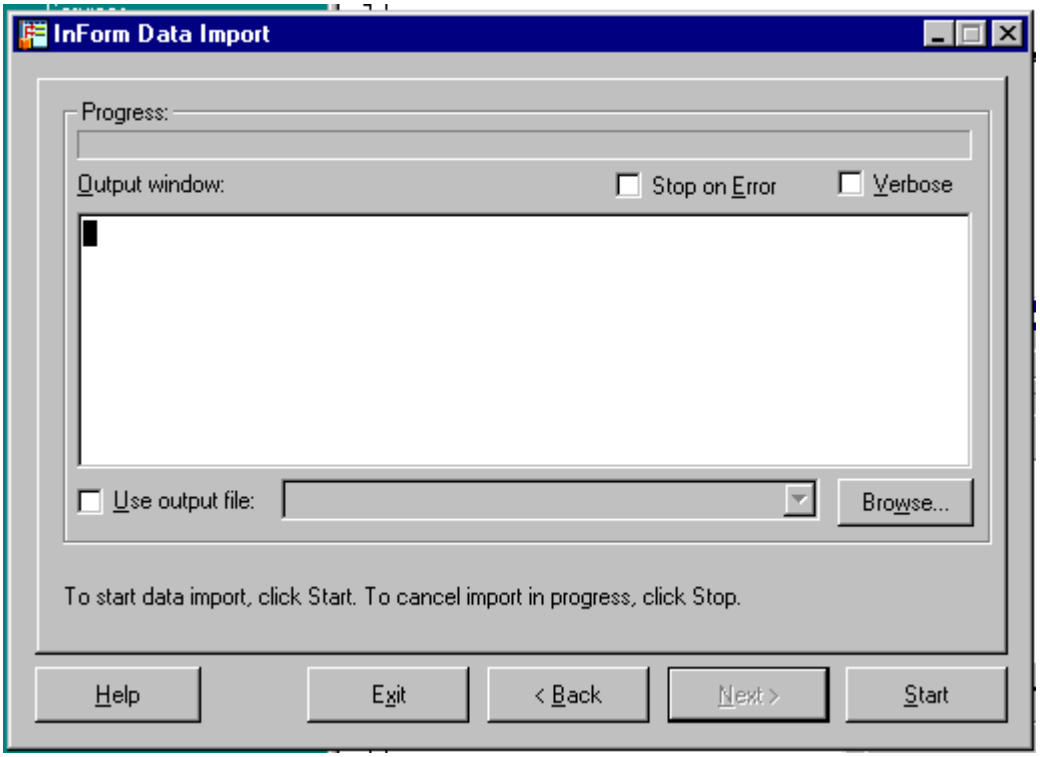


6 In the **Name** field, type the name of an InForm user who has the appropriate rights for the data you are importing:

To import data that matches this InForm system activity	User needs these rights
Add patient clinical data	Enter Data into a CRF
Update patient clinical data	Edit Data on a CRF

7 In the **Password** field, type the user password, then click **Next**.

The Summary window appears.



8 Optionally, select any of the following:

- **Stop on Error**—To instruct the InForm Data Import to stop if it encounters an error.
- **Verbose**—To instruct the InForm Data Import to generate detailed messages as it processes the file.
- **Use output file**—Specify the filename to save the output file as a text file.

9 Click **Start**.

The InForm Data Import processes the import file, writes messages to the message area and the output file, if specified, and adds or updates data in the database.

10 Close the InForm Data Import. To view the data that you imported, you must stop and restart the trial.

Date and time validation

The InForm Data Import utility performs the following validation checks for datetime data:

Validation check	Description
DISPLAY attribute of CRF datetime control.	The MedML DISPLAY attribute must be set to True for the components of the CRF datetime control into which a date, time, or datetime item is being imported. The InForm Data Import utility ignores date and time components for which the DISPLAY attribute is False, and it does not import them.
Year value of import field.	If the imported year is outside the range allowed in the CRF datetime control, the InForm Data Import utility issues an error.
Day value of import field.	If the Year and Month values are valid, the InForm Data Import utility checks for a valid day value.
REQUIRED attribute of CRF datetime control.	If any datetime component is coded as REQUIRED in the CRF datetime control and is missing in the import file, the InForm Data Import utility issues an error.
UNKNOWN attribute of CRF datetime control.	If an imported datetime component is coded UNK, but the UNKNOWN attribute of the CRF datetime control is not enabled, the InForm Data Import utility issues an error.
Consistency checking.	If consistency checking is enabled in the CRF datetime control, the InForm Data Import utility issues an error if a datetime component is present in the import file without a higher component, or an UNK value, in the following sequence: `ss:mm:hh dd/mm/yy` For example, if the imported datetime includes a Day field without a Month, the InForm Data Import utility issues an error.
Hour and Minute values of import field.	The values for both Hour and Minute must be within valid ranges or coded as UNK.

Additional InForm Data Import attributes

Attribute	Description	Available values
CLEARCRF	Indicates whether to clear the specified form.	TRUE, FALSE
COMMONFORM	Indicates whether a form is a common CRF.	TRUE, FALSE
DELETEITEMSET	Indicates whether to delete an itemset.	TRUE, FALSE
FORMSTATUS	Modifies the status of a form.	• FREEZE • UNFREEZE • LOCK • UNLOCK • DELETE • UNDELETE
OVERRIDE	Specifies whether a patient screening or enrollment has been overridden.	TRUE, FALSE
UNDELETEITEMSET	Indicates whether to undelete an itemset.	TRUE, FALSE

Running the InForm Data Import from the command line

While you are creating or editing your map file, use the InForm Data Import in interactive mode. After your map file is stable, it may be more convenient to run the utility from the command line, or to insert the import command in a batch file that is scheduled to run periodically.

Note: Phase Forward strongly recommends that you run the InForm Data Import through the PFConsole utility. For more information, see *Using the PFConsole utility* (on page 7) for detailed information about the PFConsole utility.

A space is required between the parameter name and its value.

```
PFImport [-?] [-autorun] [-verbose] [-parse] [-trial <trialname>] [-errstop]
-norules [[-user <username>] [-password <password>] [@<rsp_file>] [-xml
<xml_file>...<xml_file>]] [-AUTOCODE] [-template <map_file>] [-import
<data_file>]
```

Use the following command line parameters. You must include a space between the parameter name and its value.

Parameter	Variable	Description
PFImport		Starts the InForm Data Import.
-?		Displays command help.
-autorun		Runs the InForm Data Import in a command window.
-verbose		Indicates that the InForm Data Import will write detailed messages as it processes the input file.
-parse		Indicates that the import will run without actually importing data into the database. Use this function to test the syntax of your MedML file before it is imported into the database.
-trial	*trialname*	Specifies the trial into which you are importing data. Use the full pathname of the trial.
-errstop		When the errstop parameter is specified, the InForm Data Import will stop processing when it encounters an error. When errstop is not specified, the tag containing the error is skipped and the import continues with the next data tag in the file.
-norules		Indicates that rules will not run during the import. This parameter is required. This option is valid only if there are no synchronization connections defined (when using the InForm Unplugged application).
-user	*username*	The name of an InForm user who has the appropriate rights for the data you are importing.
-password	*password*	The password associated with the user specified by the user parameter.

Parameter	Variable	Description
-validate		Checks to make sure that all required XML tags exist and the specified control paths can be found, without loading data. Optionally, use this parameter to validate an XML file before importing it.
-unit		Converts imported units to base units and stores both imported and base values. To use this parameter, the Rule Manager (InFormRule MTS package) must be running.
@	*RSP_file*	Indicates that you are submitting a response file (RSP extension) that contains the names of the XML files to process. Specify the full pathname of the response file. Use this option or the -xml option or the -template and -import options.
-xml	*xml_file*	Indicates that you are submitting one or more XML files to process. If you are submitting multiple XML files, separate them with a space. Use this option, the @ option, or the -template and -import options.
-AUTOCODE		Indicates that the specified XML file or files contain autocode data.
-template	*map_file*	If you are using the data and map file option, indicates that the next parameter is the map file name. Specify the full pathname of the map file. Use this option or the @ option or the -xml option. If you specify a map file, you must also specify an import file with the -import option.
-import	*data_file*	Indicates that the next parameter is the import file name. Specify the full pathname of the data file. Use this option or the @ option or the -xml option. If you specify an import file, you must also specify a map file with the -template option.
-output	output_file	Indicates that you want to save the output in a text file. Specify the full path and file name you want to give to the file. When you use this parameter, output messages are not displayed in the Output window.

Enhancing your data import

Log off of the InForm application server

Log off of the InForm application server before running the import. You can keep the server running.

Stop the WWW Publishing Service

Stop the WWW Publishing Service to prevent others from entering data as you perform an import. Make sure you turn the WWW Publishing Service back on when your import is complete.

Run the Oracle update statistics script

Run the Oracle update statistics script immediately after you import files. A single import can enter in as much data as you might manually enter in 6 weeks.

Change the home page

Before you start the import, change the home page of the trial and warn users that they may experience slowness in the system while you are running the import.

Organize by patient

If possible, sort the data you are importing by patient ID.

CHAPTER 5

Using the MedML Installer utility

In this chapter

Overview of the MedML Installer utility 66

Setting up a trial with the MedML Installer utility 67

Validation checks 68

Launching the MedML Installer utility 70

About the MedML Installer utility window 71

Running the MedML Installer utility 73

About MedML Installer utility output messages 75

Running the MedML Installer utility from the command line 76

Overview of the MedML Installer utility

The MedML Installer utility:

- Parses the XML files that you generate with MedML tags.
- Validates the files against the following:
  - The MedML schema file (MedMLSchema.xsd) that is provided in the Phase Forward base installation.
  - The constraints enforced by the database schema.
- Loads the study components that the files define into the database.

Setting up a trial with the MedML Installer utility

The study database stores the data collected during a specific trial, including clinical data and tables that define the following information:

- CRF contents and format.
- Time and events schedule.
- Edit checks.
- User and site information.
- Sponsor personnel.
- System configuration information.
- Workflow assignments and user permissions.

After all trial components have been defined, use the MedML Installer utility to load the following definitions:

- The Base study, which is delivered with the InForm application and contains components that are common to trials.
- The trial-specific definitions for components in your trial. These definitions are found in a set of XML files.

To specify the information that defines a study in a study database:

1 Create a set of XML files that define all of the components of a study, using the MedML schema.

2 Process the XML files through the MedML Installer utility.

For more information, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

Validation checks

When you install metadata with the MedML Installer utility, the program checks the metadata to make sure that the MedML syntax is correct and that all required elements are present.

Selection value checks

The MedML Installer utility performs checks that affect selection values in radio or checkbox group controls. These checks enforce selection value syntax and semantics, and help identify problems before they are incorporated into a trial. Detected errors appear in the output section in the MedML Installer window and also in the output file, if you choose to create one.

The MedML Installer utility:

- Verifies that the selection value attribute is specified for all child simple controls. If the selection value attribute is incomplete, an error occurs in both strict and nonstrict modes.

  Note: In nonstrict mode, the InForm Architect application allows you to load definitions that are not fully compliant with the MedML specification for trial design purposes only. Once the trial is complete, use strict mode to ensure that the definitions you load are fully compliant with the MedML specification. For more information, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

- Verifies that the selection value attribute is specified for all children of a group control. If the selection value attribute is incomplete, an error occurs in both strict and nonstrict modes.
- Checks for data type consistency among all child controls based on the following criteria:
  - If there are no simple control children, all selection values are assumed to be strings and therefore, no data type consistency is enforced.
  - If one or more children are simple controls, then all simple controls must be of the same data type (for example, string, integer, or float) and user-defined selection values for non-simple controls must be of the same data type.

  If you are processing in strict mode and the data type among child controls is inconsistent, an error occurs. If you are processing in nonstrict mode, a warning message appears.
- Checks for duplicate selection values. If duplicate values exist, an error occurs in both strict and nonstrict modes.
- Checks for empty selection values for non-simple child controls. If an empty selection exists, a warning message appears if you are using verbose mode in both strict and nonstrict modes.

Coding mapping checks

When you use the Central Coding application to code patient data in an InForm trial, trial designers must develop mapping definitions to specify the:

- Items to be coded.
- Items that hold coded data.
- Relationship between those items listed above and specific code targets and context items in

a coding dictionary.

When installing coding mapping definitions, the MedML Installer utility checks that the:

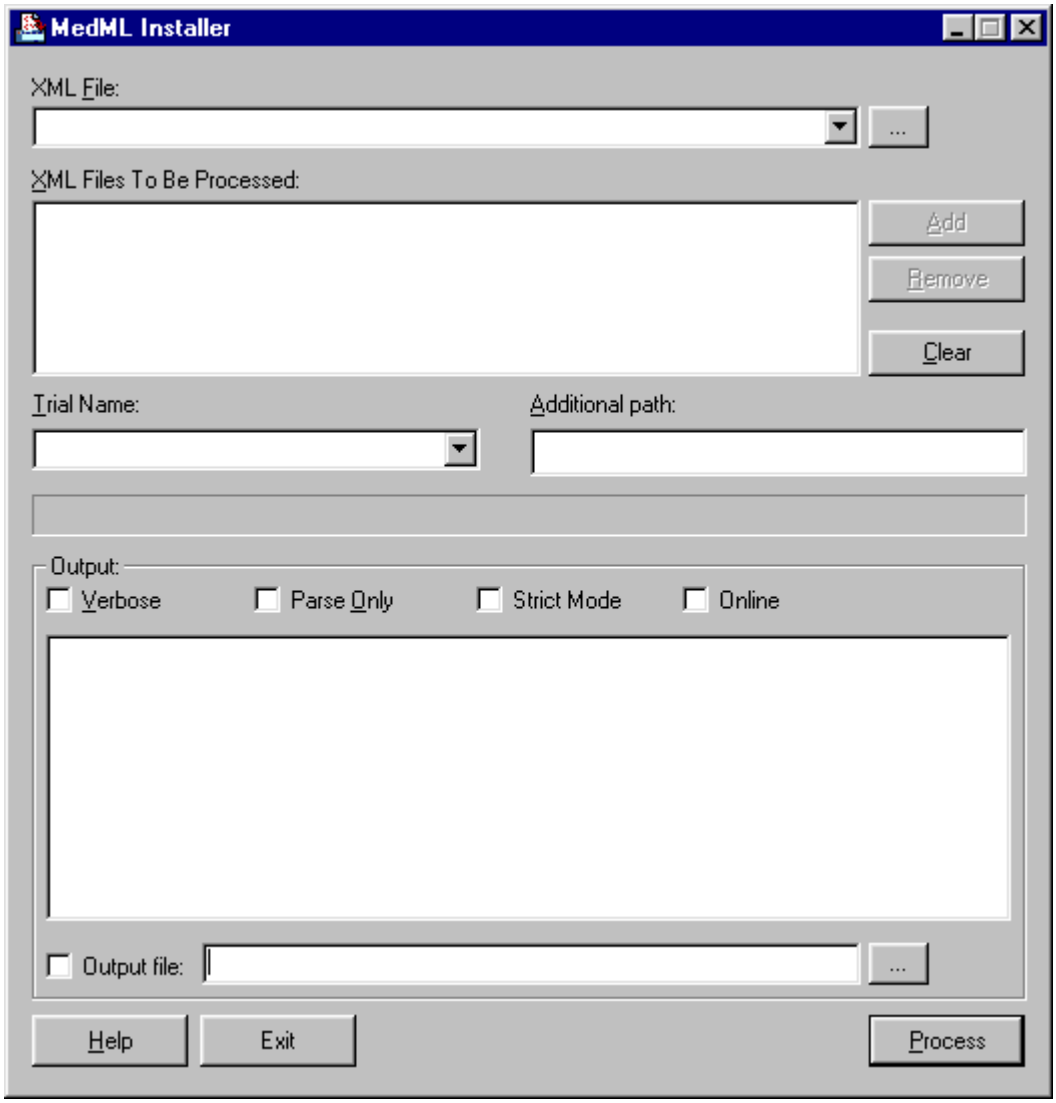
- Verbatim path exists and is complete.
- Specified dictionary is valid.
- Specified code targets and context items are valid.
- Code target and context item:
  - Names are unique within a mapping.
  - Paths exist and are complete.
- Code target paths point to top-level field controls or calculated controls.
- Repeating parts of a verbatim path and its context item paths, or a verbatim path and its code target paths, agree with each other. Verbatim and code target controls, or verbatim and context item controls, must be on the same form if the form is repeating and must be in the same itemset if the coded data appears in an itemset.
- Specified verbatim type is valid.

Launching the MedML Installer utility

To launch the MedML Installer utility, do one of the following:

- Select **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm MedML Installer**.
- Run the utility from the command line. For more information, see *Running the MedML Installer utility from the command line* (on page 76).

About the MedML Installer utility window



Field	Description
XML File	The name of the file to add to (or modify in) the build.
XML Files to be Processed	A list of all of the files to be included in the next build.
Trial Name	The name of the trial on which you are working.
Additional Path	If you are using a response (RSP) file to process a group of XML files, type the name of the root directory where the XML files that are referenced in the RSP file are located.
Verbose	Instructs the MedML Installer utility to create a detailed description of what happened during the build.

Field	Description
Parse Only	Instructs the MedML Installer utility to check the input file for XML errors and compatibility with the InForm database without loading data.
Strict Mode	Instructs the MedML Installer utility to accept only complete component definitions. By default, the checkbox is not selected, indicating nonstrict mode. In nonstrict mode, the MedML Installer utility accepts incomplete component definitions in which not all dependent components are present. For example, it accepts a radio control definition in which not all of the element definitions have been loaded previously into the database. **Note:** This option is available only if you start the utility from the command line with the -notstrict option. Nonstrict mode is intended only for a trial development environment. Do not use it in production. If you use the InForm Unplugged application in production to synchronize multiple copies of a trial on distributed computers, you cannot create a connection if you previously loaded trial definition data on the source computer in nonstrict mode. Similarly, once a connection is defined, you can only load trial definition data with strict MedML processing.
Online	Instructs the MedML Installer utility to start the trial in online mode. By default, the checkbox is selected. In online mode, if a trial is not started when you start the MedML Installer utility, the utility starts the trial and does not shut it down when the installation is complete. This mode enables users to view the results of installing trial definition data immediately. If you deselect this checkbox, the MedML Installer utility starts the trial in offline mode. In offline mode, the MedML Installer utility stops the trial when installation is complete. Users cannot connect to the trial until it is restarted. **Note:** This option is available only if you start the tool from the command line with the -online option.
Output file	A report of what happened during the build.

Definitions are processed from the bottom up. Make sure that your component definitions and XML files are organized in the correct order. The elemental components, such as controls, must be defined, imported, and read first, followed by items that reference them.

For more information, see *Running the MedML Installer utility from the command line* (on page 76).

Running the MedML Installer utility

Running the MedML Installer utility for the first time

To start testing your XML files in the trial, use the MedML Installer utility to build the files into the trial database.

To run the MedML Installer utility:

1 Launch the MedML Installer utility. For more information, see *Launching the MedML Installer utility* (on page 70).

The MedML Installer window appears.

2 In the **XML File** field, select the XML file to process, or click **Browse** and locate the file.

> **Note:** You can also list all the files to process in a response (RSP) file and type the response file name in the **XML File** field.

3 Click **Add**.

4 Repeat steps 1-3 for all your XML files.

5 From the **Trial Name** drop-down list, select the name of the trial.

6 Optionally, select additional checkboxes in the MedML Installer window to specify the way that the MedML Installer utility will process the files. For more information, see *About the MedML Installer window* (on page 71).

7 Click **Process**.

The MedML Installer utility processes the XML files and loads information into the trial database. When all the XML files have been processed, the message **Completed Successfully** appears in the MedML Installer Output window.

For more information on error messages, see *About the MedML Installer output messages* (on page 75).

Updating a trial that is already in progress

When you change a file in a trial that is already in progress, you must save it, determine which files are impacted by the changes, and process those files again. You must also reprocess any file that references the items in the file to which you made changes. For example, if you make a change to a section, you must reprocess all files that contain references to that section.

When you use the MedML Installer utility to update a trial that is already in progress, you must stop and restart the trial definition with the PFAdmin utility for the changes to take effect.

To update a trial that is already in progress:

1 Limit access to the trial by changing the Directory Security properties of the trial's virtual directory in **Internet Information Server** through **Microsoft Windows Components**. For more information, see the *Installation and Configuration Guide*.

2 Launch the MedML Installer utility.

The MedML Installer window appears.

3 In the **XML File** field, select the file to which you have made changes.

4 Click **Add**.

The XML file appears in the list of XML files to be processed.

5 From the **Trial Name** drop-down list, select the name of the trial**.**

6 Optionally, select additional checkboxes in the MedML Installer window to specify the way that the MedML Installer utility will process the files. For more information, see *About the MedML Installer window* (on page 71).

7 Click **Process**.

The MedML Installer utility processes the XML files and loads information into the trial database. When all the XML files have been processed, the message **Completed Successfully** appears in the MedML Installer Output window.

For more information on error messages, see *About the MedML Installer output messages* (on page 75).

The XML file is included in the build the next time you click **Process**.

8 Stop and restart the trial. For more information, see the *Installation and Configuration Guide* or the InForm online Help.

Removing XML files from the build

Regulatory authorities have strict regulations against removing data from a trial. XML files that are part of a trial cannot be removed. You can instead remove all references to any unnecessary files in the other XML files, then reprocess all the files using the MedML Installer utility.

About MedML Installer utility output messages

The MedML Installer utility generates three types of output messages:

- **Informational**—Describe conditions that probably will not cause serious problems with your build.
- **Warning**—Describe events that might be detrimental and should be fixed, but do not stop the build process.
- **Failure**—Describe events that stop the build process.

If the MedML Installer utility does not complete successfully, to determine where the error occurred:

- Check the output messages.
- Make sure that you started the InForm application before you ran the MedML Installer utility.
- Make sure that the build is in the correct directory.
- Make sure that all the parameters are accurately fulfilled.
- Check for correct paths to input files.
- Make sure components are processed in the correct order and that all necessary components are present in the XML file or database.

Running the MedML Installer utility from the command line

Note: Phase Forward strongly recommends that you run the MedML Installer utility through the PFConsole utility. For more information, see ***Using the PFConsole utility*** (on page 7).

To run the MedML Installer utility from the command line, use the following syntax:

```
PFConsole PFMMinst -trial trialname [-verbose] [-?] [-help] [-autorun] [-
notstrict] [-online] [-outfile output file name] [-parse] -xml [@]
response_file ... [@] XMLFileN
```

Note: Parameters in brackets [ ] are optional.

Command line parameters

Parameter	Variable	Description
PFConsole utility		Starts the PFConsole utility.
PFMMinst		Starts the MedML Installer utility.
-trial	trialname	The name of the trial you are building.
-verbose		Specifies whether to generate details of the build and display them during the build.
-?		Displays the syntax of the command line for running the MedML Installer utility.
-help		Opens the online help for the MedML Installer utility.
-autorun		Runs the MedML Installer utility in a command window. Phase Forward recommends that you use the PFConsole utility to run the MedML Installer utility from the command line. For more information, see ***Using the PFConsole utility*** (on page 7).

Parameter	Variable	Description
-notstrict		Opens the MedML Installer dialog, which has a checkbox for specifying strict mode. By default the checkbox is empty, indicating nonstrict mode. In nonstrict mode, the MedML Installer utility accepts incomplete component definitions in which not all dependent components are present; for example, it accepts a radio control definition in which not all of the element definitions have been previously loaded into the database. **Note:** Use nonstrict mode only in a trial development environment. Never load production trial definitions in nonstrict mode. If you use the InForm Unplugged application to synchronize multiple copies of a trial on distributed computers, the software prevents you from creating a connection if you have previously loaded trial definition data on the source computer in nonstrict mode. Once a connection is defined, you only load trial definition data with strict MedML processing.
-online		Opens the MedML Installer utility dialog box with a checkbox available for specifying online mode. By default the checkbox is selected. • **Online mode**—If a trial is not started when you start the MedML Installer utility, the utility starts the trial and does not shut it down when the installation is complete. This mode allows users to view the results of installing trial definition data immediately. • **Offline mode**—If a trial is not started when you start the MedML Installer utility, the utility starts the trial, loads the trial definition data, and stops the trial when complete. Users cannot connect to the trial until it is restarted.
-outfile	output filename	Specifies whether to save the details of the build in an output file, and indicates the path and file name in which to save it.
-parse		Instructs the utility not to commit the build to the database. If you run the MedML Installer utility in Parse mode, all information is run against, but not committed to, the database. This mode is useful for testing for errors.
-xml	XMLFile	Specifies the name of the XML file to include in the build.
@<response>		Specifies the name of a response file that consists of a list of XML files to be processed.

CHAPTER 6

Using the InForm Data Export

In this chapter

Overview of the InForm Data Export utility 80

Running the InForm Data Export utility 81

Exporting coded controls 83

Exporting data into a CDD 87

Exporting Name Value Pairs 90

Exporting data in Oracle Clinical format 93

Running the InForm Data Export from the command line 98

Overview of the InForm Data Export utility

InForm Data Export output options

The InForm Data Export exports data from the InForm database in several formats. The following output options are available in the InForm Data Export:

- **AutoCode**—Facilitates the use of external coding tools to code items.
- **CDD**—Exports data into a Customer-Defined Database.
- **Name Value Pairs**—Creates a pipe-delimited file that consists of data path names and data values.
- **Oracle® Clinical**—Creates a text file that can be used to import information into an Oracle Clinical database.

Note: Use of the InForm Data Export is limited to trials with fewer than 100,000 patients.

Running the InForm Data Export utility

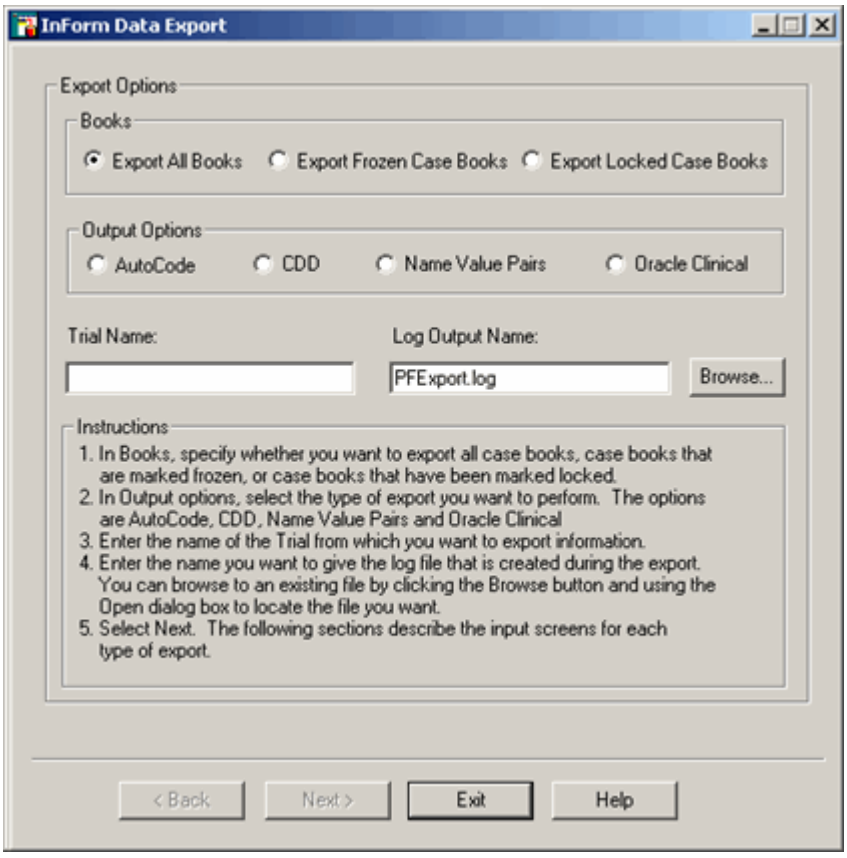
All of the InForm Data Export utility options run under the same executable.

Note: To run the InForm Data Export utility, the InForm application server does not need to be running.

To run the InForm Data Export utility:

1 Click **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm Data Export**.

The InForm Data Export main window appears.



2 In the **Books** section, select which cases to export.

3 In the **Output Options** section, select the format to use for the export. The options are:

- AutoCode
- CDD
- Name Value Pairs
- Oracle Clinical

For more information, see *InForm Data Export output options* (on page 80).

4 In the **Trial Name** field, type the name of the study from which you want to export

information.

5 In the **Log Output Name** field, type the path and file name to give the log file that is created during the export, or click **Browse** and locate the file to use as the log file.

6 Click **Next**.

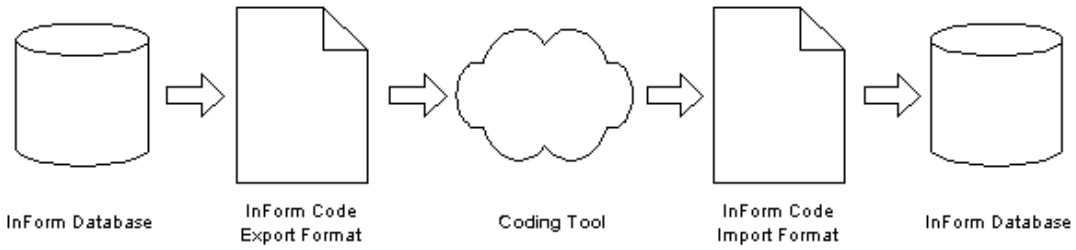
The following sections describe the input screens for each type of export:

- *Exporting coded controls* (on page 83).
- *Exporting data into a CDD* (on page 87).
- *Exporting Name Value Pairs* (on page 90).
- *Exporting data in Oracle Clinical format* (on page 93).

Exporting coded controls

The InForm Data Export utility can export source item values (Verbatim values) and their corresponding destination targets (Code Value values) with information that allows the values to be mapped back during import.

The coding process uses several steps, as shown in the flowchart:



For more information, see *Importing coded items* (on page 55).

For more information on coding the controls through the InForm Architect application, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

Required elements

The following data elements are required during export:

- SOURCE (VERBATIM) and TARGET (CODE VALUE) control mapping information.
- The full path of the SOURCE controls that are carried by the CODEMAPKEY and SOURCEID attributes, and are exported as GUIDs.
- The original value of the source control. This is also required during import in case the value of the source control is changed between export and import.
- The value of the code (destination) control.

Example

Example of the XML code:

```
<AUTOCODEDATA>
<AUTOCODESET CODESETTYPE="AE">
<AUTOCODE VERBATIM="Headache" CODEVALUE="">
  <CODEMAP
     CODEMAPKEY="840C35C7-BD2D-4C3A-AA4E-EF396CDCD44B"
     SOURCEID="FDBFA029-B175-47D4-B16D-D6618E90D8A0"/>
</AUTOCODE>
<AUTOCODE VERBATIM="Vomiting" CODEVALUE="">
  <CODEMAP
     CODEMAPKEY="5C05A861-91F5-4DDE-87C3-CB04634A7520"
     SOURCEID="B18EF874-8503-4058-B364-49BE41744003"/>
</AUTOCODE>
</AUTOCODESET>
</AUTOCODEDATA>
```

This XML is used for both export and import. In this sample, the CODEVALUE tag is empty, which indicates that the verbatim is not yet coded. During export, the tag could have a value if the data that is exported has coded values that need to be recoded. After import, the

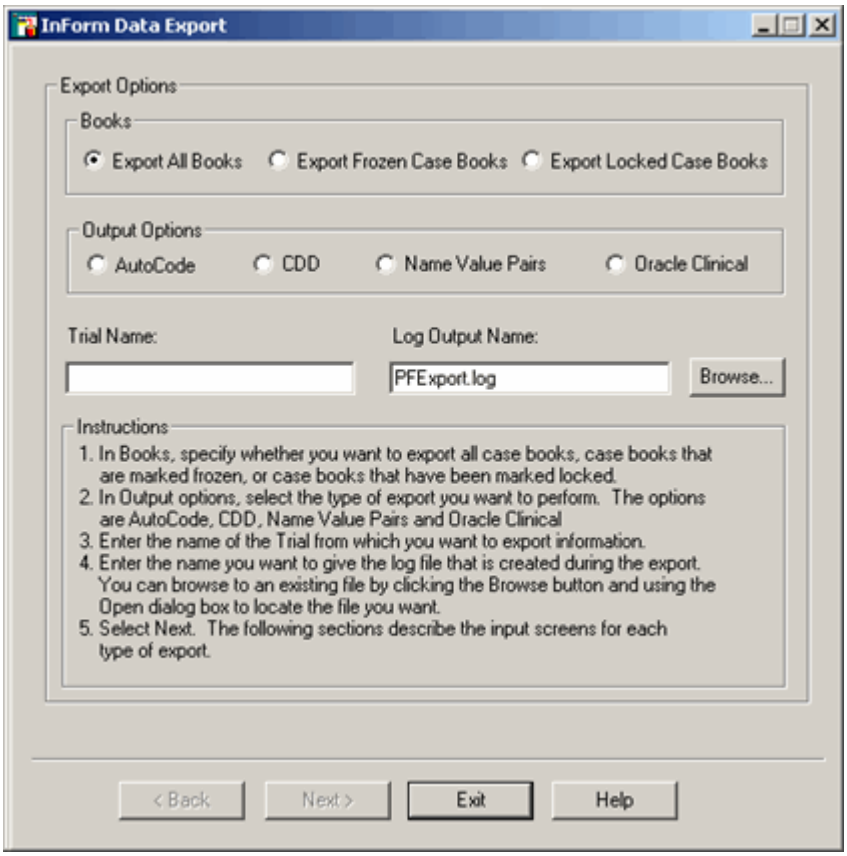
CODEVALUE tag will have values.

The complete set of items to be coded is wrapped in an AUTOCODESET object with the qualifier of *type*. The type is specified at design time, in order to allow autocode fields to be grouped more easily.

Running the export for AutoCode items

To run the export:

1 Click **Start** > **Programs** > **Phase Forward**> **InForm 4.6** > **InForm Data Export**.

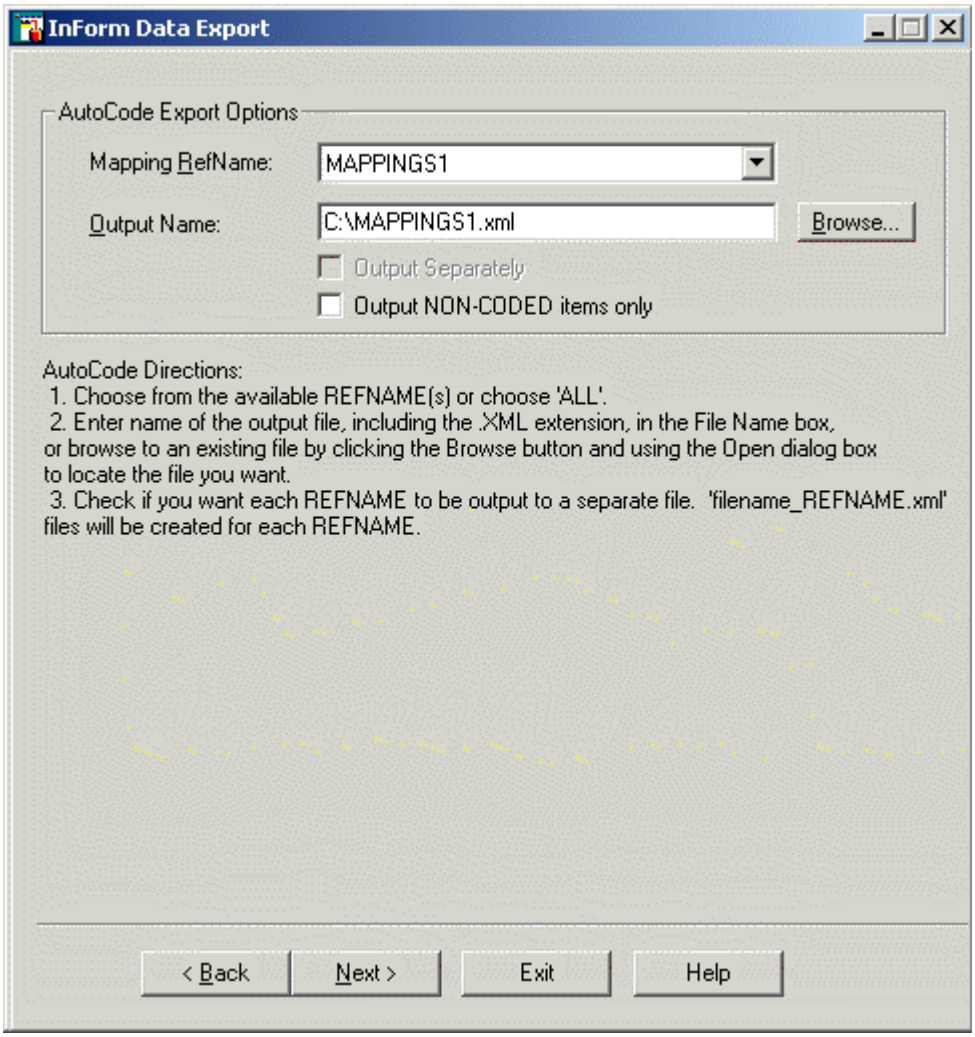


2 In the Export Options window:

- In the **Books** section, select which case books to export.
- In the **Output Options** section, select **Autocode**.
- Specify the **Trial Name** and the **Log Output Name**, or click **Browse** and locate the log output name.

3 Click **Next**.

The AutoCode Export Options window appears.



4 In the **Mapping RefName** drop-down list, select an autocode target set(s). To export all target sets, select **All**.

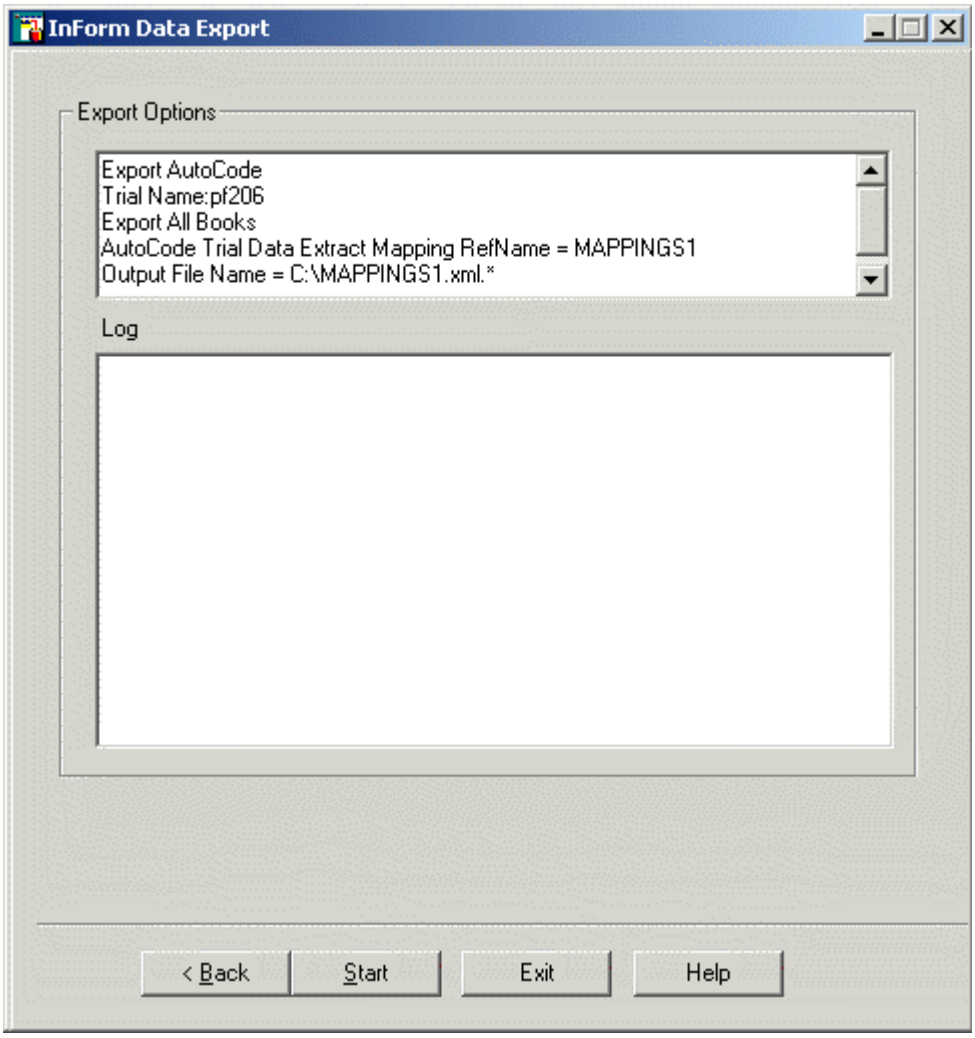
5 In the **Output Name** field, type the name of the output file, including the XML extension, or click **Browse** and locate the file.

6 Select **Output Separately** to export each mapping RefName to a separate file. A file with the name OutputName_REFNAME.xml will be created for each mapping RefName that is defined in the system.

7 Select **Output NON-CODED items only** to output only non-coded items.

8 Click **Next**.

The **Log** section appears below the **Export Options** section.



9 Click **Start**.

Exporting data into a CDD

Overview

The **CDD** output option reads the CDD mapping parameters and all of the data in the InForm database, then populates the CDD in the following order:

- Site data
- Comments
- Patient information
- CRF data

This is useful for repopulating the database when the CDD definition changes after a trial starts.

Moving data to a CDD

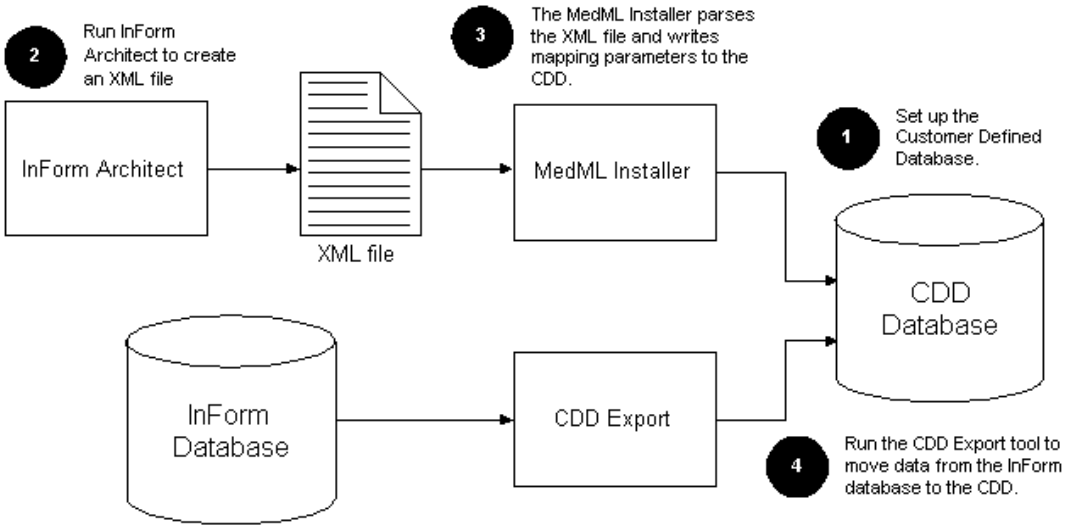
To move data from the InForm trial database to a CDD:

1 Use the InForm Architect application to generate a CDD definition that includes parameters that specify how to map data from the InForm database to the CDD. Save the definition to an XML file. For more information, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

2 Run the MedML Installer utility to parse the XML file and write the mapping parameters to the InForm database. For more information, see ***Using the MedML Installer utility*** (on page 65).

3 Run the InForm Data Export utility using the **CDD** output option to read the data in the InForm database and populate the CDD database. For more information, see ***Running the export for CDD data*** (on page 88).

The flowchart shows the export process:



Running the export for CDD data

Note: Before you run the InForm Data Export utility for CDD data, you should first stop the InForm application server.

To export data to a CDD:

1 On the CDD Export Options window, type the name of the ODBC DSN that is defined for the CDD to which to output data. For more information about creating a new DSN, see *Creating a new DSN for the export* (on page 88).

2 In the **User Name** and **Password** fields, type the user name and password that is used to access the CDD.

3 Select the **New Schema** checkbox to drop the current database and create a new schema based on the RefName associated with the DSN, and type the name of the tablespace in which to put the new schema in the **Table Space** field.

4 Click **Next**.

The Export Summary screen appears.

WARNING: Make sure that all the information is correct before you proceed to the next step. When you run the InForm Data Export utility, all data in the current database will be lost.

5 Click **Finish**.

The InForm Data Export utility begins to populate the CDD and displays messages to indicate its progress.

Creating a new DSN for the export

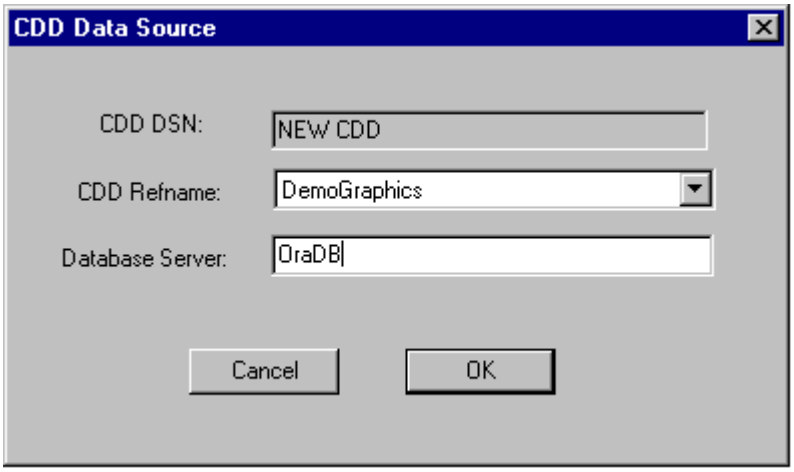
To create a new DSN:

1 On the CDD Export Options window, in the **DSN List** field, type the new name of the ODBC DSN for the CDD to which to output data.

2 Type the user name and password for accessing the CDD.

3 Click **Create DSN**.

The CDD Data Source window appears. Your new DSN appears in the first field.



4 From the **CDD RefName** drop-down list, select the RefName to associate with the new DSN.

5 In the **Database Server** field, type the name of the server to use.

6 Click **OK**.

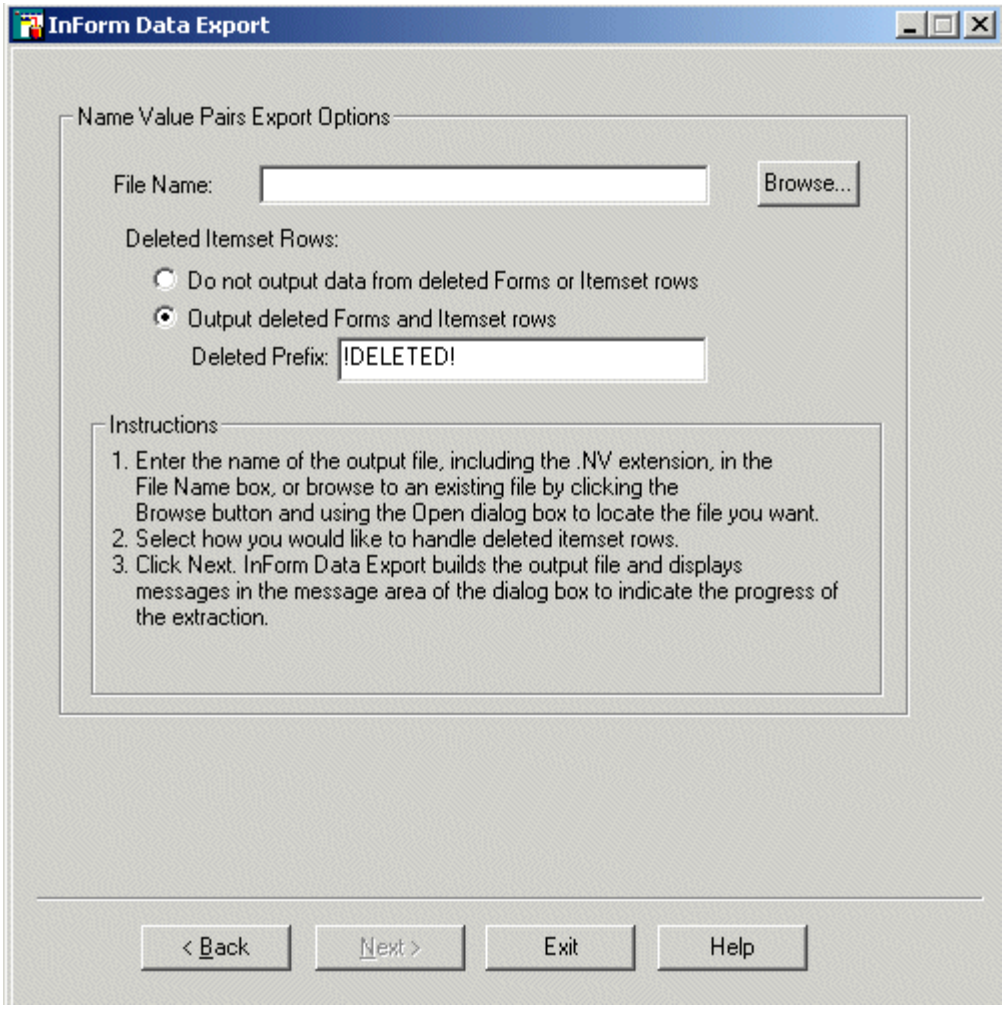
The CDD Export Options window reappears, and the new DSN appears in the drop-down list.

Note: If you create a new DSN for the export, the new DSN is valid only in the InForm Data Export session you are in when you create it. It will be removed when you close the InForm Data Export utility. To create a DSN to use outside of the InForm Data Export utility, use the ODBC Manager or PFAdmin CONFIG CDD Setup commands. For more information on PFAdmin commands, see the *Installation and Configuration Guide*.

Exporting Name Value Pairs

The **Name Value** output option exports data from the InForm database into a plain text file. It is useful for extracting data for conversion or analysis.

Note: The Name Value output option exports only named values that occur in patient data; it does not export values from the Reg Docs or Visit Reports forms.



To export a Name Value data file:

1 In the **File Name** field, type the name of the output file, including the NV extension, or click **Browse** to locate the file.

2 In the **Deleted Itemset Rows** section, select one of the following:

- Select **Do not output data from deleted Forms or Itemset rows** to instruct the InForm Data Export utility not to output data from deleted forms or itemset rows.
- Select **Output Deleted Forms and Itemset Rows** to export data from deleted forms or itemset rows.

  Then, in the **Deleted Prefix** box, specify the prefix to add to each deleted itemset row that

the InForm Data Export utility exports.

3 Click **Next**.

The InForm Data Export utility builds the output file and displays messages to indicate the progress of the extraction.

Output file format

In the output file that is created by running the export for name value pairs, fields are separated by pipes, and inapplicable fields are left blank, using the following data:

Name Value field	Description
Patient identifier	Patient initials followed by patient number in parentheses.
RefName path	Path of RefNames in the following order: • Visit. • Repeating formset index, which is used to indicate which instance of a recurring unscheduled visit is referenced. If the visit is not an unscheduled visit, the value is 1.000. • Form. • Repeating form index. • Section. • Itemset. • Itemset index, which is used to indicate which row of an itemset is referenced. • Item. • Control.
Normalized value	Value after conversion into the base units that are specified in the database.
Entered value	Value as entered for the control.

Output file format for associated forms

In the output file that is created by running the export for associated forms, fields are separated by pipes, and inapplicable fields are left blank, using the following data:

Associated form field	Description
Patient identifier	Patient initials followed by patient number in parentheses.

Associated form field	Description
RefName path	Path of RefNames in the following order. • Visit. • Repeating formset index, which is used to indicate which instance of a recurring unscheduled visit is referenced. If the visit is not an unscheduled visit, the value is 1.000. • Form. • Repeating form index. • Patient. The path for the first form appears in this order, and the path for the associated form appears after the first form, and is surrounded by quotation marks.
Normalized value	Value after conversion into the base units that are specified in the database.
Entered value	Value as entered for the control.

Example

The following export file fragment illustrates some of the data exported for the patient PA1(1).

```
PA1(1)|Visit1|1|EYE|VISION||0|LNEAR.EYEGROUP||
PA1(1)|Visit1|1|EYE|VISION||0|LNEAR.EYEGROUP.EYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|LNEAR.EYEGROUP.UNEYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|RNEAR.EYEGROUP||
PA1(1)|Visit1|1|EYE|VISION||0|RNEAR.EYEGROUP.EYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|RNEAR.EYEGROUP.UNEYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|LDIST.EYEGROUP||
PA1(1)|Visit1|1|EYE|VISION||0|LDIST.EYEGROUP.EYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|LDIST.EYEGROUP.UNEYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|RDIST.EYEGROUP||
PA1(1)|Visit1|1|EYE|VISION||0|RDIST.EYEGROUP.EYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|VISION||0|RDIST.EYEGROUP.UNEYE_TC|20.000000|20
PA1(1)|Visit1|1|EYE|RETINAL||0|RWOOL.WOOLGROUP||N
PA1(1)|Visit1|1|EYE|RETINAL||0|LWOOL.WOOLGROUP||N
PA1(1)|Visit1|1|EYE|RETINAL||0|RNICK.NICKRADIO||N
PA1(1)|Visit1|1|EYE|RETINAL||0|LNICK.NICKRADIO||N
PA1(1)|Visit1|1|HH|PT|PT|1|THERAPYTEXT.THERAPYTEXT||Name of a doctor
```

The following export file fragment is an example of a repeating form:

```
ABC(101)|CommonCRF|1.000|L_ConMeds|291130301735008.000|L_ConMeds||0.000|CMHi
ddenJ.CMHiddenJ||test
```

The following export file fragment is an example of an association:

```
BBB(111)|CommonCRF|1.000|LAE1|319514859682043.000| "BBB(111)|
CommonCRF|1.000|L_ConMeds|319587403827450.000"
```

Exporting data in Oracle Clinical format

Overview

The **Oracle Clinical** output option of the InForm Data Export utility outputs data from the InForm database in a format that you can upload to an Oracle Clinical database using the Oracle Clinical Upload application.

The Oracle Clinical output option of the InForm Data Export utility exports data in the following output file formats:

- **OA6 file**—Main output that contains rows whose ID/key changed since the date of the last run. If a path is not mapped, the InForm Data Export utility does not output the corresponding data. Use the OA6 file when you import data to an Oracle Clinical database.
- **DEL file**—Contains information about deleted records. You can reference this file to manually delete data from external databases.
- **NM file**—Contains the paths of unmapped elements.
- **log file**—Contains information about the export.

Transferring data to the Oracle Clinical upload application

To set up data transfer to the Oracle Clinical Upload application:

1 Use the InForm Architect application to generate an Oracle Clinical mapping definition for the study, and save it in a location that you can access while running the MedML Installer utility.

The mapping definition is a set of XML tags that specify how the controls in the study map to Oracle Clinical database fields. For information about generating an Oracle Clinical mapping definition, see the *Setting Up a Trial with InForm Architect and MedML Guide*.

2 Load the mapping definition into the study database using the MedML Installer utility to process the mapping XML file. For information about using the MedML Installer utility, see the MedML online help, or *Using the MedML Installer utility* (on page 65).

3 Produce a fixed-length or character-separated flat file of the mapped data using the InForm Data Export utility.

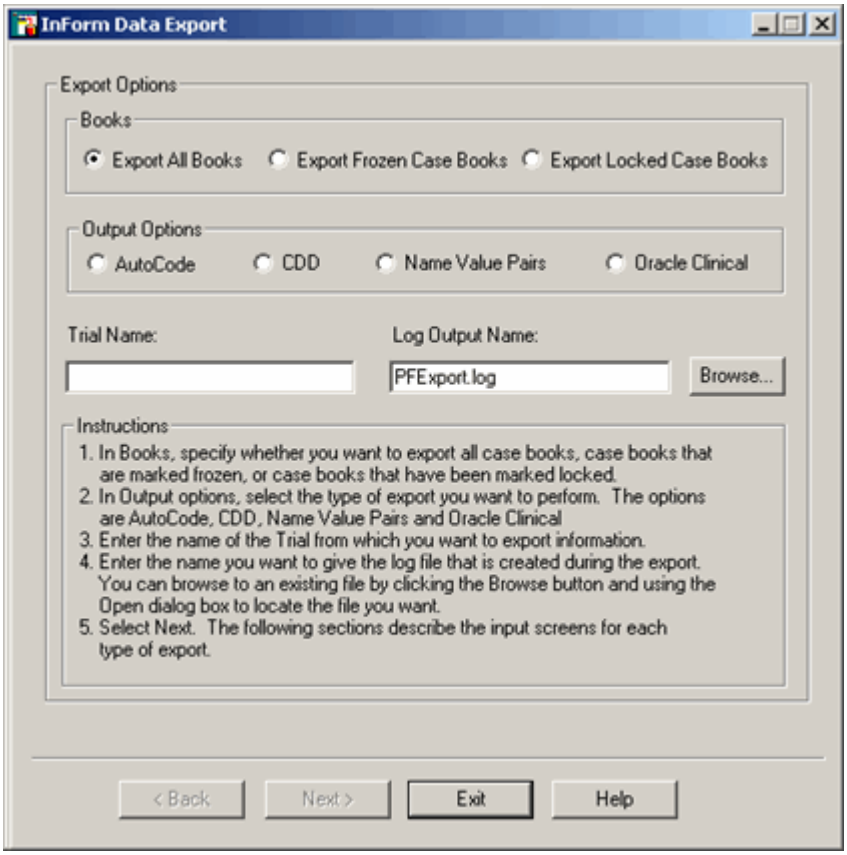
4 Import the output file into an Oracle Clinical database by using the Oracle Clinical Upload application.

Running the export in Oracle Clinical format

To run the Oracle Clinical data export:

1 Click **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm Data Export**.

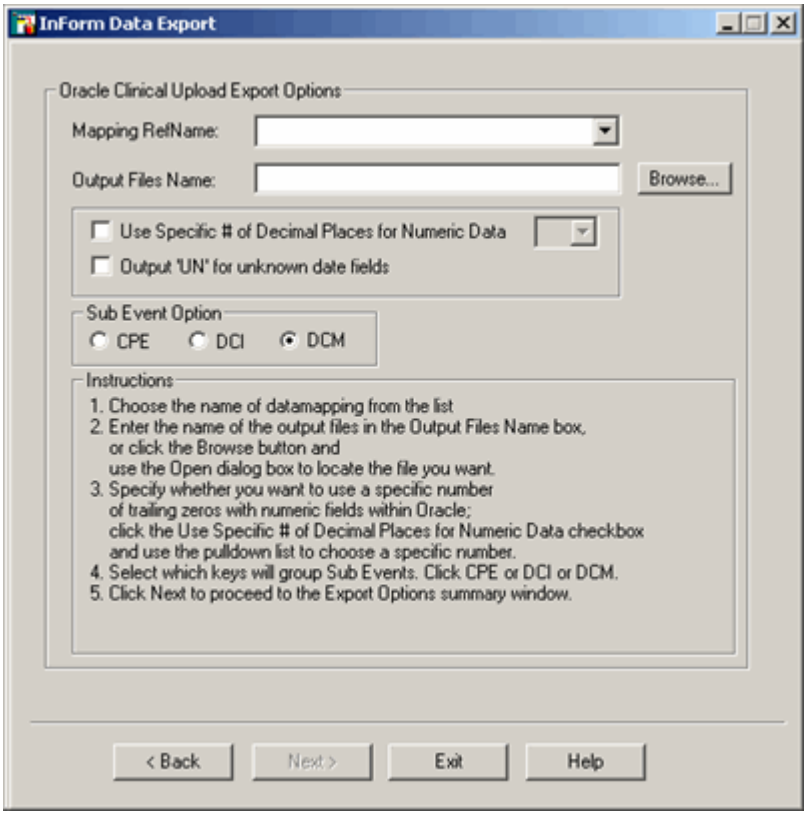
The InForm Data Export window appears.



- In the **Books** section, select which case books to export.
- In the **Output Options** field, select **Oracle Clinical**.
- Specify the **Trial Name** and the **Log Output Name**, or click **Browse** and locate the log output name.

2 Select **Next**.

The Oracle Clinical Upload Export Options window appears.

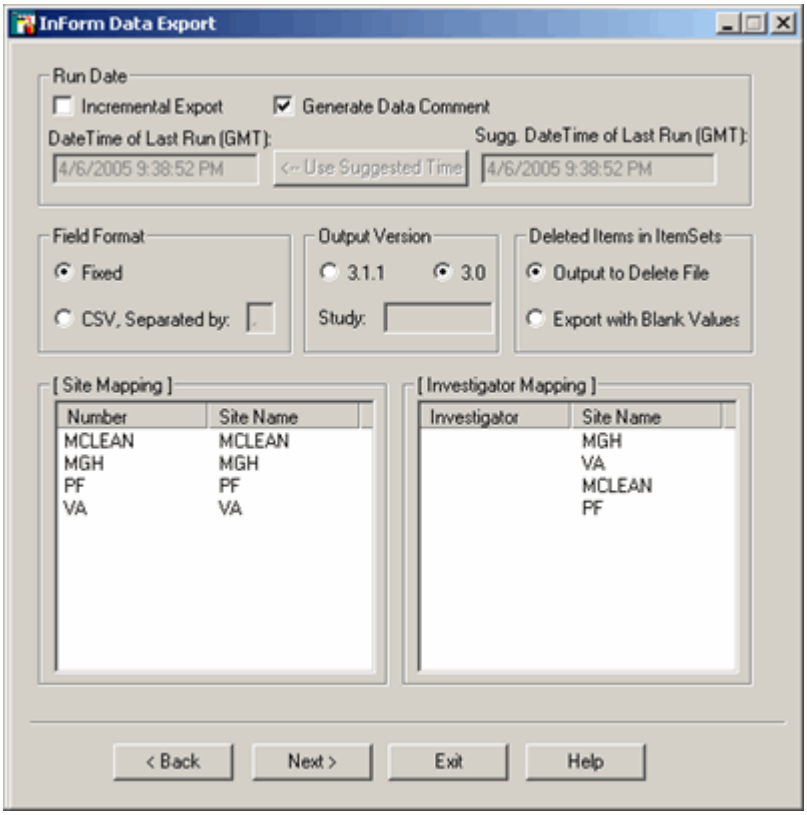


3 In the **Mapping RefName** drop-down list, select the RefName of the Oracle Clinical mapping definition that you created with the InForm Architect application.

- In the **Output Files Name** field, specify the name of an output file. The InForm Data Export adds the suffixes for each file type it generates to the name that you specify.
- Select the **Use Specific # of Decimal Places for Numeric Data** checkbox and use the drop-down list to specify the number of trailing zeros to use in numeric fields within the Oracle database.
- Select the **Output UN for unknown data fields** checkbox to flag unknown data fields in your export.
- Select a **SubEvent** option:
  - **CPE**—SubEvents are grouped by Clinically Planned Event.
  - **DCI**—SubEvents are grouped by Data Collection Instrument, a customer-defined grouping of Data Collection Modules.
  - **DCM**—SubEvents are grouped by Data Collection Module, a customer-defined grouping of data items.

4 Click **Next**.

The Run Date window appears.



5 In the **Run Date** section:

- Select **Incremental Export** to receive an export of only the data entered after a certain date/time.
  - If you selected **Incremental Export**, specify the DateTime after which any data entered is to be exported, or click **Use Suggested Time** to enter the time shown in the **Sugg. DateTime of Last Run (GMT)** box.
- Select **Generate Data Comment** to export comments as well as data.

The run date is also used to determine if a visit date, site number, patient number, or investigator number has changed since the last run date. If there has been a change, a delete record is produced for the appropriate item.

Note: By default, the values in the **DateTime of Last Run** and **Suggested DateTime of Last Run** fields are expressed in Greenwich Mean Time (GMT). If you enter a different date and time than the default or suggested values, you must be sure to use GMT.

6 In the **Field Format** section:

- Select **Fixed** to
- Select **Comma-separated value (CSV)** to

7 In the **Output Version** section, select 3.1.1 or 3.0 to specify which version of the Oracle Clinical Batch Upload tool to use.

Note: If you select version 3.1.1, you must type the name of the trial.

8 In the **Deleted Items in ItemSets** section:

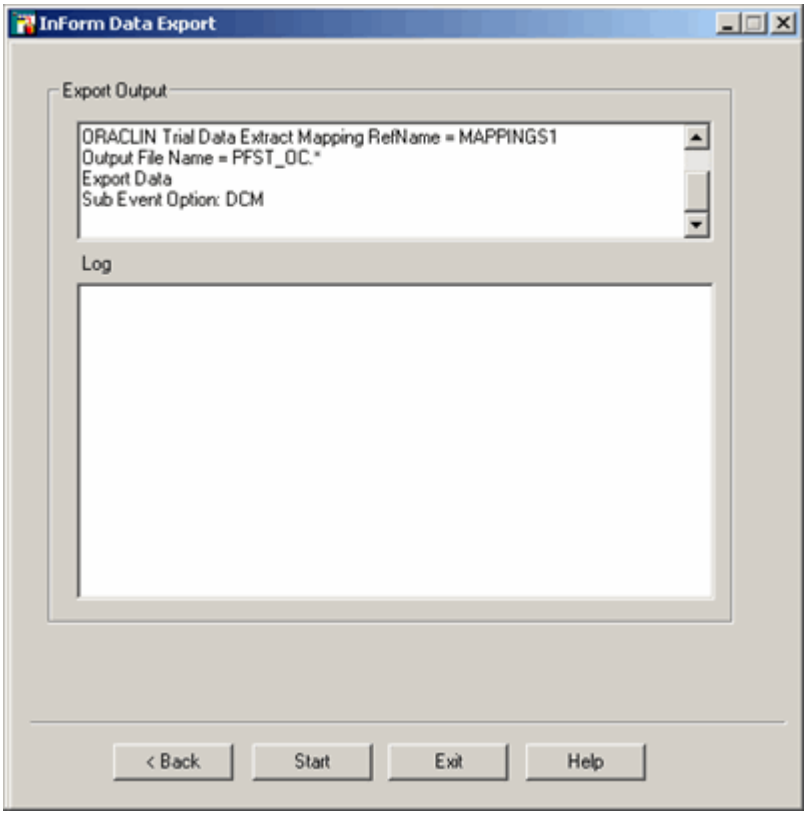
- Select **Output to Delete File** to output the deleted itemset rows to the DEL file and to resequence the remaining itemset rows.
- Select **Export with Blank Value t**o keep the deleted itemset rows in the export file with blank values.

9 In the **Site Mapping** section, map the site mnemonics and numbers by double-clicking the numbers to change them.

10 In the **Investigator Mapping** section, map the site mnemonics and investigators by double-clicking the investigators to change them.

11 Click **Next**.

The Export Output window appears with the data that you specified.



12 Click **Start** to begin the export.

The InForm Data Export begins to create the OA6 file and display progress messages in the log area of the window. These messages are logged in the LOG file whose name you specified in the Oracle Clinical Upload Export Options window.

13 When the export finishes, click **Exit**.

Running the InForm Data Export from the command line

To run the InForm Data Export from the command line, use the following parameters.

Note: Phase Forward strongly recommends that you run the InForm Data Export through the PFConsole utility. For more information, see ***Using the PFConsole utility*** (on page 7).

Parameter	Variable	Description
PFConsole utility		Starts the PFConsole utility.
PFExport		Starts the InForm Data Export.
-autorun		Runs the InForm Data Export in a command window. For more information, see *Using the PFConsole utility* (on page 7).
-?		Displays the usage statement.
-help		Displays the online help for the InForm Data Export.
-Trial	trial name	The name of the trial from which to export data.
-outfile	output log file name	Indicates that the next parameter is the name of an output file. Full pathname of an output file that contains the text of messages displayed by the InForm Data Export. Optional.
-cdd	ODBC DSN Name	Indicates that you are running the CDD Output tool. ODBC DSN name of the CDD to use as the export target of the CDD Output tool is required. If you specify a DSN name, you must also provide –user and –password parameters.
-user	ODBC user name	Indicates that the next parameter is a user name. User name of the user you set up for CDD access by the CDD Output tool; required if you specify a DSN name.
-password	ODBC password	Indicates that the next parameter is the user password. Password of the CDD user you set up for CDD access by the CDD Output tool is required if you specify a DSN name.
-CreatDSN	OraConnStr	For CDD export only, creates a new DSN within the CDD database.

Parameter	Variable	Description
-RefName	CDDRefName	For CDD export only, drops the current user and creates a new schema that is defined by the RefName.
-TBSP	OraTabSpace	For CDD export only, the name of the tablespace in which to put the new schema when creating a new DSN within the CDD database.
-crfhelp		Indicates that you want to export CRFHelp.
-nvfile	output nv file name	Indicates that you are running the InForm Data Export for Name Value pairs. Name of the export file that is created when you run the export utility for Name Value pairs is required if you specify the –nvfile flag.
-DelPrefix	string to prefix data marked as deleted	String that is added to a row of data in the export file to indicate that the data had been deleted from the InForm application prior to the export.
-Autocode		Indicates that you want to export for autocoding.
-REFNAME	targetset refname | ALL	One of the following: • The RefName of the predefined Autocode targetset. • ALL, to export all predefined Autocode targetsets.
-ACOUTFILE	output file name (*.xml)	Name of the XML to which you want to export.
-SEPARATE		Output each of the RefNames that are indicated above into separate files. If not specified, all RefNames are output to one file.
-NONCODEDONLY		Output only non-coded items.
-template	Oracle Clinical template file name	For export only to the Oracle Clinical database, required. Specify the path name of the template file to use.
-oraexport	DATA| SAVETEMPLATE| CREATETEMPLATE	For Oracle Clinical export only, required. Specify either to export data, save the template as an XML file, or create the template from an XML file.
-numberdecimalplaces	# of decimal places	For Oracle Clinical export only, optional. Specify the number of trailing decimal places, between 0 and 10.
-SubEventOption	CPE|DCI|DCM	Determines which keys will group SubEvents.

Parameter	Variable	Description
-OutputUN		Outputs UN for unknown date fields, if present.
-sysadmin		Name of the System Administrator with rights to export CRF Help. Indicates that the next parameter is the System Administrator ID. This ID is necessary to export CRFs.
-outputtype	ALL|EPIC|STANDARD	Indicates the type of export.
-customheader	text of header	For PDF format only, optional. Allows you to indicate the name of a custom header.
-template	template path name	For export only to the Oracle Clinical database, required. Specify the path name of the template file to use.
-oraexport	DATA|SAVETEMPLATE|CREATETEMPLATE	For export only to the Oracle Clinical database, required. Specify either to export data, save the template as an XML file, or create the template from an XML file.
-numberdecimalplaces	# of decimal places	For export only to the Oracle Clinical database, optional. Specify the number of trailing decimal places, between 0 and 10.
-RefName	defined CDD RefName	For CDD Export only, drops the current user and creates a new schema defined by the RefName.
-CreateDSN	Oracle connection string	For CDD Export only, creates a new DSN within the CDD database.
-TBSP	Oracle tablespace	For CDD Export only, the name of the tablespace in which to put the new schema.

Example

The following is a sample syntax for running the InForm Data Export from the command line:

```
PFConsole
PFExport [-autorun] [-?] [-help] [-Trial trial name] [-outfile outfile name]
[-cdd ODBC DSN Name] [-user ODBC username] [-password ODBC password]
[-CreatDSN OraConnStr][-RefName CDDRefName][-TBSP OraTabSpace]
[-crfhelp]
[-nvfile nv file name [-DelPrefix string to prefix data marked as deleted]]
[-Autocode [-REFNAME targetset refname|ALL] -ACOUTFILE output file name
(*.xml)[-SEPARATE] [-NONCODEDONLY]]
[-Template Oracle Clinical Template file name
[-REFNAME targetset refname]
[-Ora Export DATA|SAVETEMPLATE|CREATETEMPLATE]
[-NumberDecimalPlaces # of decimal places]
[-SubEventOption CPE|DCI|DCM]
[-OutputUN]]
```

Oracle Clinical fields

The following are the Oracle Clinical fields for Oracle Clinical data exports:

Oracle Clinical field	Description
Investigator	Investigator (FINO) number—Customer-defined number to uniquely identify an investigator.
Site	Site number—Customer-defined character value (mostly numbers, may have leading zeros) to uniquely identify the site.
Patient number	Patient identification—A numeric value that is assigned to a patient by the investigator. A patient number is unique across a protocol.
Document number	Customer-defined number to identify the document.
Clinical Planned Event (CPE) name	Customer-defined visit name—A character value that describes in detail the name of the visit as specified in the protocol timetable.
Subevent Number	A unique value that is assigned to a particular event (patient, visit and DCM) when the result date or time information does not match the previous date or time for the given event.
Visit Date	Date of visit—Format YYYYMMDD.
Visit Time	Time of visit—Format HHMMSS. (Depends on protocol. Leave blank if not used.) For lab data, if a result has time associated with the DCI Name, you must use LAB TIME.
DCI Name	Data collection instrument (DCI) Name—Customer-defined name to identify a grouping of data collection modules (DCMs). For example, lab data DCMs HEMATOLOGY_01 and CHEMISTRY_01 might be assigned to the DCI Name LAB.
DCM Name	Data collection module name—Customer-defined name to identify the grouping of data. For example, Red Blood Cell count, Hemoglobin, and Hematocrit might be assigned to the HEMATOLOGY_01 DCM.
DCM Subset Name	Customer-defined name to specify how data is divided into smaller subsets by data type.
Question Group Name	Customer-defined name to specify how different results are grouped together.
Question Name	Data point identifier (for example, lab test performed) that is translated into a standard variable name as provided by the customer.

Oracle Clinical field	Description
Occurrence Number	Occurrence number that identifies the test, and that is used to associate repeating data, such as repeated labs. Default value should be 0 (zero). A change in occurrence number would be used to indicate a retest value.
Repeating Sequence Number	Increment for each result starting at 1 (one). Can also be used in conjunction with the Occurrence number. (A common repeat sequence number indicates a relationship between values.)
Question Value	Answer to the question, or the test result. Can contain numeric or text values as needed.
Comment	Data comment text (optional); also used to indicate record deletions.
Qualifying Question Value	Customer-defined value.
Study	Study name that is in the form CI_PROT (for example, 1008_32); no leading zeros on CI or PROT.

Format of output files

Format of OA6 file

This section of an OA6 file shows the fields that are exported according to the table in *Oracle Clinical fields* (on page 101). Each field below corresponds to a field that is listed, in order of presentation. The file is fixed format.

```
0001      001       1111                              V1                  0
20000617      DEMOGRAPHICS/DRUG SENSITIVITIEDEMOGRAPHICS 01 DEMOG   PATIENT
INFORMATION 01        PTINTL              0  1  AAA
0001      001       1111                              V1                  0
20000617      ELIGIBILITY                   ELIGIBILITY 01  ELIGIB
ELIGIBILITY 01              ELIG                    0  1  Yes
0001      001       1111                              V3/RANDOMIZATION    0
20000404      SUBJECT STATUS-SCREENING      PATIENT STAT 02 STATUS1 PATIENT
STATUS 02             PHASE               0  1  Screening
0001      001       1111                              V3/RANDOMIZATION    0
20000404      SUBJECT STATUS-SCREENING      PATIENT STAT 02 STATUS1 PATIENT
STATUS 02             STATUS              0  1  Completed
```

Format of DEL file

This section of a DEL file shows the records for an investigator key that was changed from 6666 to 5555. This delete file was exported with comma-separated values turned on, meaning that a delete record is generated for every Visit/Visit Date for this investigator's patients.

```
6666,002,1111,,V10,0,20000802,,,,,,,0,1,Investigator key has changed from
'6666' to '5555',DELETE,
6666,002,1111,,V5,0,20011108,,,,,,,0,1,Investigator key has changed from
'6666' to '5555',DELETE,
```

CHAPTER 7

Using the InForm Performance Monitor utility

In this chapter

Overview 104

Starting the InForm Performance Monitor utility 105

Capturing performance statistics 106

Performance Monitor output options 109

Managing the InForm Performance Monitor data 110

Examples of using the InForm Performance Monitor utility 111

Overview

The InForm Performance Monitor utility runs on the desktop where an InForm application server is running. When a session of the InForm application is active, the InForm Performance Monitor utility listens for and captures InForm application server messages about specific types of InForm activities, and displays the messages in a window where you can arrange and save them. You can use this utility to:

- Provide information that helps with performance tuning during trial development and implementation.
- Capture performance data for troubleshooting.

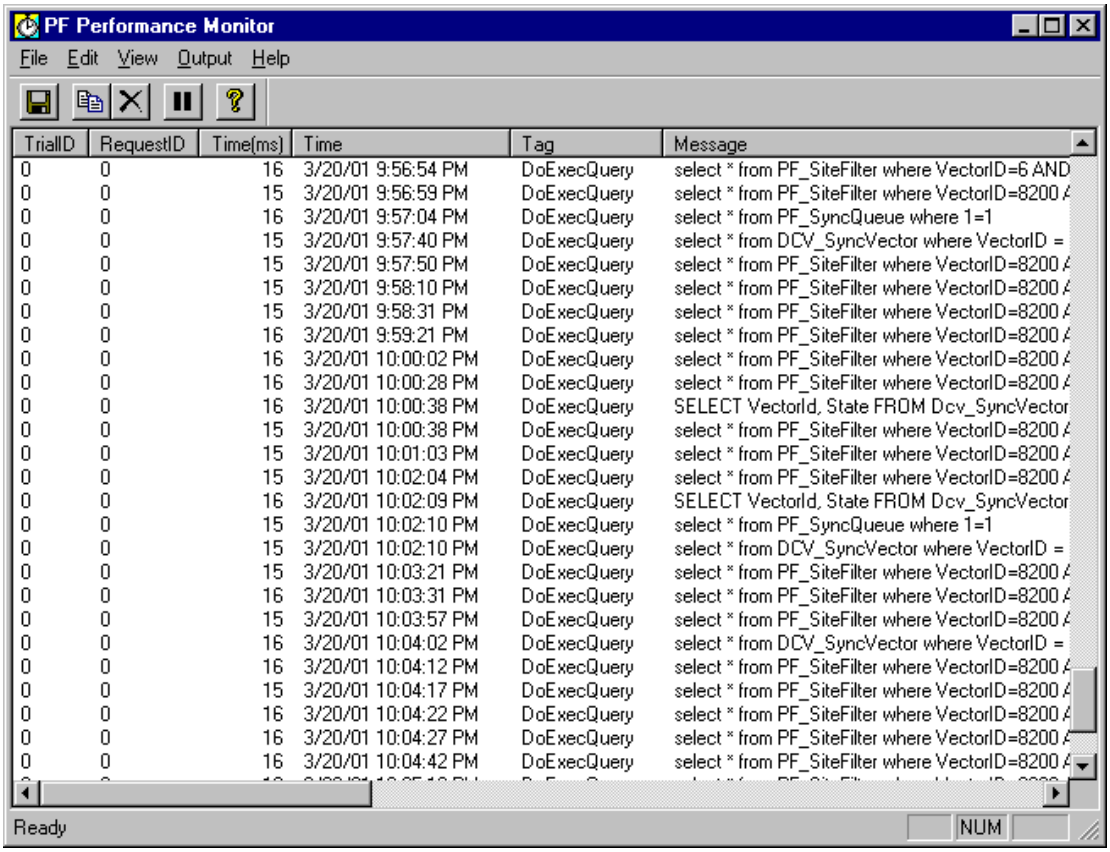
Note: The InForm Performance Monitor utility is not intended for extended use in a production environment because it can impact server performance over time. Phase Forward suggests running the InForm Performance Monitor utility at 5-10 minute intervals.

Starting the InForm Performance Monitor utility

To start the InForm Performance Monitor utility:

1 Select **Start** > **Programs** > **Phase Forward** > **InForm 4.6** > **InForm Performance Monitor**.

The Performance Monitor window opens. If one or more InForm servers on the machine are running, the InForm Performance Monitor utility begins recording InForm application server messages.

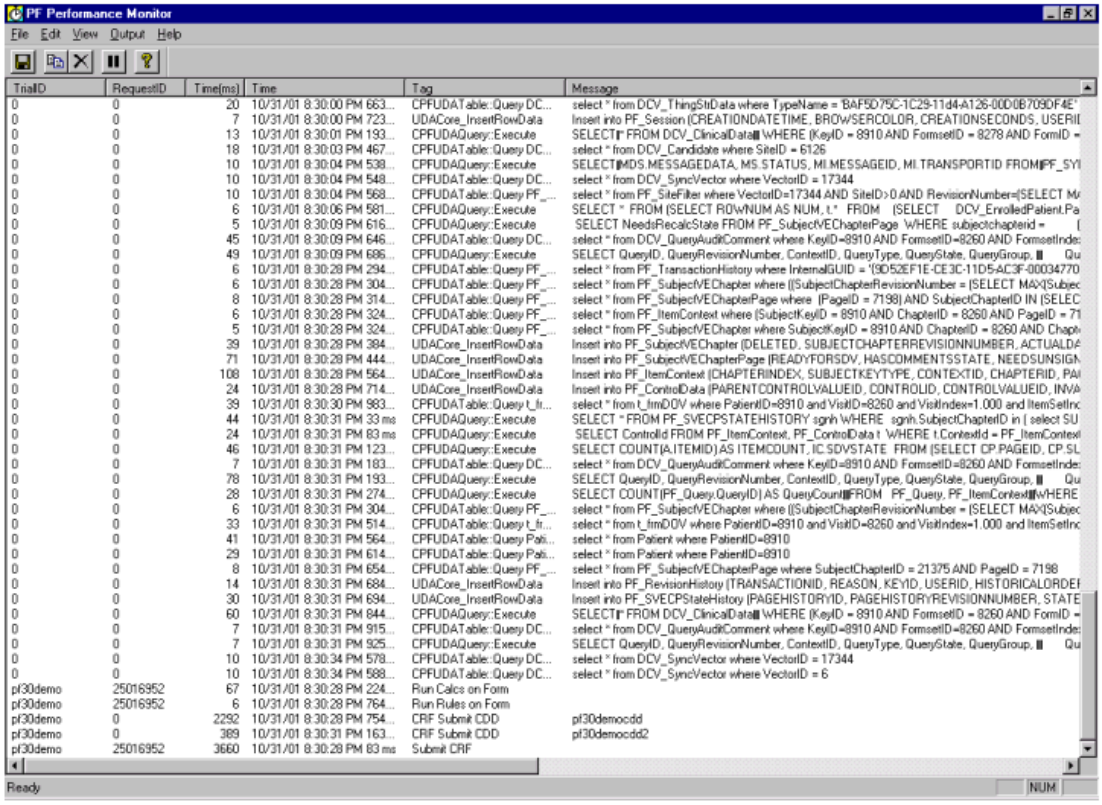


Note: The InForm Performance Monitor utility is used by Phase Forward staff to assess and enhance the performance of a trial, and evaluate server activities by quantifying the amount of time each activity requires. Although a brief description of its messages is detailed in ***Capturing performance statistics*** (on page 106), the use and interpretation of the InForm Performance Monitor utility is at the sole discretion of Phase Forward staff.

Capturing performance statistics

For each activity in a session of the InForm application, the InForm Performance Monitor utility captures the following data and displays it in the Performance Monitor window:

- **Trial ID**—Specifies the trial that is associated with an InForm Performance Monitor utility message. If a message is not trial-specific, it contains a value of 0.
- **Request ID**—Contains a numeric value. The InForm Performance Monitor utility messages with the same, non-zero RequestID value are associated with some common activity. For example, in the figure below, the messages that have a RequestID value of 25016952 are all associated with submitting a CRF (last line of figure).



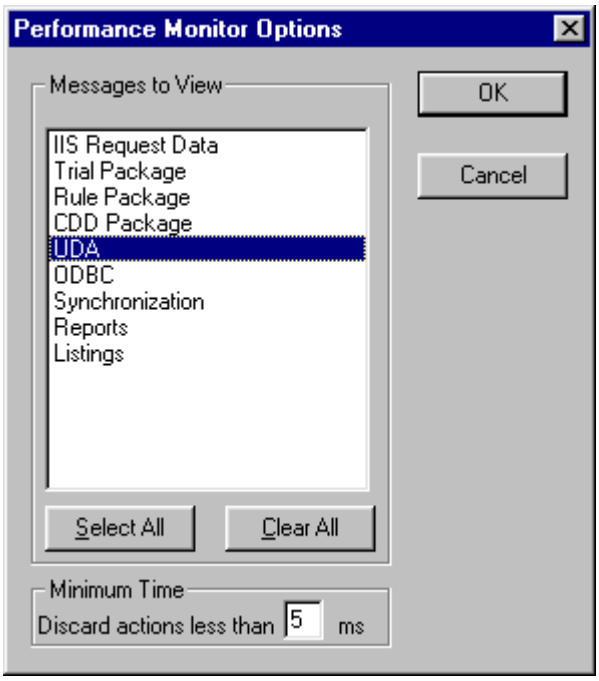
- **Time (ms)**—Elapsed time in milliseconds from the user-issued request to the server response.
- **Time of transaction**—The time specified is GMT.
- **Tag**—Text string that describes the activity that is associated with the message.
- **Message**—Text string that provides detailed information that relates to the activity.

Viewing messages from specific subsystems

To choose message filters:

1 Select **View** > **Options**.

The Performance Monitor Options dialog appears.



2 In the **Messages to View** section, select one of the following filters to receive messages from only the specified subsystems:

Filter	Subsystem
IIS Request Data	The InForm ISAPI.
Trial package	The InForm trial MTS package, by InForm application server.
Rule package	The InForm rule package, by InForm application server.
CDD package	The InForm CDD package, by InForm application server.
UDA	The InForm database.
ODBC	The InForm ODBC.
Synchronization	The InForm Unplugged application.
Reports	The InForm report.
Listing	The InForm listing.
InForm service	The InForm service.
PFImport	The InForm Data Import.

To select multiple subsystems, hold down the **Ctrl** key while you select.

3 In the **Discard actions less than** field in the **Minimum Time** section, specify the threshold above which to record actions (in milliseconds).

4 Click **OK**.

Selecting InForm servers

To select an InForm application server from which to capture information:

1 Select **View** > **Servers**.

The InForm Server Selection List dialog appears.

2 Select the server from which to capture messages. To select multiple servers, hold down the **Ctrl** key while you select.

3 Click **OK**.

Performance Monitor output options

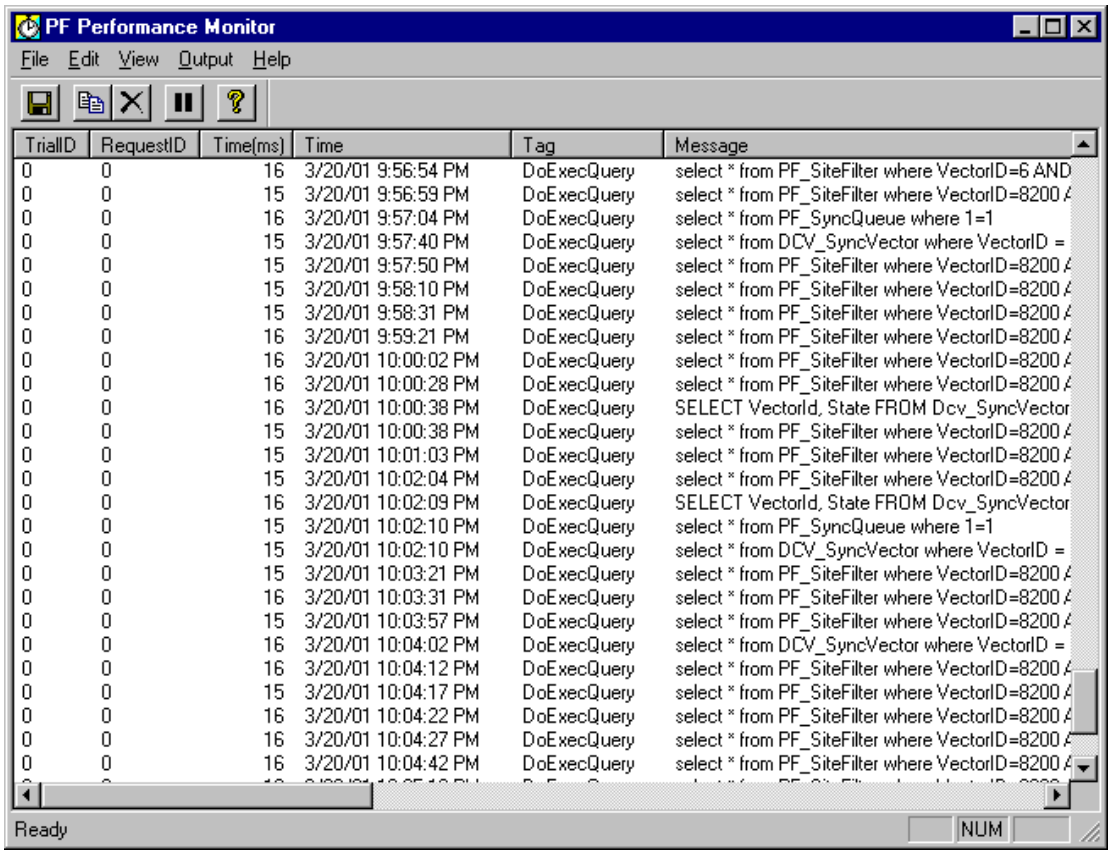
Note: Before you select an output option, select the statistics to capture as described in *Capturing performance statistics* (on page 106).

Select any of the following output options for the InForm Performance Monitor utility messages:

Output option	Description	Action
View output online	Displays messages in the Performance Monitor window. (Default.)	Select **Output** > **Output to Window**
Stream output to a file	Sends messages to a file instead of (or in addition to) displaying them in the Performance Monitor window.	Select **Output** > **Stream to File**
Save a performance log	Saves the performance log as comma-separated text to a specified file.	Select **File** > **Save As**

Managing the InForm Performance Monitor data

To change the display in the Performance Monitor window:



Task	Action
Sort a column	Click the column heading bar.
Select a single message	Click the message.
Select multiple contiguous messages	Hold down the **Shift** key while clicking the messages.
Select multiple noncontiguous messages	Hold down the **Ctrl** key while clicking the messages.
Select all messages	Select **Edit** > **Select All**.
Copy selected messages	Select **Edit** > **Copy**, or click the **Copy** button in the toolbar.
Delete selected messages	Select **Edit** > **Delete**, or click the **Delete** button in the toolbar.
Clear all messages	Select **Edit** > **Reset**.

Examples of using the InForm Performance Monitor utility

Testing rule script efficiency

After you develop a set of rules for a trial and add patient data to your test database, you can run the InForm Performance Monitor utility to:

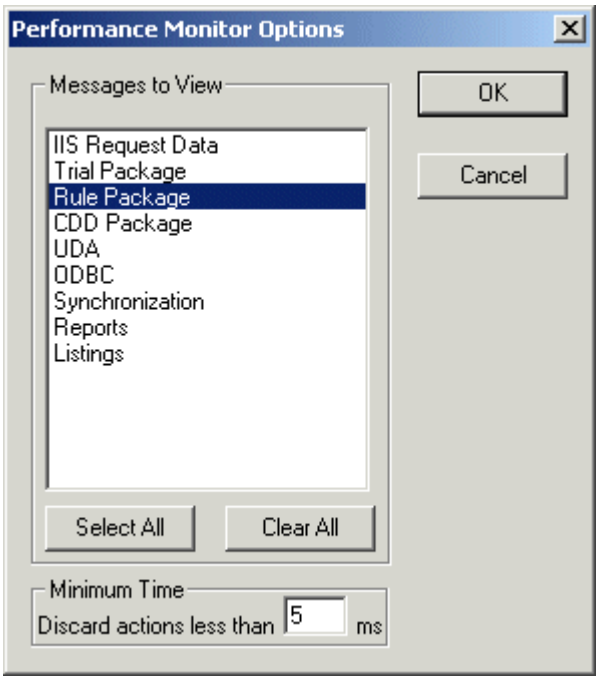
- Verify that rules are firing when expected, and that they are running against the expected rule contexts.
- Check for unusually long rule execution times.
- Determine how much of a transaction submit time is occupied by rule processing.

Capturing rule processing data

To capture rule processing data:

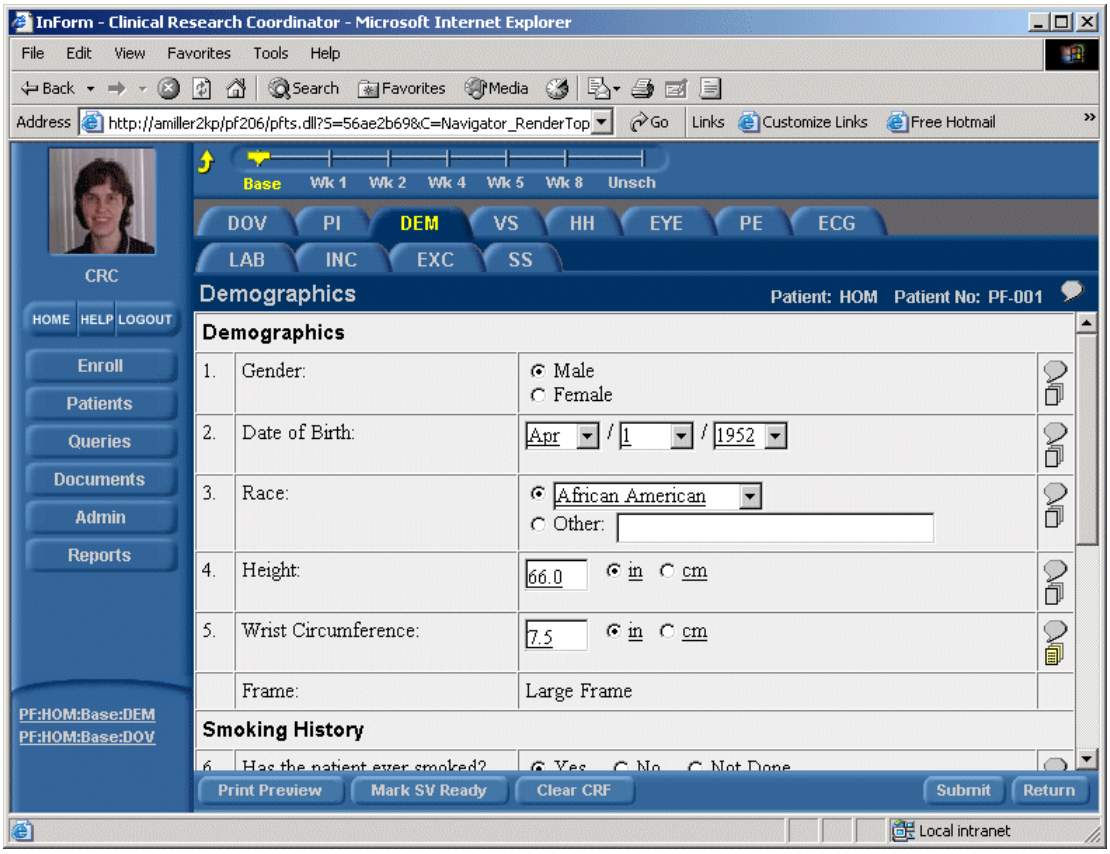
1 In the Performance Monitor window, select **View** > **Options**.

The Performance Monitor Options dialog appears.



2 Clear all filters except **Rule Package** to set the message filter to capture rule data.

3 Start the InForm trial and display the DEM form in the Baseline visit for one of the test patients.

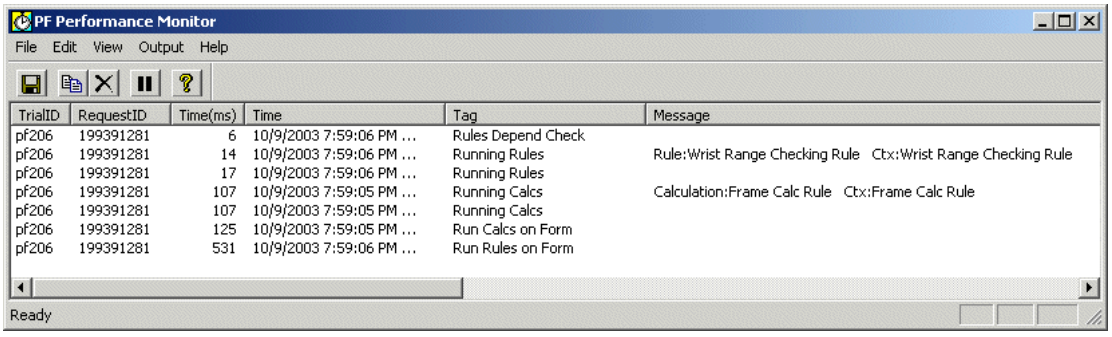


4 Type or edit the value for the **Wrist Circumferenc**e item, but do not click **Submit**.

5 In the Performance Monitor window, select **Edit** > **Reset** to clear the display.

6 In the InForm application, click **Submit**.

7 Return to the Performance Monitor window and check the messages.



The messages indicate that:

- The **Wrist Range Checking** and the **Frame Calc** rules fired as expected.
- The contexts for the rules were present.

The InForm Performance Monitor utility records the times that are required to check dependencies and to run the rules and calculations, as well as the total time to process rules

on the form.

Note: To sort the messages by request times, click the Time (ms) column header. When you are gathering data on multiple rules, sorting by time can highlight rules that require unusually large amounts of time to process.

Reviewing SQL query performance

If response times have degraded and your trial database accumulates data, you can use the InForm Performance Monitor utility to isolate long-running SQL queries. If you find long-running SQL queries, perform any of the following:

- Distribute tablespaces differently.
- Improve indexing on certain database tables.
- Run update statistics on the database.

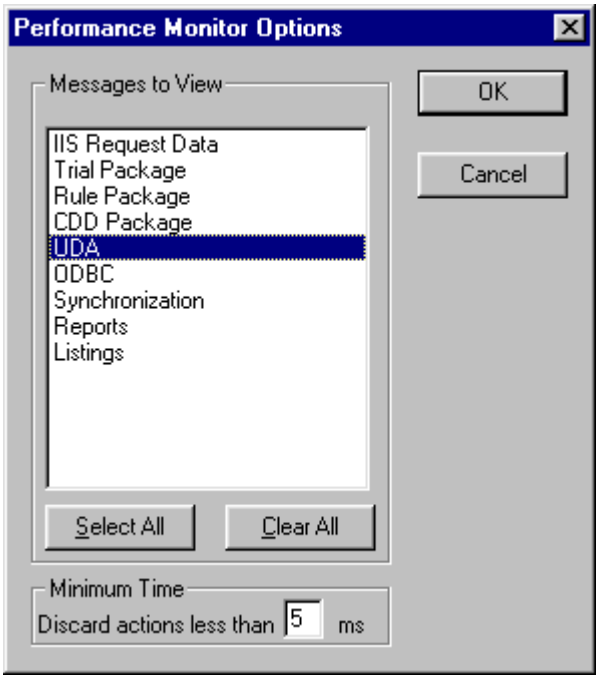
Note: The InForm Performance Monitor utility is not intended for extended use in a production environment because it can impact server performance over time. Phase Forward suggests running the InForm Performance Monitor utility at 5-10 minute intervals.

Capturing SQL query performance data

To capture SQL query performance data:

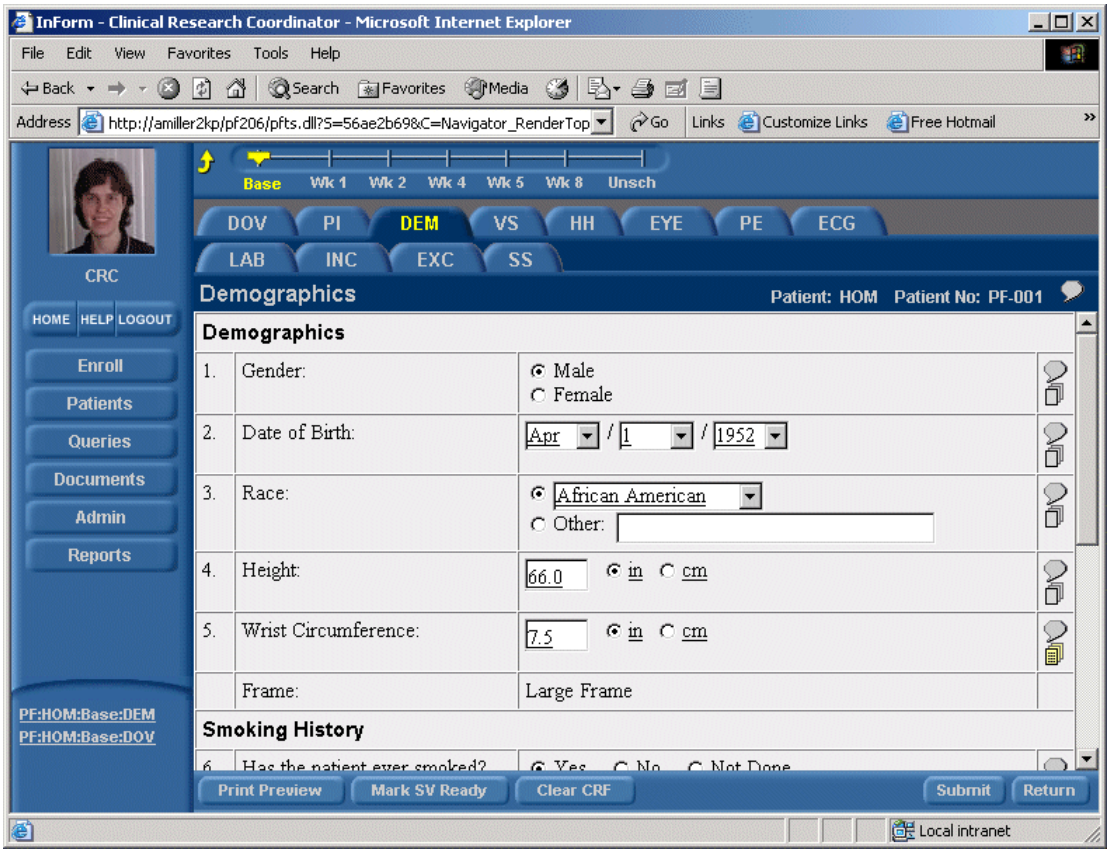
1 From the Performance Monitor window, select **View** > **Options**,

The Performance Monitor Options dialog appears.



2 Clear all filters except **UDA** to set the message filter to capture rule data.

3 Start the InForm trial and display the DEM form in the Baseline visit for one of the test patients.



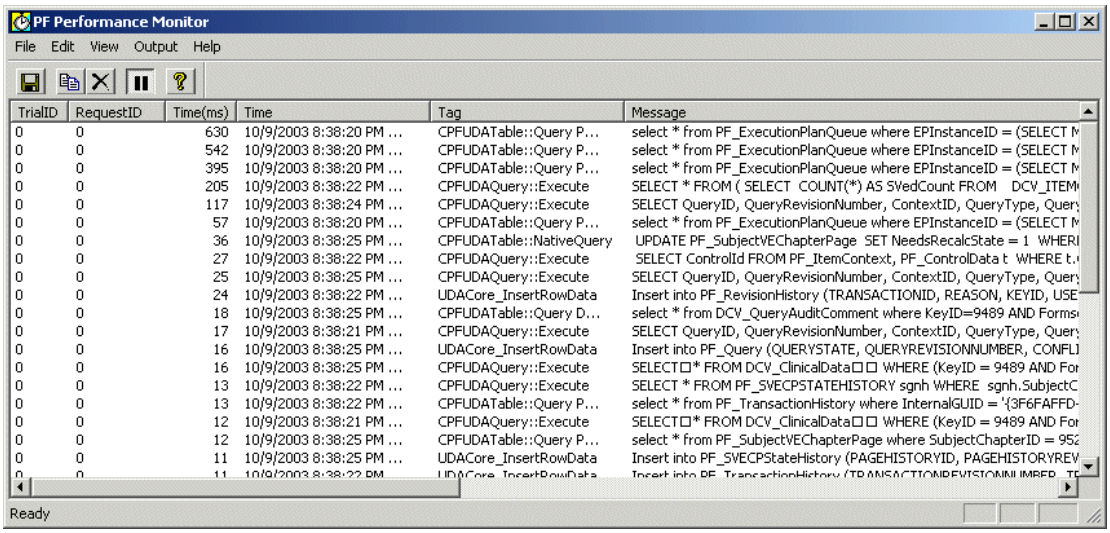
4 Type or edit the value for the **Wrist Circumferenc**e item, but do not click **Submit**.

5 In the Performance Monitor window, select **Edit** > **Reset** to clear the display.

6 Select **Output** > **Stream to File** to instruct the InForm Performance Monitor utility to stream the messages to a file, and specify the file name in which to store the messages.

7 In the InForm application, click **Submit**.

8 Return to the Performance Monitor window and check the messages.



If requested to do so by Phase Forward support, send them the saved message file.

Note: To filter the stream of messages by the time required to execute the request, use the **Minimum Time** section of the Performance Monitor Options dialog. The InForm Performance Monitor utility displays only messages for requests that require more than the specified number of milliseconds to execute.

To determine a normal request time, enter 0 in the **Discard actions less than** field in the **Minimum Time** section, then compare the data in the **Time** field on the Performance Monitor window for each captured message.

CHAPTER 8

Using the InForm Report Folder Maintenance utility

In this chapter

Overview 118

Folder structure for multiple trials or sponsors 119

Setting up a folder structure for multiple trials or sponsors 124

Copying report folders 127

Overview

The InForm Report Folder Maintenance utility is a Windows application that can do the following:

- Copy reporting folders and their contents to target folders.
- Define and create folder structures when using a single reporting server for
  - multiple trials
  - multiple sponsors, or
  - multiple trials within multiple sponsors.
- Automatically update any links to drill-down reports in InForm standard reports that were copied to a target location.
- Allow you to associate a copied report with a new reporting package at the target location.

Only users with System Administrator privileges can run the InForm Report Folder Maintenance utility.

Note: This includes saved reports as well as standard report definitions.

You install the InForm Report Folder Maintenance utility on the reporting (Cognos 8 Business Intelligence) server as part of the Reporting and Analysis installation and configuration. You can find it at this path:

```
\c8\bin\PFMTRSetupUtil.exe
```

The InForm Report Folder Maintenance utility copies only reports and report definitions. It does not copy the folders that contain links to legacy ASP reports, the InForm Trial Management package, or any published trial-specific clinical package. For more information about the association between reports and report packages, see *Report package association* (on page 125).

Note: You must complete the instructions in the *Configuring a Trial for Reporting* section in the *Installation and Configuration Guide* before performing the procedures in this chapter.

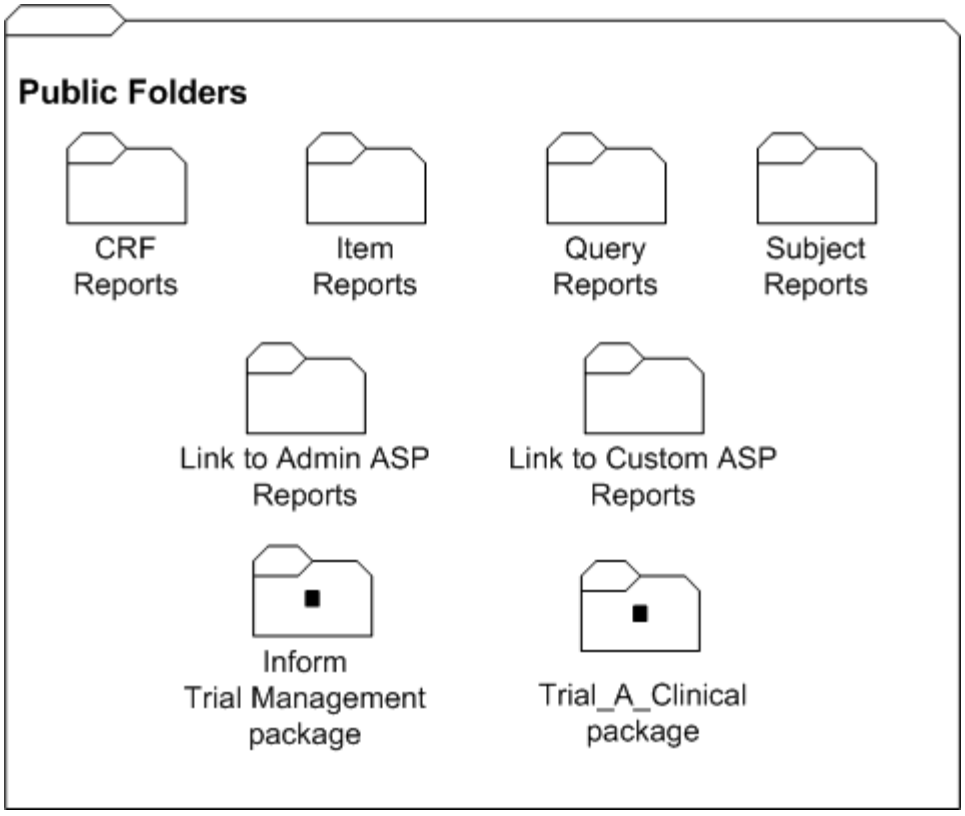
Folder structure for multiple trials or sponsors

Setting up the initial folder structure

To set up the initial folder structure:

- Import the standard reports archive. This archive includes all standard reports, as well as the InForm Trial Management package.
- Publish a trial-specific clinical package for clinical reports.

After you complete these steps, the InForm Report Folder Maintenance utility creates folders on the reporting server for the standard reports, legacy ASP reports, and the reporting packages. These folders appear in the **Public Folders** tab of the Reporting and Analysis portal. The following illustration shows the folders that appear after the initial setup of a single trial.



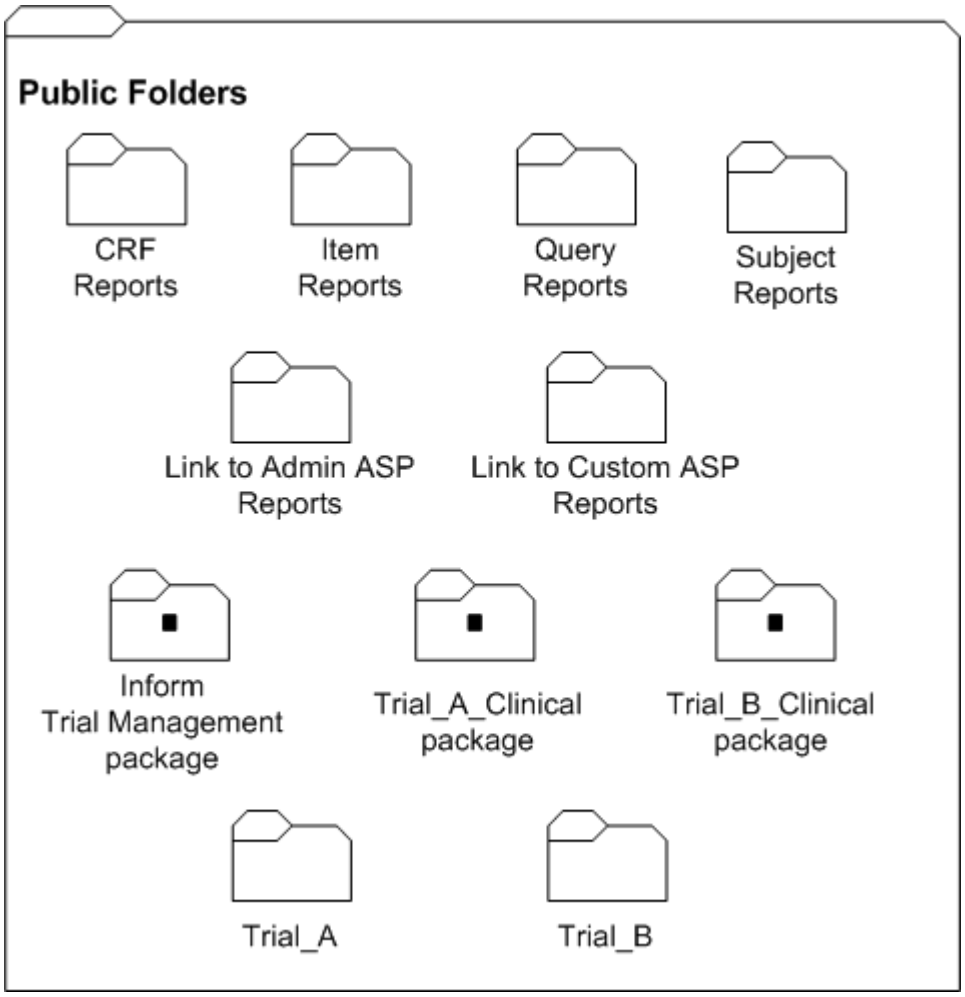
Folder structure for multiple trials

To set up several trials for a single sponsor, consider creating a separate folder for each trial. PFRInit does this automatically.

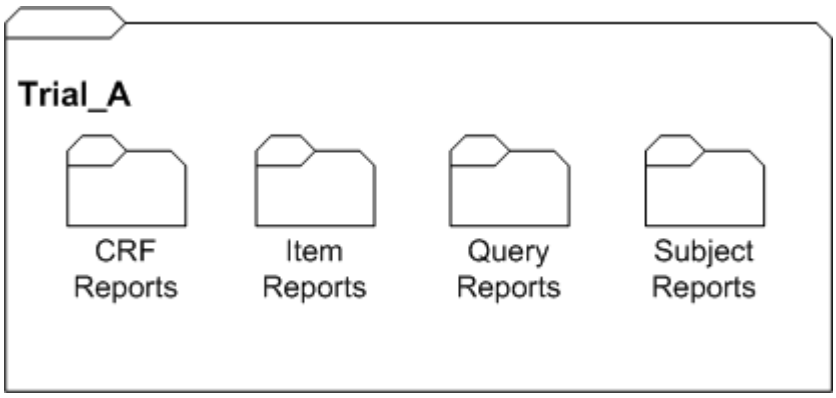
The following illustration shows how the **Public Folders** might look in the Reporting and Analysis portal with folders for two trials–Trial A and Trial B.

Note that this configuration uses three different packages:

- InForm Trial Management package—Shared by both Trial_A and Trial_B.
- **Trial_A_Clinical package**—Used only for Trial_A.
- **Trial_B_Clinical package**—Used only for Trial_B.



The contents of each trial-specific folder are set up to include the default reporting folder structure. The following illustration shows the contents of the Trial_A folder.



This structure ensures that you can save trial-specific properties, such as prompt values and report schedules, for a specified report.

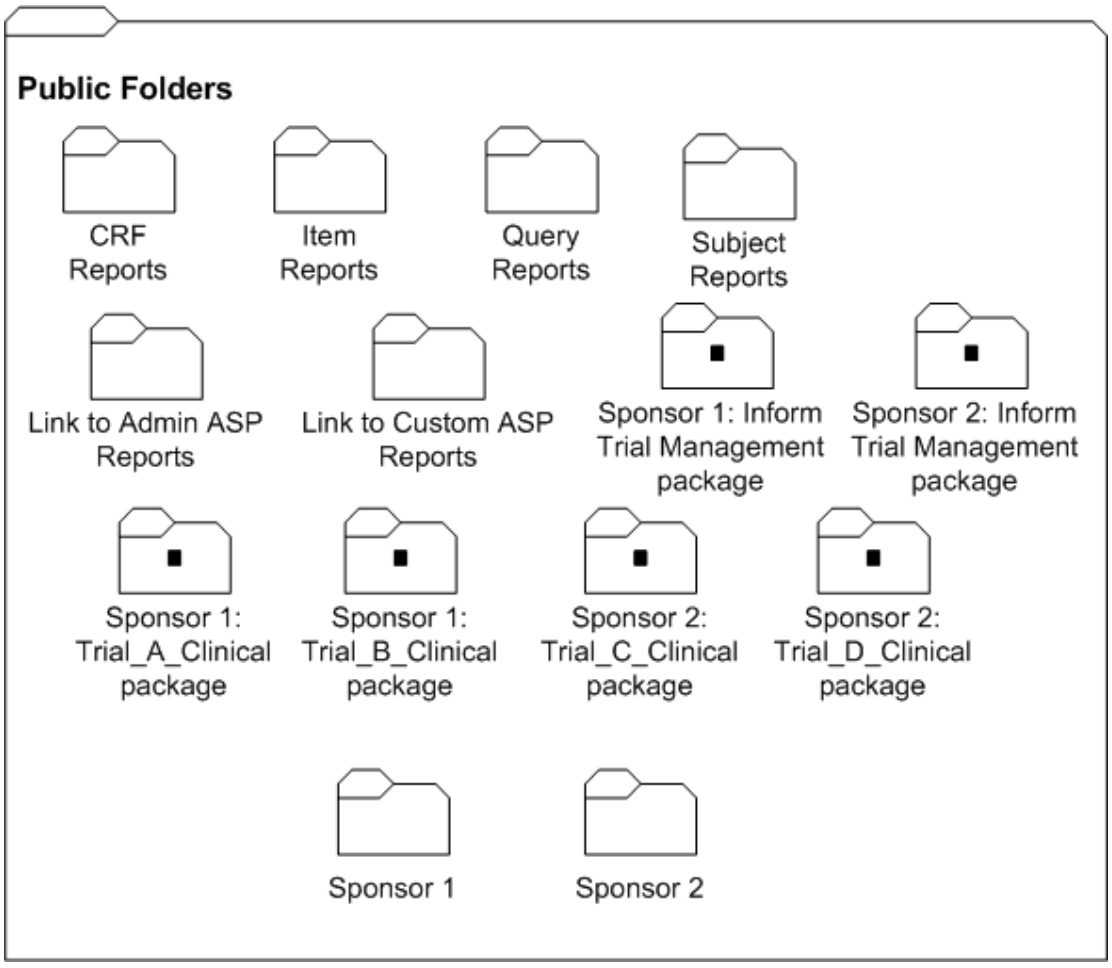
Folder structure for multiple sponsors, multiple trials

To set up folders for several sponsors on one reporting server, consider creating subfolders for each sponsor under the **Public Folders** tab. You can only do this if you use InForm Report Folder Maintenance utility to perform all the configuration steps.

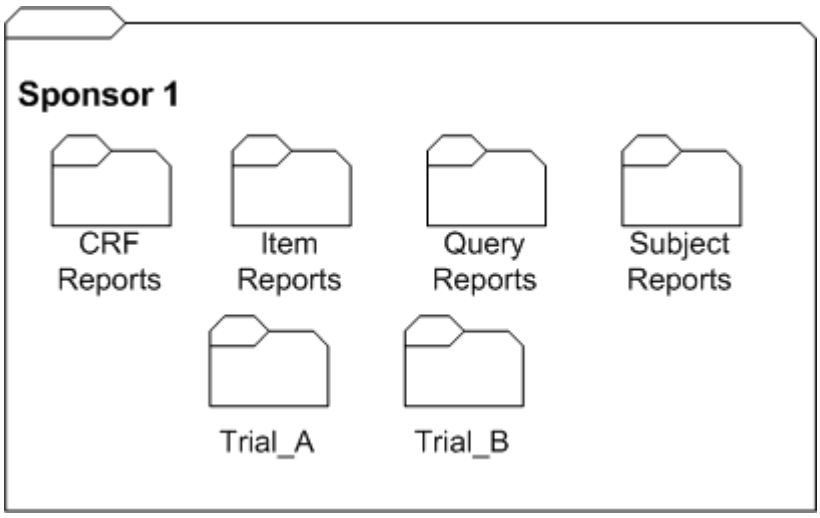
The following illustration shows how the **Public Folders** might look in the Reporting and Analysis portal with folders for two different sponsors, each hosting two different trials.

The following example shows:

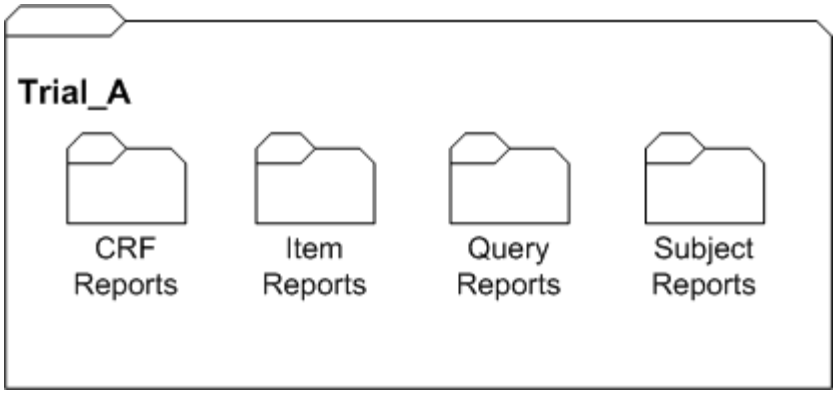
- Four clinical packages: two for Sponsor 1 and two for Sponsor 2.
- Two InForm Trial Management packages: one for Sponsor 1 and one for Sponsor 2.



The contents of each sponsor-specific folder are set up to include the default reporting structure for a single trial. The following illustration shows the contents of the Sponsor 1 folder:



The Trial_A folder contains all of the default standard report folders.



Setting up a folder structure for multiple trials or sponsors

Setting up reporting packages

All reporting packages must reside at the **Public Folders** level only; you cannot add a package to a subfolder. Therefore, if you are using several packages for different sponsors and trials, folders for each of these packages will appear in the **Public Folders** tab of the Reporting and Analysis portal.

Trial-specific clinical package

Each trial uses a unique trial-specific clinical package. For details on how to publish a trial-specific clinical package, see the *Installation and Configuration Guide*.

InForm Trial Management package

Trials that share a sponsor can share an InForm Trial Management package. This way, any revisions or updates to the package automatically apply to all trials for the sponsor.

You import the InForm Trial Management package by importing the Op model and Reports deployment archive. When you imported this archive as part of your installation, you imported both the standard reports folders and the InForm Trial Management package.

However, to be able to apply revisions to packages for individual trials, import a trial management package for each trial.

To import InForm Trial Management packages for different sponsors or trials:

1 Log on to the trial as the InForm system administrator.

2 Click **Reports**.

3 Select **Tools** > **Deployment**.

4 Click the **Import** tab.

5 In the **Op package and reports** entry, click **Set properties**.

6 Click the **Deployment** tab.

7 Click the **InForm Trial Management** checkbox. Make sure that all other options are deselected.

8 Click the pencil icon next to **InForm Trial Management**.

9 Change the name of the InForm Trial Management package to reflect the sponsor or trial that will be using the package.

10 Click **OK**.

11 Click **Import Now**.

12 Click **Finish**.

The Import screen appears.

Creating new folders for multiple trials or sponsors

Report package association

Each report is associated with the package (either the InForm Trial Management package or a trial-specific package) that was used to create it.

When you copy reports using the InForm Report Folder Maintenance utility, you can specify the package name to be used, if necessary. A report that you have copied to a new location must be associated with the package that will be used by the trial or sponsor who works with the report.

For reports that you created with the InForm Trial Management package:

- **Multiple trials, single sponsor**—You are not required to specify a new package name; the InForm Trial Management package is installed with every trial.
- **Multiple trials, multiple sponsors**—Each sponsor should provide an instance of the InForm Trial Management package; therefore, you may need to specify a new package name for the copied reports in the target location.

For reports that you created with the trial-specific clinical package, you must always specify a new package name for the copied reports in the target location.

Report validation

The InForm Report Folder Maintenance utility validates copied reports against the associated packages when you copy the reports. Therefore, the packages must exist before you begin the copy operation. For more information, see *Setting up reporting packages* (on page 124).

The InForm Report Folder Maintenance utility stops whenever it encounters any error. You must correct the error, delete all objects within the target folder, and run the tool again to copy all reports. For example, although you can copy report definitions that contain clinical report elements from one trial to another, the schema of the clinical portion of the reporting database is unique to each trial; therefore, the clinical package that is generated for each trial is unique. If a clinical report contains a report element that does not exist in the target package, the report cannot be validated after it is copied.

Report validation

The InForm Report Folder Maintenance utility validates copied reports against the associated packages when you copy the reports. Therefore, the packages must exist before you begin the copy operation. For more information, see *Setting up reporting packages* (on page 124).

The InForm Report Folder Maintenance utility stops whenever it encounters any error. You must correct the error, delete all objects within the target folder, and run the utility again to copy all reports. For example, although you can copy report definitions that contain clinical report elements from one trial to another, the schema of the clinical portion of the reporting database is unique to each trial; therefore, the clinical package that is generated for each trial is unique. If a clinical report contains a report element that does not exist in the target package, the report cannot be validated after it is copied.

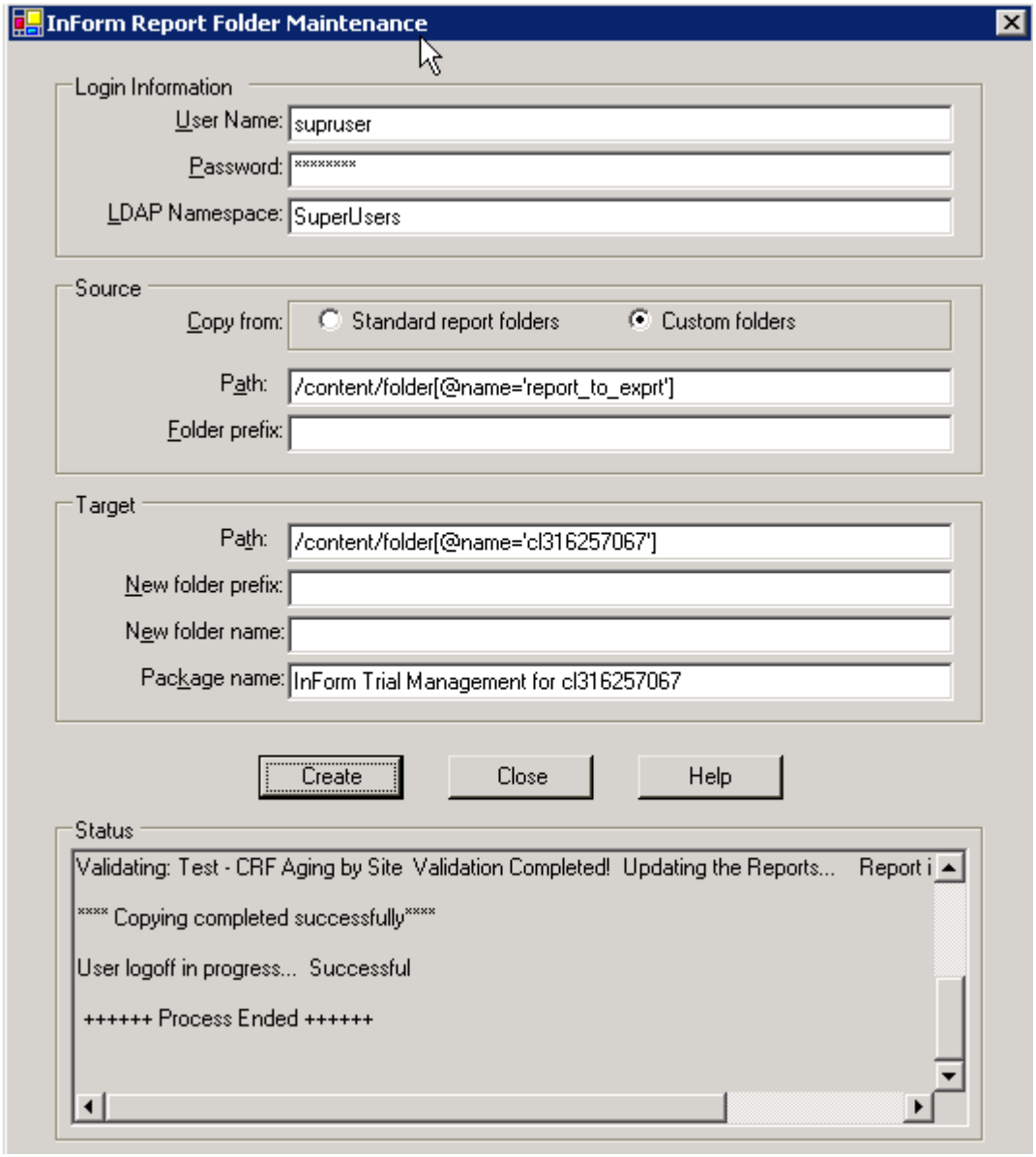
Report copy considerations summary

Source Package	Type of data in report	New package name required?	Report modifications required at the target location?
Trial-specific clinical package	Trial management data only.	Yes.*	No.
Trial-specific clinical package	Clinical data, or a mixture of clinical and trial management data.	Yes.*	Depends on the clinical packages and their elements. You may have to alter the report to ensure that it can be validated and copied to the target location.
Single sponsor, multiple trials			
InForm Trial Management package	Trial management data only.	No.	No.
Multiple sponsors, multiple trials			
InForm Trial Management package	Trial management data only.	Yes. Each sponsor folder should have its own InForm Trial Management package.	No.

*The package must exist before you can copy the folders.

Copying report folders

To copy report folders:



1 On the reporting server, double-click the following file:

`\crn\bin\PFMTRSetupUtil.exe`

The InForm Report Folder Maintenance window appears.

2 In the **Login Information** section, type the following information:

- User name and password to log on to the reporting server.
- Name of the LDAP namespace that is used for authentication by Sun One Directory Server.

3 In the **Source** section, enter the following information:

- **Copy from**—Select the **Standard report folders** radio button to copy standard report folders, or select the **Custom folders** radio button to copy custom report folders.
- **Path**—Type the path for the existing folder structure to copy. Do not include the folder name here. You can copy the source path from the source folder properties.
- **Folder prefix**—Specify the prefix that identifies the folders to copy. For example, if you used T1 as a prefix for all folders belonging to Trial 1, specify T1 in the **Folder prefix** field to instruct the InForm Report Folder Maintenance utility to copy the folders with the T1 prefix.

4 In the **Target** section, enter the following information:

- **Path**—Type the target path for the folder. Do not include the folder name.
- **New Folder prefix**—Specify a prefix for the target folder name. For example, if you are creating the folder structure for Trial 2, you might type T2 as the prefix for the new folder.
- **New Folder name**—Type the name of the target folder if it is different from the name of the source folder. You can specify a new folder name (the folder will be created) or an existing folder.
- **Package name**—Type the name of the new package that should be associated with any trial-specific reports. Since each trial clinical package is unique, reports that contain clinical (trial-specific) data must be associated with a new clinical package when you copy them to a new location.

> **Note:** The package must exist before you perform the copy.

5 Click **Create**.

The InForm Report Folder Maintenance utility begins to copy the folder structure to the new location. The **Status** section displays ongoing status and notifies you when the copy is complete.

6 On the InForm server, identify the top level reporting folder for the trial in the InForm Admin page.

APPENDIX A

Sample Data Import XML

In this appendix

Overview 130

Importing screening and enrollment data 131

Importing new patient clinical data 132

Updating existing patient clinical data 133

Importing new itemset data 134

Editing an existing itemset 136

Deleting data from an itemset 137

Undeleting data from an itemset 138

Adding data to an unscheduled visit 139

Transferring patient records 140

Overview

The import file for the MedML file option is an XML file that contains tags that specify the type of processing to perform during import, and the import data destinations and values. Cut and paste these samples and use any text editor that creates plain text files to edit the data to create your own import files.

- *Importing screening and enrollment data* (on page 131).
- *Importing new patient clinical data* (on page 132).
- *Updating existing patient clinical data* (on page 133).
- *Importing new itemset data* (on page 134).
- *Editing an existing itemset* (on page 136).
- *Deleting data from an itemset* (on page 137).
- *Undeleting data from an itemset* (on page 138).
- *Adding data to an unscheduled visit* (on page 139).
- *Transferring patient records* (on page 140).

Importing screening and enrollment data

This sample file below contains the necessary data tags to import screening and enrollment data for patient XYZ at site PF.

```
<?xml version="1.0"?>
<CLINICALDATA>
<!--

Feature:Screen and Enroll
Description:This file sets up the user to test other XML features
-->

<!-- Screen Patient -->
<SCREEN SITEMNEMONIC="PF">
<DATA TAG="screen.0.patientinitials.patientinitials" VALUE="XYZ"/>
<DATA TAG="screen.0.eligible.eligible" VALUE="yes"/>
<DATA TAG="screen.0.datescreened.date" MONTH="1" DAY="6" YEAR="1999"/>
<DATA TAG="screen.0.dob.dob" MONTH="11" DAY="11" YEAR="1959"/>
</SCREEN>

<!-- Enroll Patient -->
<ENROLL PATIENTINITIALS="XYZ" SITEMNEMONIC="PF" PATIENTNUMBER="BK-XYZ"
ENROLL="TRUE">
<DATA TAG="consent.0.consentdate.date" MONTH="1" DAY="6" YEAR="1999"/>
<DATA TAG="consent.0.patientnumber.patientnumber" VALUE="BK-XYZ"/>
<DATA TAG="inclusion.0.age_inc.yesno" VALUE="1"/>
<DATA TAG="inclusion.0.hyper_inc.yesno" VALUE="1"/>
<DATA TAG="inclusion.0.understand_inc.yesno" VALUE="1"/>
<DATA TAG="inclusion.0.agree_inc.yesno" VALUE="1"/>
<DATA TAG="exclusion.0.secondary_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.malignant_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.allergyhistory_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.myocardial_ex.yesno" VALUE="0"/>
<DATA TAG="exclusion.0.monitor_ex.yesno" VALUE="0"/>
</ENROLL>
</CLINICALDATA>
```

Importing new patient clinical data

The sample file below contains the necessary data tags to import clinical data to patient XYZ at site PF.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Form and Item comments
Description:This example demonstrates basic form and item comments
Requirements:PF_XYZ-Enroll.xml
-->

<!-- Demographics form -->
<PATIENTDATA
PATIENTINITIALS="XYZ"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="DEM"
COMMENT="This is the Demographics Form Comment">
<DATA TAG="DEM.0.GENDER.GENDERRADIO" VALUE="1" COMMENT="Gender Comment"/>
<DATA TAG="DEM.0.DEMDOB.dob" MONTH="2" DAY="14" YEAR="1961" COMMENT="Date Of
Birth Comment"/>
<DATA TAG="DEM.0.RACE.RACEGROUP" CHILDSELECTED="RACETEXT" COMMENT="Race Text
Comment"/>
<DATA TAG="DEM.0.RACE.RACEGROUP.RACETEXT" VALUE="10k" COMMENT="Race Text
Value Comment"/>
<DATA TAG="DEM.0.HEIGHT.PFHT_TC" VALUE="74" UNIT="Inches" COMMENT="Height
Comment"/>
<DATA TAG="DEM.0.WRISTCIRC.PFWC_TC" VALUE="6.0" UNIT="Inches"
COMMENT="Height Unit Comment"/>
<DATA TAG="DEM.0.FRAME.FRAME_CC" VALUE="1"/>
<DATA TAG="SH.0.SMOKE.SMOKERADIO" VALUE="Y"/>
<DATA TAG="SH.0.EVERSMOKED.SMOKERADIO" VALUE="N"/>
<DATA TAG="SH.0.WHATSMOKED.SMOKEGROUPRADIO" CHILDSELECTED="SMOKEGROUP"/>
<DATA TAG="SH.0.WHATSMOKED.SMOKEGROUPRADIO.SMOKEGROUP" VALUE="SMOKES"/>
<DATA TAG="SH.0.WHATSMOKED.SMOKECHECKBOX" VALUE="cigarette,pipe"/>
<DATA TAG="SH.0.HOWMUCHSMOKED.SMOKERADIO2" CHILDSELECTED="NUMTEXT"/>
<DATA TAG="SH.0.HOWMUCHSMOKED.SMOKERADIO2.NUMTEXT" VALUE="10"/>
<DATA TAG="SH.0.YRSSMOKED.SMOKERADIO2" VALUE="NDElement"/>
</PATIENTDATA>

</CLINICALDATA>
```

Updating existing patient clinical data

The sample file below contains the necessary data tags to edit or clear existing data about a patient.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Edit an existing patient's form record
Description:This example demonstrates editing one item, then clearing an
item's value
Requirements:The following must be run prior to this script
PF_XYZ-Enroll.xml
PF_XYZ-DEMandCommentts.xml
-->

<!-- Demographics form -->
<EDITPATIENTDATA
PATIENTINITIALS="XYZ"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="DEM"
REASONOTHER="updated data">
<DATA TAG="DEM.0.HEIGHT.PFHT_TC" VALUE="75" UNIT="Inches" />
</EDITPATIENTDATA>

<!-- Demographics form -->
<EDITPATIENTDATA
PATIENTINITIALS="XYZ"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="DEM"
REASONOTHER="updated data">
<DATA TAG="DEM.0.WRISTCIRC.PFWC_TC" CLEARVALUE="TRUE"/>
</EDITPATIENTDATA>
</CLINICALDATA>
```

Importing new itemset data

The sample file below contains the necessary data tags to import new itemset data.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Submitting Itemset Data
Description:This example demonstrates to submitting itemset data
Requirements:The following must be run prior to this script
PF_XYZ-Enroll.xml
-->

<!-- non explicit itemset index (append to existing itemsets if exists) -->
<PATIENTDATA
PATIENTINITIALS="ITM"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
<DATA TAG="PT.PT.THERAPYTEXT.THERAPYTEXT" VALUE="aspirin"/>
<DATA TAG="PT.PT.DOSAGETEXT.DOSAGETEXT" VALUE="1 tablet"/>
<DATA TAG="PT.PT.DOSEDATE.DOSEDATE" MONTH="12" DAY="23" YEAR="1998"/>
<DATA TAG="PT.PT.DISCPULLDOWN.DISCPULLDOWN" VALUE="Not Effective"/>
</PATIENTDATA>

<!-- Create at itemset index 2 -->
<PATIENTDATA
PATIENTINITIALS="ITM"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
COMMENT="ItemSet #2 - explicit index of 2">
<DATA TAG="PT.PT.THERAPYTEXT.THERAPYTEXT" VALUE="aspirin3"/>
<DATA TAG="PT.PT.DOSAGETEXT.DOSAGETEXT" VALUE="1 tablet"/>
<DATA TAG="PT.PT.DOSEDATE.DOSEDATE" MONTH="12" DAY="23" YEAR="1998"/>
<DATA TAG="PT.PT.DISCPULLDOWN.DISCPULLDOWN" VALUE="Not Effective"/>
</PATIENTDATA>

<!-- Force itemset index 3 -->
<PATIENTDATA
PATIENTINITIALS="ITM"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
ITEMSETINDEX="3">
<DATA TAG="PT.PT.THERAPYTEXT.THERAPYTEXT" VALUE="Bucket"/>
<DATA TAG="PT.PT.DOSAGETEXT.DOSAGETEXT" VALUE="1 tablet"/>
<DATA TAG="PT.PT.DOSEDATE.DOSEDATE" MONTH="12" DAY="23" YEAR="1998"/>
<DATA TAG="PT.PT.DISCPULLDOWN.DISCPULLDOWN" VALUE="Not Effective"/>
</PATIENTDATA>

<!-- Create at itemset index 4 -->
<PATIENTDATA
PATIENTINITIALS="ITM"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
<DATA TAG="PT.PT.THERAPYTEXT.THERAPYTEXT" VALUE="To Delete"/>
<DATA TAG="PT.PT.DOSAGETEXT.DOSAGETEXT" VALUE="To Delete"/>
<DATA TAG="PT.PT.DOSEDATE.DOSEDATE" MONTH="12" DAY="23" YEAR="1998"/>
<DATA TAG="PT.PT.DISCPULLDOWN.DISCPULLDOWN" VALUE="Not Effective"/>
</PATIENTDATA>
```

```
</CLINICALDATA>
```

Editing an existing itemset

The sample file below contains the necessary data tags to edit a row in an existing itemset.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Edit and Itemset item
Description:This example demonstrates editing an itemset item
Requirements:The following must be run prior to this script
PF_XYZ-Enroll.xml
PF_XYZ-ItemSet.xml
-->

<!-- explicit itemset index -->
<EDITPATIENTDATA
PATIENTINITIALS="XYZ"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
ITEMSETINDEX="2"
REASONOTHER="test reason"
<DATA TAG="PT.PT.THERAPYTEXT.THERAPYTEXT" VALUE="aspirinEdit"
COMMENT="ItemLevelComment"/>
</EDITPATIENTDATA>

</CLINICALDATA>
```

Deleting data from an itemset

The sample file below contains the necessary data tags to delete data from an existing itemset.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Delete an Itemset record (DELETEITEMSET tag)
Description:This example demonstrates to Delete an existing itemset record
Requirements:The following must be run prior to this script
PF_XYZ-Enroll.xml
PF_XYZ-ItemSet.xml
PF_XYZ-ItemSetDelete.xml
-->

<!-- non explicit itemset index (append to existing itemsets if exists) -->
<EDITPATIENTDATA
PATIENTINITIALS="XYZ"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
DELETEITEMSET="TRUE"
ITEMSETINDEX="4"
REASONOTHER="test reason"
</EDITPATIENTDATA>

</CLINICALDATA>
```

Undeleting data from an itemset

The sample file below contains the necessary data tags to recover deleted data from an itemset.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Undelete an Itemset record (UNDELETEITEMSET tag)
Description:This example demonstrates to Undelete a previously deleted
itemset.
Requirements:The following must be run prior to this script
PF_XYZ-Enroll.xml
PF_XYZ-ItemSet.xml
PF_XYZ-ItemSetDelete.xml
-->

<EDITPATIENTDATA
PATIENTINITIALS="XYZ"
SITEMNEMONIC="PF"
FORMSETREFNAME="Visit1"
FORMREFNAME="HH"
SECTIONNAME="PT"
ITEMSETNAME="PT"
UNDELETEITEMSET="TRUE"
ITEMSETINDEX="4"
REASONOTHER="test reason"
</EDITPATIENTDATA>

</CLINICALDATA>
```

Adding data to an unscheduled visit

The sample file below contains the necessary data tags to add data to an unscheduled visit for a particular patient.

```
<?xml version="1.0"?>
<CLINICALDATA>

<!--
Feature:Unscheduled Visits
Description:This example demonstrates the sequence of XML submits to use
Unscheduled Visits
Requirements:PF_XYZ-Enroll.xml
-->

<!-- DOV form -->
<PATIENTDATA
PATIENTINITIALS="XYZ" SITEMNEMONIC="PF"
FORMSETREFNAME="UnschVisit" FORMREFNAME="DOV" NEWUNSCHEDVISIT="TRUE">
<DATA TAG="DOV.0.DOV.DOV" MONTH="2" DAY="1" YEAR="1999"/>
</PATIENTDATA>

<!-- VitalSigns form -->
<PATIENTDATA
PATIENTINITIALS="XYZ" SITEMNEMONIC="PF" FORMSETINDEX="1"
FORMSETREFNAME="UnschVisit" FORMREFNAME="VS">
<DATA TAG="VS.0.DATEASSESS.COMMONDATE" MONTH="3" DAY="1" YEAR="1999"/>
<DATA TAG="VS.0.WEIGHT.PFWT_TC" VALUE="150"  UNIT="Pound"/>
<DATA TAG="VS.0.TEMPTEXT.TEMPTEXT" VALUE="98.7" UNIT="Fahrenheit"/>
<DATA TAG="VS.0.BPREADING.BPREADINGGROUP.SYSTEXT" VALUE="130"/>
<DATA TAG="VS.0.BPREADING.BPREADINGGROUP.DIASTEXT" VALUE="85"/>
</PATIENTDATA>

<!-- DOV form -->
<PATIENTDATA
PATIENTINITIALS="XYZ" SITEMNEMONIC="PF"
FORMSETREFNAME="UnschVisit" FORMREFNAME="DOV" NEWUNSCHEDVISIT="TRUE">
<DATA TAG="DOV.0.DOV.DOV" MONTH="2" DAY="2" YEAR="1999"/>
</PATIENTDATA>

<!-- VitalSigns form -->
<PATIENTDATA
PATIENTINITIALS="XYZ" SITEMNEMONIC="PF" FORMSETINDEX="2"
FORMSETREFNAME="UnschVisit" FORMREFNAME="VS">
<DATA TAG="VS.0.DATEASSESS.COMMONDATE" MONTH="3" DAY="2" YEAR="1999"/>
<DATA TAG="VS.0.WEIGHT.PFWT_TC" VALUE="150"  UNIT="Pound"/>
<DATA TAG="VS.0.TEMPTEXT.TEMPTEXT" VALUE="98.7" UNIT="Fahrenheit"/>
<DATA TAG="VS.0.BPREADING.BPREADINGGROUP.SYSTEXT" VALUE="130"/>
<DATA TAG="VS.0.BPREADING.BPREADINGGROUP.DIASTEXT" VALUE="85"/>
</PATIENTDATA>

</CLINICALDATA>
```

Transferring patient records

This example shows the tags in a MedML file that is used to transfer several patients. Note that each pair of PATIENTSITECHANGE tags defines current and destination site information for one patient.

```
<?xml version="1.0"?>
<CLINICALDATA>
    <PATIENTSITECHANGE REASON="new patient address">
        <NEWSITE SITEMNEMONIC="MCLEAN"/>
        <CURRENTSITE SITEMNEMONIC="PF" PATIENTNUMBER="1003"/>
    </PATIENTSITECHANGE>
```

<!----For patient 1001 the file specifies a new patient number because a patient already exists at the McLean Hospital site with the same patient number.>

```
    <PATIENTSITECHANGE REASON="new patient address">
        <NEWSITE SITENAME="McLean Hospital" PATIENTNUMBER="1002"/>
        <CURRENTSITE SITEMNEMONIC="PF" PATIENTNUMBER="1001"/>
    </PATIENTSITECHANGE>
```

<!----The DUPLICATEORDER tag indicates that patient DDD is the second patient with those initials to be screened at the Phase Forward site.>

```
    <PATIENTSITECHANGE REASON="new patient address">
        <NEWSITE SITEMNEMONIC="MCLEAN"/>
        <CURRENTSITE SITENAME="Phase Forward" PATIENTINITIALS="DDD"
            DUPLICATEORDER="2"/>
    </PATIENTSITECHANGE>
</CLINICALDATA>
```

Index

B

Base study • 63
Build Path From Database dialog box • 20
building an item path • 20

C

CDD
  exporting into • 82
  moving data into • 82
  order of populating • 82
CDD Export Options window • 82
checkboxes • 16
checking
  map against the import file • 27
clinical data • 63
coding export, running • 80
coding mapping validation checks • 64
coding XML • 79
command
  start/wait • 10
command line
  command line • 72
Comment field • 19
comments
  exporting • 82
control
  nested • 16
creating import file • 15
CRF data • 82

D

data
  clinical • 63
  moving into CDD • 82
data and map file, importing • 15
data type consistency • 64
date fields • 15
datetime fields • 16
DEL file format • 96
diagram of export process • 82
duplicate selection values • 64

E

editing
  map files • 17
empty selection values • 64
enrolling • 34
Entered values field • 86
exporting
  into a CDD • 82
  Name Value data file • 86
  Oracle Clinical Upload • 88
  process diagram • 82

F

failure messages • 71
Field Definition dialog box • 19
folders
  initial reporting setup • 115
  reporting setup, multiple sponsors • 116
  reporting setup, multiple trials • 116
format
  of DEL file • 96
  of OA6 file • 96
  of output files • 96

I

Ignore This Field in field definition • 19
import file
  characteristics • 15
  creating • 15
importing • 22
  data and map file • 15
InForm item path • 19
InForm Report Folder Maintenance utility
  about • 119
  copy considerations • 121
  mulitple sponsors • 122
  report validation • 120
InForm Server
  logging off before importing • 58
information messages • 71
Initial reporting folder setup • 115
input field type, specifying • 19
item comments • 17
item path
  building • 20
  entering explicitly • 21

L

logging off InForm Server • 58

M

map file
  navigating • 26
  saving • 30
  specifying and editing • 17

MedML Installer
  definition of • 62
  initial run • 69
  introduction • 62
  loading definitions • 63
  non-strict mode • 64
  output messages • 71
  running • 69, 70
  running from the command line • 72
  strict mode • 64
messages
  failure • 71
  information • 71
  output • 71
  warning • 71
moving data into a CDD • 82
multiple trials • 116
multiple-selection controls • 16, 22

N

navigating the map file • 26
nested controls • 16
non-strict mode, MedML Installer • 64

O

OA6 file format • 96
Oracle Clinical
  definitions of terms • 95
  fields for upload • 95
  upload • 88
Oracle Update Statistics script • 58
output file format • 86, 96
output messages
  MedML Installer • 71

P

packages
  report/package associations • 120
  reporting • 119
  trial-specific clinical • 119
path
  RefName • 20
patient data
  clinical • 35
  entering enrollment • 34
  entering screening • 33
  updating existing • 40
Patient field of Field Definition dialog box • 19
Patient identifier field • 86
patient information • 82
patient record transfer • 51
  sample MedML file • 46
PFConsole • 10
  MedML Installer utility syntax • 72
  using within a script • 10
populating a CDD, order of • 82

R

RefName path
  RefName path • 20, 86
removing XML files from the build • 70
report validation • 120
report/package association • 120
reporting folder structure
  multiple sponsors • 116
reporting folder structure for multiple trials • 116
reporting folders
  initial setup • 115
reporting packages, setting up • 119
running MedML Installer
  for the first time • 69
  from the utility • 69

S

saving the map file • 30
screening • 33
script
  Oracle Update Statistics • 58
  using PFConsole within • 10
selection value validation checks • 64
setting up a trial • 63
site data • 82
Site Mnemonic field • 19
specifying
  input field type • 19
  map files • 17
  submission Type • 18
sponsors, mulitple • 122
sponsors, multiple • 116
start/wait command • 10
Stop WWW Publishing Service command • 58
strict mode, MedML Installer • 64
study database
  defining • 63
  description of • 63
study, Base • 63
submission type
  specifying • 18
  window • 18

T

time fields • 16
transferring patient records • 51
  sample MedML file • 46
trial
  defining components of • 63
  loading definitions • 63
  setting up • 63
trials, multiple • 116
trial-specific clinical package • 119

U

unit symbol for previous field • 19
units • 16

Update Statistics Oracle script • 58
updating
existing patient clinical data • 40

V

validation checks • 64
coding mapping • 64
selection value • 64

W

WWW Publishing Service, stopping • 58

X

XML for coded controls • 79